AIX Version 7.2

Commands

Note

Before using this information and the product it supports, read the information in “Notices” on page 4761.

This edition applies to AIX Version 7.2 and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

About this document

This document provides users and system administrators with complete information about AIX commands.

Highlighting

The following highlighting conventions are used in this document:

Bold	Identifies commands, subroutines, keywords, files, structures, directories, and other items whose names are predefined by the system. Also identifies graphical objects such as buttons, labels, and icons that the user selects.
Italics	Identifies parameters whose actual names or values are to be supplied by the user.
`Monospace`	Identifies examples of specific data values, examples of text similar to what you might see displayed, examples of portions of program code similar to what you might write as a programmer, messages from the system, or information you should actually type.

Case-sensitivity in AIX

Everything in the AIX® operating system is case-sensitive, which means that it distinguishes between uppercase and lowercase letters. For example, you can use the **ls** command to list files. If you type LS, the system responds that the command `is not found`. Likewise, **FILEA**, **FiLea**, and **filea** are three distinct file names, even if they reside in the same directory. To avoid causing undesirable actions to be performed, always ensure that you use the correct case.

ISO 9000

ISO 9000 registered quality systems were used in the development and manufacturing of this product.

a

The following AIX commands begin with the letter *a*.

ac Command

Purpose

Prints connect-time records.

Syntax

/usr/sbin/acct/ac [**-d**] [**-p**] [**-w** *File*] [*User* ...]

Description

The **ac** command prints the total connect time for all users or the connect time for specified users. Records are based on who logged in during the life of the current **wtmp** data file.

Connect-time records are created by the **init** and the **login** programs and are collected in the **/var/adm/wtmp** file, if that file exists. The root user or a member of the **adm** group should create the **/var/adm/wtmp** file with an initial record length of 0 (zero). Records should be processed periodically to keep the file from becoming too full. If the file has not been created, the following error message is returned:

```
No /var/adm/wtmp
```

If the file becomes too full, additional **wtmp** files are created. These files can be printed, if specified with the **-w** flag.

Flags

Item	Description
-d	Creates a printout for each day, from midnight to midnight.
-p	Prints connect-time totals by individual login. Without this flag, a total for the time period is printed.
-w *File*	Specifies a **wtmp** file other than the **/var/adm/wtmp** file.

Security

Access Control: This command should grant execute (x) access to all users.

Examples

1. To obtain a printout of the connect time for all users who logged in during the life of the current **wtmp** data file, enter:

   ```
   /usr/sbin/acct/ac
   ```

2. To obtain a printout of the total connect time for users `smith` and `jones`, as recorded in the current **wtmp** data file, enter:

   ```
   /usr/sbin/acct/ac smith jones
   ```

3. To obtain a printout of the connect-time subtotals for users `smith` and `jones`, as recorded in the current **wtmp** data file, enter:

   ```
   /usr/sbin/acct/ac  -p smith jones
   ```

Files

Item	Description
/usr/sbin/acct/ac	Contains the **ac** command.
/var/adm/wtmp	Contains the active data file for the collection of connect-time records.

accept, reject Command

Purpose

Accepts/rejects print requests.

Syntax

accept *Destinations*

reject [**-r** *Reason*] *Destination*

Description

The **accept** command allows the queuing of print requests for the named *Destinations*. A *Destination* can be either a printer or a class of printers. To find out the status of a destination, run **lpstat -a** command.

The **reject** command prevents queuing of print requests for the named *destinations*. A *destination* can be either a printer or a class of printers. To find out the status of a destination, run **lpstat -a** command.

If you enter `accept -?` or `reject -?`, the system displays the command usage message and returns 0.

Flags

Item	Description
-r *Reason*	Assigns a *Reason* for rejection of requests. The *Reason* applies to all of the specified *Destination*s. The **lpstat -a** command reports the reason. If it contains blanks, *Reason* must be enclosed in quotes. The default reason is `unknown reason` for existing destinations, and `new destination` for destinations just added to the system but not yet accepting requests.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

/var/spool/lp/*

acctcms Command

Purpose

Produces command-usage summaries from accounting records.

Syntax

/usr/sbin/acct/acctcms [**-t** | **-a** [**-o**] [**-p**]] [**-c**] [**-j**] [**-n**] [**-s**] [*File* ...]

Description

The **acctcms** command reads each file specified by the *File* parameter, adds and sorts all records for identically named processes, and writes the records to standard output. By default, the output file is in binary format. Input files are usually in the **acct** file format.

When you use the **-o** and **-p** flags together, the **acctcms** command produces a report that combines prime and nonprime time. Prime and nonprime times are defined by entries in the **/etc/acct/holidays** file. Prime times are assumed to be the period when the system is most active, such as weekdays. Saturdays and Sundays are always nonprime time for the accounting systems, as are any holidays that you specify in the **/etc/acct/holidays** file. All the output summaries are of total usage, except for number of times run, CPU minutes, and real minutes, which are split into prime and nonprime minutes.

Flags

Item	Description

-a Displays output in ASCII summary format rather than binary summary format. Each output line contains the command name, the number of times the command was run, total kcore time (memory measurement in kilobyte segments), total CPU time, total real time, mean memory size (in K-bytes), mean CPU time per invocation of the command, and the CPU usage factor. The listed times are all in minutes. The **acctcms** command normally sorts its output by total kcore minutes. The unit kcore minutes is a measure of the amount of memory used (in kilobytes) multiplied by the amount of time it was in use. This flag cannot be used with the **-t** flag.

Use the following options only with the -a option:

-o
Displays a command summary of non-prime time commands.

-p
Displays a command summary of prime time commands.

When you use the **-o** and **-p** flags together, the **acctcms** command produces a report that combines prime and non-prime time. Prime and non-prime times are defined by entries in the **/etc/acct/holidays** file. Prime times are assumed to be the period when the system is most active, such as weekdays. Saturdays and Sundays are always non-prime time for the accounting systems, as are any holidays that you specify in the **/etc/acct/holidays** file. All the output summaries are of total usage, except for number of times run, CPU minutes, and real minutes, which are split into prime and non-prime minutes.

The default items have the following headings in the output:

```
          TOTAL COMMAND SUMMARY


COMMAND  NUMBER  TOTAL     TOTAL    TOTAL
NAME     CMDS    KCOREMIN  CPU-MIN  REAL-MIN


MEAN    MEAN     HOG     CHARS   BLOCKS
SIZE-K  CPU-MIN  FACTOR  TRNSFD  READ
```

-c Sorts by total CPU time rather than total kcore minutes. When this flag is used with the **-n** flag, only the **-n** flag takes effect.

-j Combines all commands called only once under the heading `other`.

-n Sorts by the number of times the commands were called. When this flag is used with the **-c** flag, only the **-n** flag takes effect.

-o Displays a command summary of nonprime time commands. You can use this flag only when the **-a** flag is used.

-p Displays a command summary of prime time commands. You can use this flag only when the **-a** flag is used.

-s Assumes that any named files that follow this flag are already in binary format.

-t Processes all records as total accounting records. The default binary format splits each field into prime and nonprime time sections. This option combines the prime and non-prime time parts into a single field that is the total of both, and provides upward compatibility with old style acctcms binary summary format records. This flag cannot be used with the **-a** flag.

Security

Access Control: This command should grant execute (x) access only to members of the **adm** group.

Examples

To collect daily command accounting records in a `today` file and maintain a running total in a `total` file, add the following to a shell script:

```
acctcms File . . . > today
cp total previoustotal
acctcms -s today previoustotal > total
acctcms -a -s total
```

The *File* parameters that you specify are redirected to a file called `today`, added to the previous total (in a file renamed `previoustotal`) to produce a new total (called `total`). All files are binary files. In the last line, the **-a** flag displays the `total` file in ASCII format so you can view the report.

Files

Item	Description
/etc/acct/holidays	Specifies prime and nonprime time for accounting records.
/usr/sbin/acct/acctcms	Contains the **acctcms** command.

acctcom Command

Purpose

Displays summaries of process-accounting records for selected processes.

Syntax

/usr/sbin/acct/acctcom [[**-q** | **-o** *File*] | [**-a**] [**-b**] [**-c** *Classname*] [**-f**] [**-h**] [**-i**] [**-k**] [**-m**] [**-r**] [**-t**] [**-v**] [**-w** [-X] [**-W**]] [**-C** *Seconds*] [**-g** *Group*] [**-H** *Factor*] [**-I** *Number*] [**-l** *Line*] [**-n** *Pattern*] [**-O** *Seconds*] [**-u** *User*] [**-e** *Time*] [**-E** *Time*] [**-s** *Time*] [**-S** *Time*] [**-@** [*WparName*]] [*File ...*]

Description

The **acctcom** command reads process accounting records from files specified by the *File* parameter from standard input or from the **/var/adm/pacct** file. Then the **acctcom** command writes the records you request to standard output. This command is stored in the **/usr/sbin/acct** directory, for access by all users.

If you do not specify a *File* parameter and if standard input is assigned to a workstation or to the **/dev/null** file, as when a process runs in the background, the **acctcom** command reads the **/var/adm/pacct** file.

If you specify a *File* parameter, the **acctcom** command reads each file chronologically by process completion time. Usually, the **/var/adm/pacct** file is the current file that you want the **acctcom** command to examine. Because the **ckpacct** procedure keeps this file from growing too large, a busy system may have several **pacct** files. All but the current file have the path name **/var/adm/pacct***?*, where *?* (question mark) represents an integer.

Each record represents one completed process. The default display consists of the command name, user name, tty name, start time, end time, real seconds, CPU seconds, and mean memory size (in kilobytes). These default items have the following headings in the output:

```
COMMAND                    START   END    REAL    CPU     MEAN
NAME    USER    TTYNAME    TIME    TIME   (SECS)  (SECS)  SIZE(K)
```

If a process was run by the root user, the process name is prefixed with a # (pound sign). If a process is not assigned to a known workstation (for example, when the **cron** daemon runs the process), a ? (question mark) appears in the TTYNAME field.

Note:

1. The **acctcom** command only reports on processes that have finished. Use the **ps** command to examine active processes.
2. If a specified time is later than the current time, it is interpreted as occurring on the previous day.

Flags

Item	Description
-a	Shows some average statistics about the processes selected. The statistics are displayed after the output records.
-b	Reads backwards, showing the most recent commands first. This flag has no effect when the **acctcom** command reads standard input.
-c *Classname*	Selects processes belonging to the specified class. **Note:** Accounting data cannot be retrieved for a deleted class.
-C *Seconds*	Shows only processes whose total CPU time (system time + user time) exceeds the value specified by the *Seconds* variable.
-e *Time*	Selects processes existing at or before the specified time. You can use the current locale to specify the order of hours, minutes, and seconds. The default order is *hh:mm:ss*.
-E *Time*	Selects processes ending at or before the specified time. You can use the current locale to specify the order of hours, minutes, and seconds. The default order is *hh:mm:ss*. If you specify the same time for both the **-E** and **-S** flags, the **acctcom** command displays the processes that existed at the specified time.
-f	Displays two columns related to the `ac_flag` field of the **acct.h** file: the first indicates use of the **fork** command to create a process, the second indicates the system exit value.
-g *Group*	Selects processes belonging to the specified group. You can specify either the group ID or the group name.
-h	Instead of mean memory size, shows the fraction of total available CPU time consumed by the process (hog factor). This factor is computed as: `(total CPU time) / (elapsed time)`
-H *Factor*	Shows only the processes that exceed the value of the *Factor* parameter. This factor, called the hog factor, is computed as: `no(total CPU time) / (elapsed time)`
-i	Displays columns showing the number of characters transferred in read or write operations (the I/O counts).
-k	Instead of memory size, shows total kcore minutes (memory measurement in kilobyte segments used per minute of run time).
-l *Line*	(lowercase L) Shows only processes belonging to workstation **/dev/***Line*.
-I *Number*	(uppercase i) Shows only processes transferring more than the specified number of characters.
-m	Shows mean main-memory size. This is the default. The **-h** flag or **-k** flag turn off the **-m** flag.

Item	Description
-n *Pattern*	Shows only commands matching the value of the *Pattern* variable, where *Pattern* is a regular expression. Regular expressions are described in the **ed** command. In addition to the usual characters, the **acctcom** command allows you to use a + (plus sign) as a special symbol for the preceding character.
-o *File*	Copies selected process records to the specified file, keeping the input data format. This flag suppresses writing to standard output. This flag cannot be used with the **-q** flag.
-O *Seconds*	Shows only processes with CPU system time exceeding the specified number of seconds.
-q	Displays statistics but not output records. The statistics are the same as those displayed using the **-a** flag. The **-q** flag cannot be used with the **-o** flag.
-r	Shows CPU factor. This factor is computed as: `(user-time) / (system-time + user-time)`
-s *Time*	Shows only those processes that existed on or after the specified time. You can use the current locale to specify the order of hours, minutes, and seconds. The default order is *hh:mm:ss*.
-S *Time*	Shows only those processes starting at or after the specified time. You can use the current locale to specify the order of hours, minutes, and seconds. The default order is *hh:mm:ss*.
-t	Shows separate system and user CPU times.
-u *User*	Shows only processes belonging to the specified user. Enter one of the following for the *User* variable: a user ID, a login name to be converted to a user ID, a # (pound sign) to select processes run by the root user, or a ? (question mark) to select processes associated with unknown user IDs.
-v	Eliminates column headings from the output.
-w	Displays the class names to which the processes belong.
-W	Prints all available characters of each user name instead of truncating to the first 8 characters. The output is also widened to 132 characters allowing the user name to use the additional space. The **-W** option is mutually exclusive with the **-X** option. When both flags are used the second flag is ignored.
-X	Print all available characters of each user name instead of truncating to the first 8 characters. The user name is also moved to the last column of the output. The **-X** option is mutually exclusive with the **-W** option. When both flags are used the second flag is ignored.
-@ [*WparName*]	Displays summaries of process-accounting records for selected processes per workload partition. If a workload partition is specified using the *WparName* parameter, the accounting records for the specified workload partition are displayed. If no workload partition is specified, the accounting records for all of the workload partitions are displayed. A workload partition name is displayed for each record. The -@ option is not supported when executed within a workload partition.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display information about processes that exceed 2 seconds of CPU time, enter:

   ```
   /usr/sbin/acct/acctcom -O 2 < /var/adm/pacct
   ```

   The process information is read from the **/var/adm/pacct** file.
2. To display information about processes belonging to the `finance` group, enter:

   ```
   /usr/sbin/acct/acctcom -g Finance < /var/adm/pacct
   ```

   The process information is read from the **/var/adm/pacct** file.
3. To display information about processes that belong to the **/dev/console** workstation and that run after 5 p.m., enter:

   ```
   /usr/sbin/acct/acctcom -l /dev/console -s 17:00
   ```

   The process information is read from the **/var/adm/pacct** file by default.
4. To display all information about processes on a machine that has greater than 8 character user names, enter:

   ```
   /usr/sbin/acct/acctcom -X < /var/adm/pacct
   ```

   The process information is read from the **/var/adm/pacct** file.
5. To display information about processes that are run inside the `warpath` WPAR, use the following command:

   ```
   acctcom -@ warpath < /var/adm/pacct
   ```

   The process information is read from the **/var/adm/pacct** file.
6. To display information about processes that are run on all WPARs, use the following command:

   ```
   acctcom -@ < /var/adm/pacct
   ```

   The process information is read from the **/var/adm/pacct** file.

Files

Item	Description
/usr/sbin/acct/acctcom	Contains the **acctcom** command.
/var/adm/pacct	Contains the current process accounting file.
/etc/group	Contains the basic group attributes of groups.
/etc/passwd	Contains the basic attributes of users.

acctcon1 or acctcon2 Command

Purpose

Performs connect-time accounting.

Syntax

acctcon1 [**-l** *File*] [**-o** *File*] [**-p**] [**-t**] [-X]

acctcon2 [-X]

Description

acctcon1

The **acctcon1** command is called by the **runacct** command to convert a sequence of login and logoff records (read from standard input) to a sequence of login session records (written to standard output). Input is normally redirected from the **/var/adm/wtmp** file. The input file can be a file other than **/var/adm/wtmp**, as long as it is in the correct format.

The **acctcon1** command displays the following in ASCII format:

- Login device
- User ID
- Login name
- Prime connect time (seconds)
- Non-prime connect time (seconds)
- Session starting time (numeric)
- Starting date and time (in date/time format)

The **acctcon1** command also maintains a list of ports on which users are logged in. When the **acctcon1** command reaches the end of its input, the command writes a session record for each port that still appears to be active. Unless the **-t** flag is used, the **acctcon1** command assumes that input is a current file and uses the current time as the ending time for each session still in progress.

The summary file generated with the **-l** flag helps an administrator track line usage and identify bad lines. All hang-ups, terminations of the **login** command, and terminations of the login shell cause the system to write logoff records. Consequently, the number of logoffs is often much higher than the number of sessions.

acctcon2

The **acctcon2** command, also called by the **runacct** command, converts a sequence of login session records produced by the **acctcon1** command into connect-time total accounting records. These records are merged with other total accounting records by the **acctmerg** command to produce a daily report.

Flags

Note: The following flags are used with the **acctcon1** command.

Item	Description
-l *File*	(lowercase L) Writes a line-usage summary file showing the line name, the number of minutes used, the percentage of total elapsed time, the number of sessions charged, the number of logins, and the number of logoffs. If you do not specify a file name, the system creates the information in the **/var/adm/acct/nite/lineuse** file.
-o *File*	Writes to the specified file an overall record for the accounting period, giving starting time, ending time, number of restarts, and number of date changes. If you do not specify a file name, the system creates the **/var/adm/acct/nite/reboots** file.
-p	Displays only input. Line name, login name, and time are shown in both numeric and date/time formats. Without the **-p** flag specified, the **acctcon1** command would display input, converting input to session records, and write reports.
-t	Uses the last time found in the input as the ending time for any current processes. This, rather than current time, is necessary in order to have reasonable and repeatable values for files that are not current.

Item	Description
-X	Prints and processes all available characters for each user name instead of truncating to the first 8 characters. **Note:** The following flag can be used with both the **acctcon1** and **acctcon2** commands.

Security

Access Control: These commands should grant execute (x) access only to members of the **adm** group.

Examples

1. To convert a sequence of login records (in the **/var/adm/wtmp** file) to a sequence of login session records (stored in the **/var/adm/logsess** file), include the following in a shell script:

   ```
   acctcon1 -t -l/var/adm/acct/nite/lineuse \
   -o/var/adm/acct/nite/reboots \
   </var/adm/wtmp > /var/adm/logsess
   ```

   The login session reports show an ending time that corresponds with the last time input was provided. Two reports are generated: a line-usage summary file named **/var/adm/acct/nite/lineuse**, an overall record for the accounting period, reported in the **/var/adm/acct/nite/reboots** file.
2. To convert a series of login session records (in the **/var/adm/acct/nite/ctmp** file) to a total accounting record (stored in the **/var/adm/logacct** file), include the following in a shell script:

   ```
   acctcon2 < /var/adm/acct/nite/ctmp \
    > /var/adm/logacct
   ```

Files

Item	Description
/usr/sbin/acct/acctcon1	Contains the **acctcon1** command.
/usr/sbin/acct/acctcon2	Contains the **acctcon2** command.
/var/adm/wtmp	Contains connect-time accounting data, including login, logout, and shutdown records.

acctctl Command

Purpose

Controls advanced accounting.

Syntax

acctctl fadd *file size*

acctctl frm *file*

acctctl freset *file*

acctctl fquery [*file*]

acctctl fswitch [*file*]

acctctl isystem {*time*|off}

acctctl iprocess {*time*|off}

acctctl agproc {on|off}

acctctl agke {on|off}

acctctl agarm {on|off}

acctctl trquery [*trid*] [**-@** [*wpar*]]

acctctl tron *trid* [**-@** *wpar*]

acctctl troff *trid* [**-@** *wpar*]

acctctl email {on|off|*addr*}

acctctl on [**-@** [*wpar*]]

acctctl off [**-@** [*wpar*]]

acctctl [**-@** [*wpar*]]

acctctl turacct {on|off}

Description

The administration of Advanced Accounting (AACCT) is organized around the following high level tasks, which are mostly performed by the **acctctl** command.

- Manage Accounting Data Files.
- Manage Project Definitions and Assignments.
- Manage Transactions.
- Manage Advanced Accounting Subsystem.

The -@ option is not supported when executed within a workload partition.

Managing Accounting Data Files

The first task is centered around file management. Files are pre-allocated and registered with the AACCT subsystem, so that it can continuously stream accounting data to these files. When an accounting file is filled, AACCT automatically switches to the next available registered file. If there is no such file, then incoming data might be lost, unless the administrator or the billing application quickly reacts to the problem.

Messages are sent alerting the administrator to the status of files, so that he can avoid these types of problems before they occur. The best approach is to allocate sufficient file space up front. Messages are sent, when a file approaches the full state, and when the system automatically switches to another file. Messages are sent by way of the syslog facility and email. These subsystems have to be correctly configured in order to receive messages.

When the system runs out of accounting files, it internally buffers accounting data, so data is not immediately lost. If the administrator does not respond in time and data is lost, then the system internally maintains some statistics about the outage, which it logs to the accounting subsystem, after the condition has been corrected.

Before starting AACCT, the system administrator should create the accounting files that will be needed on the system. The number and size of these files is workload dependent, so the administrator should choose values that are appropriate for the specific installation. The only recommendation is that at least two files be created, so that AACCT can remain active at all times.

The following commands are provided for managing files:

Item	Description
acctctl fadd *file size*	Allocates and defines an accounting file with specified filename and size. The size is in megabytes.

Item	Description
acctctl frm *file*	Removes the specified accounting file from the accounting subsystem. This will not remove the file from the file system.
acctctl freset *file*	Indicates that the specified file can now be reused by the accounting subsystem.
acctctl fquery [*file*]	Queries the state and current utilization of the specified file, if supplied, or all accounting files otherwise.
acctctl fswitch [*file*]	Forces accounting to switch to a new accounting file. The new file can be optionally specified.

All files must be fully qualified path names. When creating a file, ensure that the file system has enough space.

Managing Project Definitions and Assignments

The second task, Manage Project Definitions and Assignments, is supported through the projctl command. Projects are optional.

Managing Transactions

The third task, Manage Transactions, is designed to control the type of accounting data that is produced, which is configuration dependent, because applications and middleware can provide transactions. The following types of accounting are supported on all systems:

- Process
- Disk
- Network interfaces
- File systems
- System (provides global CPU and memory use)

Administrative control over these sources of accounting data is provided by enabling or disabling the accounting records that they produce. Each accounting record is assigned a unique identifier, so that report and analysis commands can apply the appropriate templates when processing the accounting file. These identifiers also serve to name the different types of accounting that is supported and are specified as parameters to the transaction specific commands. Identifiers are listed in the **sys** file.

The following commands are provided for managing transactions:

Item	Description
acctctl trquery [*trid*] [**-@** [*wpar*]]	Queries the state and name of the specified *trid*, if supplied, or of all trids, otherwise. If you specify the **-@** option without the *wpar* parameter, query trids in all active workload partitions. If you specify the **-@** option with the *wpar* parameter, query trids for the specified workload partition only.
acctctl tron *trid* [**-@** *wpar*]	Enables the specified transaction. If you specify the **-@** option with the *wpar* parameter, enable the transaction in the specified workload partition only.
acctctl troff *trid* [**-@** *wpar*]	Disables the specified transaction. If you specify the **-@** option with the *wpar* parameter, disable the transaction in the specified workload partition only.

By default, all transactions identifiers are enabled.

Not all transaction identifiers can be disabled, because some of them are derived types and are dependent on other transactions. For example, the process aggregation record is dependent on the

process record, so it can't be disabled by itself. Aggregation can be enabled or disabled, and process accounting can be enabled or disabled, but the transaction identifier that corresponds to the aggregated process record can't be disabled. Aggregation is a convenience in the sense that it sums up data internally, so that fewer records are produced. In some cases, data aggregation is provided to simplify data management.

Managing the Advanced Accounting Subsystem

The fourth task, Manage Advanced Accounting Subsystem, is concerned with controlling the execution environment of the subsystem itself. Sub-tasks are oriented towards configuring, running, stopping, and querying AACCT.

The following commands are provided for managing the subsystem:

Item	Description
acctctl email {on|off|*addr*}	Sets up e-mail notifications. If given the **on** subcommand, the last used e-mail address will be used. The e-mail address is limited to 80 characters. Mail must be configured for e-mail notification to function.
acctctl iprocess {*time*|off}	Enables process interval accounting every *time* minutes or disables process interval accounting entirely.
acctctl isystem {*time*|off}	Enables system interval accounting every *time* minutes or disables system interval accounting entirely.
acctctl agproc {on|off}	Enables or disables system-wide aggregation for processes.
acctctl agke {on|off}	Enables or disables system-wide aggregation for third party kernel extensions.
acctctl agarm {on|off}	Enables or disables system-wide aggregation for ARM transactions.
acctctl dump *pid*	Writes the accounting record for the named process into the accounting file.
acctctl on [**-@** [*wpar*]]	Starts Advanced Accounting. If you specify the **-@** option without the *wpar* parameter, start Advanced Accounting for all active workload partitions. If you specify the **-@** option with the *wpar* parameter, start Advanced Accounting for the specified workload partition only.
acctctl off [**-@** [*wpar*]]	Stops Advanced Accounting. If you specify the **-@** option without the *wpar* parameter, stop Advanced Accounting for all active workload partitions. If you specify the **-@** option with the *wpar* parameter, stop Advanced Accounting for the specified workload partition only.
acctctl [**-@** [*wpar*]]	Queries overall accounting state. If you specify the **-@** option without the *wpar* parameter, query the Advanced Accounting state for all active workload partitions. If you specify the **-@** option with the *wpar* parameter, query the Advanced Accounting state of the specified workload partition only.

Item	Description
acctctl turacct {on|off}	Enables or disables the accounting based on Scaled Performance Utilization Resources Register (SPURR) in turbo mode.

Exit Status

This command returns the following exit values:

Item	Description
0	The command executed successfully.
>0	An error occurred.

Security

Root authority is required to use this command.

Data files are created by this command. These files are owned by root, but are readable by members of the adm group.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display status, type:

   ```
   acctctl
   ```

   Output similar to the following is displayed:

   ```
   Advanced Accounting is not running.
   Email notification is off.
   The current email address to be used is not set.
   Process Interval Accounting is off.
   System Interval Accounting is off.
   System-wide aggregation of process data is off.
   System-wide aggregation of third party kernel extension data is off.
   System-wide aggregation of ARM transactions is off.
   Files: 0 defined, 0 available.
   ```

2. To turn on accounting, type:

   ```
   acctctl on
   ```

3. To add a 200 MB data file, type:

   ```
   acctctl fadd /var/aacct/acctdata1 200
   ```

4. To enable the process interval so that it collects data every 2 hours, type:

   ```
   acctctl iprocess 120
   ```

5. To set process aggregation, type:

   ```
   acctctl agproc on
   ```

6. To enable e-mail notification, type:

   ```
   acctctl email on
   ```

7. To specify an e-mail address for notification, type:

```
acctctl email user@company.com
```

8. To turn on accounting for WPARs on system, use the following command:

```
acctctl on -@
```

9. To list trids specific to a WPAR that is named `wpar1`, use the following command:

```
acctctl trquery -@ wpar1
```

A similar result will be displayed as follows:

```
NUMBER          STATE           NAME
    33          disabled       wpar-proc
    34          disabled       wpar-agg_proc
    35          disabled       wpar-agg_app
    36          enabled        wpar-system
    38          enabled        wpar-file
    39          enabled        wpar-netif
    44          disabled       wpar-agg_KE
```

Location

/usr/bin/acctctl

Files

Item	Description
/var/aacct	Default directory for accounting data files.
/var/aacct/acctdata	Default accounting data file.

Data files can be created in other locations by the system administrator.

acctdisk, acctdusg Command

Purpose

Performs disk-usage accounting.

Syntax

/usr/sbin/acct/acctdisk

/usr/sbin/acct/acctdusg [**-u** *File*] [**-p** *File*] [-X]

Description

The **acctdisk** and **acctdusg** commands are called by the **dodisk** command to perform disk-usage accounting. Usually, this procedure is initiated when the **cron** daemon runs the **dodisk** command.

Normally, the output of the **diskusg** command becomes the input of the **acctdisk** command. If a more thorough but slower version of disk accounting is needed, use the **dodisk -o** command to call the **acctdusg** command instead of the **diskusg** command.

Accounting is only done for files on the local file system for local users. System administrators who want to count remote users (such as YP clients or diskless clients) should use the **acctdusg -p** command.

acctdisk

The **acctdisk** command reads the output lines of the **diskusg** or **acctdusg** commands from standard input, converts each individual record into a total accounting record, and writes the records to standard output. These records are merged with other accounting records by the **acctmerg** command to produce the daily accounting report.

acctdusg

The **acctdusg** command is called by using the **dodisk -o** command, when a slow and thorough version of disk accounting is needed. Otherwise, the **dodisk** command calls the **diskusg** command.

The **acctdusg** command reads a list of files from standard input (usually piped from a **find / -print** command), computes the number of disk blocks (including indirect blocks) allocated to each file owner, and writes an individual record for each user to standard output. By default, the command searches for login names and numbers in the **/etc/passwd** file. You can search other files by specifying the **-p** *File* flag and variable. Each output record has the following form:

```
uid login #blocks
```

The `#blocks` value is the number of 1KB blocks utilized by the user.

Flags

Item	Description
-p *File*	Searches the specified file for login names and numbers, instead of searching the **/etc/passwd** file.
-u *File*	Places, in the specified file, records of the file names that are exempt from charges.
-X	Turns on long username support.

Security

Access Control: These commands should grant execute (x) access only to members of the **adm** group.

Examples

1. To start normal disk accounting procedures, add a line similar the following to a **crontab** file so that the **cron** daemon runs disk accounting commands automatically:

   ```
   0 2 * * 4 /usr/sbin/acct/dodisk
   ```

   In this example, the **dodisk** procedure runs at 2 a.m. (`0 2`) every Thursday (4) and the **dodisk** procedure calls the **diskusg** and **acctdisk** commands to write disk usage records to the **/usr/adm/acct/nite/dacct** file.
2. To start a thorough disk accounting procedure, add a line similar the following to a **crontab** file so that the **cron** daemon runs disk accounting commands automatically:

   ```
   0 2 * * 4 /usr/sbin/acct/dodisk -o
   ```

   In this example, the **dodisk** procedure runs at 2 a.m. (`0 2`) every Thursday (4) and the **dodisk** procedure calls the **acctdusg** and **acctdisk** commands to write disk usage records to the **/var/adm/acct/nite/dacct** file.

Files

Item	Description
/usr/sbin/acct/acctdisk	Contains the **acctdisk** command.
/usr/sbin/acct/acctdusg	Contains the **acctdusg** command.

Item	Description
/etc/passwd	Contains the basic attributes of user.
/usr/sbin/acct	Directory holding all accounting commands.

acctmerg Command

Purpose

Merges total accounting files into an intermediary file or a daily report.

Syntax

/usr/sbin/acct/acctmerg [**-a** [*Specification*]] [**-h** [*Specification*]] [**-i** [*Specification*]] [**-p** [*Specification*]] [**-q** *Filename*] [**-v** [*Specification*]] [-X] [**-t**] [**-u**] [*File ...*]

Description

The **acctmerg** command merges process, connect-time, fee, disk-usage, and queuing (printer) total accounting records (in **tacct** binary or **tacct** ASCII format, **tacctx** binary, or **tacctx** ASCII format) and then writes the results to standard output. (See the **tacct** structure in the **acct** File Format for a description of the total accounting format or **/usr/include/sys/tacct.h** for a description of the **tacctx** format). The **acctmerg** command reads the total accounting records from standard input and from the additional files (up to nine) specified by the *File* parameter. The **acctmerg** command then merges the records by identical keys, usually a user ID and name. To facilitate storage, the **acctmerg** command writes the output in binary format unless you use either the **-a**, **-v**, or **-p** flag.

The **acctmerg** command is called by the **runacct** command to produce either an intermediate report when one of the input files is full, or to merge the intermediate reports into a cumulative total. The intermediate report is stored in the **/var/adm/acct/nite(x)/daytacct** file. The cumulative report is stored in the **/var/adm/acct/sum(x)/tacct** file. The cumulative total is the source from which the **monacct** command produces the ASCII-format monthly summary report. The monthly summary report is stored in the **/var/adm/acct/fiscal** file.

The *Specification* variable allows you to select input or output fields, as illustrated in Example 1. A field specification is a comma-separated list of field numbers, in the order specified in the **tacct(x)** structure in the **acct** File Format. Field ranges may be used, with array sizes taken into account, except for the *ta_name* characters. In the following example:

```
-h2-3,11,15-13,2
```

The **-h** flag causes column headings to display for the following types of data, in this order:

- login name (2)
- prime CPU (3)
- connect time (11)
- fee (15)
- queuing system (14, as implied in the range)
- disk usage data (13)
- the login name again (2)

The default displays all fields, otherwise specified as 1-18 or 1-, and produces wide output lines containing all the available accounting data.

Queueing system, disk usage, or fee data can be converted into **tacct** records by using the **acctmerg -i** *Specification* command.

The **tacct** fields are:

No. Header	Description
1 UID	User ID number.
2 LOGIN NAME	Login name of user.
3 CPU PRIME	Cumulative CPU minutes during prime hours.
4 CPU NPRIME	Cumulative during non-prime hours.
5 KCORE PRIME	Cumulative minutes spent in the kernel during prime hours.
6 KCORE NPRIME	Cumulative during non-prime hours.
7 BLKIO PRIME	Cumulative blocks transferred during prime hours.
8 BLKIO NPRIME	Cumulative during non-prime hours.
9 RW/WR PRIME	Cumulative blocks read/written during prime hours.
10 RW/WR NPRIME	Cumulative during non-prime hours.
11 CONNECT PRIME	Cumulative connect time (minutes) during prime hours.
12 CONNECT NPRIME	Cumulative during non-prime hours.
13 DISK BLOCKS	Cumulative disk usage.
14 PRINT	Queuing system charges. (pages)
15 FEES	Fee for special services.
16 # OF PROCS	Count of processes.
17 # OF SESS	Count of login sessions.
18 # OF SAMPLES	Count of count of disk samples.

Flags

Item	Description
-a[*Specification*]	Produces output in the form of ASCII records.
-h[*Specification*]	Displays column headings. This flag implies the **-a** flag, but is effective with **-p** or **-v**.
-i[*Specification*]	Expects input files composed of ASCII records, which are converted to binary records.
-p[*Specification*]	Displays input without processing. The output is in ASCII format.
-q *Filename*	Reads the specified **qacct** file (**accrec.h** file format) and produces output records sorted by user ID and user name. These records contain the user ID, user name, and number of pages printed.
-t	Produces a single record that contains the totals of all input.
-u	Summarizes by user ID rather than by user name.
-v[*Specification*]	Produces output in ASCII format, with more precise notation for floating-point numbers.
-X	Prints and processes all available characters for each user name instead of truncating to the first 8 characters.

Security

Access Control: This command should grant execute (x) access only to members of the **adm** group.

Examples

1. To merge disk accounting file `dacct` with field specification `-i1-2,13,18` into an existing total accounting file, `tacct`, enter:

   ```
   acctmerg -i1-2,13,18 <dacct | acctmerg tacct >output
   ```

   The **acctmerg** command reads the field specifications for the user ID, login name, number of blocks, and number of disk samples (`i1-2,13,18`) from the **dacct** file, merges this information with a **tacct** record, and writes the result to standard output.
2. To make repairs to the **tacct** format file `jan2.rpt`, first enter:

   ```
   acctmerg  -v <Jan.2.rpt >jan2.tmp
   ```

   Now edit the file `jan2.tmp` as desired. This command redirects the content of `Jan2.rpt` to `Jan2.tmp`, with the output in ASCII format.
3. To redirect `Jan2.tmp` to `Jan2.rpt`, with the output in binary record format, enter the following command:

   ```
   acctmerg  -i <jan2.tmp >jan2.rpt
   ```

Files

Item	Description
/usr/sbin/acct/acctmerg	Contains the **acctmerg** command.
/usr/include/sys/acct.h	Contains the **acct** and **tacct** file formats.
/var/adm/acct/nite/daytacct	Contains an intermediate daily total accounting report in binary format.
/var/adm/acct/sum/tacct	Contains the cumulative total accounting report for the month in binary format.
/var/adm/acct/fiscal	Contains the monthly accounting summary report, produced from the records in the **/var/adm/acct/sum/tacct** file.

acctprc1, acctprc2, or accton Command

Purpose

Performs process-accounting procedures.

Syntax

/usr/sbin/acct/acctprc1 [*InFile*]

/usr/sbin/acct/acctprc2 [**-X**]

/usr/sbin/acct/accton [[-@] *OutFile*]

Description

The three **acctprc** commands, **acctprc1**, **acctprc2**, and **accton,** are called by the **runacct** command to perform process-accounting shell procedures.

The **acctprc1** command reads records from standard input that are in the **acct** format, adds the login names that correspond to user IDs, and then writes an ASCII record to standard output. This record contains the user ID, login name, prime CPU time, nonprime CPU time, the total number of characters

transferred (in 1024-byte units), the total number of blocks read and written, and mean memory size (in 64-byte units) for each process.

If specified, the *InFile* parameter contains a list of login sessions in **utmp** format, sorted by user ID and login name. If the *File* parameter is not specified, **acctprc1** gets login names from the**/etc/passwd** password file. The information in the *InFile* parameter helps distinguish among different login names that share the same user ID.

The **acctprc2** command reads (from standard input) the records written by the **acctprc1** command, summarizes them by user ID and name, and writes the sorted summaries to standard output as total accounting records.

When the **accton** command is used without parameters, process accounting is turned off. If you specify the *OutFile* parameter (an existing file), process accounting is turned on, and the kernel adds records to that file. You must specify the *OutFile* parameter for process accounting to start. The *OutFile* parameter is not created by the **accton** command. The file specified by the *OutFile* parameter must already exist with the proper group, owner, and permissions. Many shell scripts expect the **/var/adm/pacct** file.

Flags

Item	Description
-X	Process all available characters for each use rname instead of truncating to the first 8 characters. This flag also causes the **acctprc2** command to produce **tacctx** formatted binary records instead of **tacct** binary records. **Note:** This flag can only be used with the **acctprc2** command.
-@	Include workload partition process accounting records in the global WPARs accounting output file. This option is not valid inside a workload partition.

Security

Access Control: These commands should grant execute (x) access only to members of the **adm** group.

Examples

1. To add a user name to each process-accounting record in a binary file and convert the records to an ASCII file named `out.file`, enter the following commands or use the lines in a shell script:

   ```
   /usr/sbin/acct/acctprc1 < /var/adm/pacct >out.file
   ```

2. To produce a total accounting record of the ASCII output file in example 1, enter the following commands or use the lines in a shell script:

   ```
   /usr/sbin/acct/acctprc2 < out.file > \
   /var/adm/acct/nite/daytacct
   ```

   The resulting file is a binary total accounting file in **tacct** format, containing individual records sorted by user ID. The file `/var/adm/acct/nite/daytacct` is merged with other total accounting records by the **acctmerg** command to produce the daily summary record in the **/var/adm/acct/sum/tacct** file.

3. To turn off process accounting, enter:

   ```
   /usr/sbin/acct/accton
   ```

Files

Item	Description
/usr/sbin/acct/acctprc1	Contains the **acctprc1** command.

Item	Description
/usr/sbin/acct/acctprc2	Contains the **acctprc2** command.
/usr/sbin/acct/accton	Contains the **accton** command.
/etc/accton	Symbolic link to the actual **accton** command directory.
/etc/passwd	Contains the basic user attributes, including the user IDs used by the **acctprc1** command.

acctrpt Command

Purpose

Generates advanced accounting subsystem data reports.

Syntax

`acctrpt` [`-f` *filename*] [`-F`] [`-U` *uid*] [`-G` *gid*] [`-P` *projID*] [`-C` *command*] [`-b` *begin_time*] [`-e` *end_time*] [`-p` *projfile*] [`-n`]

`acctrpt` [`-f` *filename*] [`-F`] `-L` *resource* [`-b` *begin_time*] [`-e` *end_time*]

`acctrpt` [`-f` *filename*] [`-F`] `-T` [`-b` *begin_time*] [`-e` *end_time*]

`acctrpt` { `-c` | `-x` } [`-f` *filename*] [`-p` *projfile*] [`-n`]

`acctrpt` [`-b` *begin_time*] [`-e` *end_time*] [[[`-U` *uid*] [`-G` *gid*] [`-C` *command*] [**-@** *wpar*]] | [`-L` *resource* [**-@** *wpar*]]] [`-n`] [`-f` *filename*]

Description

The `acctrpt` command displays the advanced accounting statistics. advanced accounting subsystem supports process accounting, LPAR accounting, and transaction accounting.

For process accounting, users can generate accounting reports by projects, by groups, by users, by commands, or by a combination of these four identifiers. The command arguments `-U`, `-G`, `-P`, and `-C` command arguments are used to generate process accounting reports. The order in which these arguments are specified affects the order in which the data is displayed in the report. For example, the `acctrpt -U ALL -P ALL` command sorts by UID first and project second.

For LPAR accounting, users can generate accounting reports that describe the system-level use of resources, such as processors, memory, file systems, disks, and network interfaces. The system accounting interval must be enabled to collect accounting statistics for system resources. The `-L` command argument is used to generate LPAR accounting reports.

Note: The `-L` argument provides OS image level statistics, so it can also be used on systems that are not LPAR systems.

For transaction accounting, users can generate accounting reports describing application transactions. Transaction reports provide scheduling and accounting information, such as transaction resource usage requirements. These reports consume data that is produced by applications that are instrumented with the application response and measurement application programming interface (APIs). The `-T` command argument is used to generate transaction accounting reports.

If the `-U`, `-G`, `-P`, `-C`, `-L`, and `-T` command arguments are not specified, individual process accounting records are displayed.

Flags

Item	Description
-@ *wpar*	Specifies the workload partition for which the report is generated. The -@ option is not supported when executed within a workload partition.
-b *begin_time*	Specifies the begin time of an interval. The *begin_time* parameter is a 10-character string in the *MMDDhhmmyy* format, where *MM* is month, *DD* is day, *hh* is hour, *mm* is minute, and *yy* is the last 2 digits of the year. All characters are numeric. If *begin_time* is not specified, all encountered records that were written before *end_time* are considered. If neither *end_time* or *begin_time* is specified, all records are considered.
-C *command*	Displays process accounting statistics for the specified command. More than one command name can be specified using a comma-separated list. Only the first 12 characters of the base command name are considered. To display all commands, specify `-C ALL`.
-c	Displays the project definitions in human readable format.
-e *end_time*	Specifies the end time of an interval. The *end_time* parameter is a 10-character string in the *MMDDhhmmyy* format, where *MM* is month, *DD* is day, *hh* is hour, *mm* is minute, and *yy* is the last 2 digits of the year. All characters are numeric. If *end_time* is not specified, all encountered records that were written after *begin_time* are considered. If neither *end_time* or *begin_time* is specified, all records are considered.
-f *filename*	Specifies the path name of the accounting data file to be used. More than one file can be specified using a comma-separated list. If the `-f` flag is not specified, the `/var/aacct/aacctdata` file is used by default.
-F	Displays information about the specified accounting data file. The report includes the host name, partition name, machine model, and serial number of the system where the accounting data file was generated.
-G *gid*	Displays process accounting statistics for the specified GIDs. More than one GID can be specified using a comma-separated list. To display all GIDs, specify `-G ALL`.

Item	Description
-L *resource*	Displays LPAR accounting statistics for the specified resource. The *resource* parameter must be one of the following values: **cpumem** CPU and memory statistics **filesys** File system statistics **netif** Network interface statistics **disk** Disk statistics **vtarget** VSCSI target statistics **vclient** VSCSI client statistics **ALL** All LPAR resource statistics The -L argument cannot be specified with the -U, -P, -G, -C, or -T flags.
-n	Displays the IDs in numbers. By default, names are displayed.
-P *projID*	Displays process accounting statistics for the specified project ID. More than one project ID can be specified using a comma-separated list. To display all projects, specify -P ALL.
-p *projfile*	Specifies the project definition file to be used to resolve the projects associated with the transaction records. If -p is not specified, the projects are resolved using the currently loaded projects.
-T	Displays transaction accounting statistics. The -T argument cannot be specified with -U, -P, -G, -C, or -L flags.
-U *uid*	Displays process accounting statistics for the specified UIDs. More than one UID can be specified using a comma-separated list. To display all UIDs, specify -U ALL.
-x	Displays the project definitions in the project definition file format.

Exit Status

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To generate a file header report from the `/var/aacct/acctdata` data file, type:

   ```
   acctrpt -F -f /var/aacct/acctdata
   ```

2. To generate process accounting report by Users from the `/var/aacct/acctdata` data file, type:

   ```
   acctrpt -U ALL -f /var/aacct/acctdata
   ```

3. To generate a process accounting report for user ID 256 and user ID 257 and command `uname` from the `/var/aacct/acctdata` data file, type:

   ```
   acctrpt -U 256 257 -C uname -f /var/aacct/acctdata
   ```

4. To generate a process accounting report by projects and by users from the`/var/aacct/acctdata` data file, type:

   ```
   acctrpt -P ALL -U ALL -f /var/aacct/acctdata
   ```

5. To generate CPU and Memory statistics from the `/var/aacct/acctdata` data file, type:

   ```
   acctrpt -L cpumem -f /var/aacct/acctdata
   ```

6. To display the project definitions associated with the accounting records, type:

   ```
   acctrpt -c -f /var/aacct/acctdata
   ```

   Information similar to the following is displayed:

   ```
   PROJNAME        PROJID   AGGR       ORIGIN

   System          0        ENABLED    LOCAL
   ```

7. To display the associated IDs in numbers, type:

   ```
   acctrpt -P ALL -f /var/aacct/acctdata -n
   ```

Standard Output

Based on the `-f` option, the `acctrpt` command displays the following values in the File Header report.

Item	Description
File Name	The full path name of the accounting data file.
Open Date	The timestamp of first transaction record in the data file.
Last Close Date	The timestamp of last transaction record in the data file.
Host Name	The host where the data file was produced.
Partition Name	The partition where the data file was produced.
Partition ID	The partition number where the data file was produced.
System Model	The system model where the data file was produced.
System ID	The system serial number where the data file was produced.

Based on one or more of the `-P`, `-G`, `-U`, or `-C` options, the `acctrpt` command displays the following values in the Process Accounting report.

Item	Description
PROJID	The project name (Project ID).
UID	The user name (User ID).
GID	The group name (Group ID).
CMD	The base name of the executed command.
CNT	The count of transaction records aggregated per row of accounting report.
CPU	The CPU time (in seconds).
LFILE	The local File I/O (in MB).
DFILE	Other File I/O (in MB).
LSOCKET	The local socket I/O (in MB).
RSOCKET	Other socket I/O (in MB).
DMEM	Page seconds of disk pages.
PMEM	Page seconds of real pages.
VMEM	Page seconds of virtual memory.

Based on the `-L cpumem` option, the `acctrpt` command displays the following values in the CPU and Memory LDAP Accounting report.

Item	Description
CNT	The count of transaction records aggregated per row of accounting report.
IDLE	The CPU idle time (in seconds).
IOWAIT	The CPU I/O wait time (in seconds).
SPROC	The system process time (in seconds).
UPROC	The user process time (in seconds).
INTR	The interrupt time (in seconds).
IO	The number of I/Os.
PGSPIN	The number of page swap-ins.
PGSPOUT	The number of page swap-outs.
LGPGUTIL	The average utilization of large page pool.
PGRATE	The average page rate (per second).
PMEMUTIL	The average amount of physical memory that is allocated to an LPAR (in MB).
IOMEMUTIL	The average utilization of I/O memory entitlement (in MB).

Based on the `-L filesys` option, the `acctrpt` command displays the following values in the File Systems LPAR Accounting report.

Item	Description
CNT	The count of transaction records aggregated per row of accounting report.
DEVNAME	The device name.
MOUNTPT	The mount point name.

Item	Description
FSTYPE	The file system type.
RDWR	The number of reads and writes.
OPEN	The number of file opens.
CREATE	The number of file creates.
LOCKS	The number of file locks.
XFERS	The data transferred (in MB).

Based on the `-L netif` option, the `acctrpt` command displays the following values in the Network Interfaces LPAR Accounting report.

Item	Description
CNT	The count of transaction records aggregated per row of accounting report.
NETIFNAME	The network interface name.
NUMIO	The number of I/Os.
XFERS	The data transferred (in MB).

Based on the `-L disk` option, the `acctrpt` command displays the following values in the Disks LPAR Accounting report.

Item	Description
CNT	The count of transaction records aggregated per row of accounting report.
DISKNAME	The disk name.
BLKSZ	The disk block size (in bytes).
XFERS	The number of disk transfers.
READ	The number of reads from the disk.
WRITE	The number of writes to the disk.

Based on the `-L vtarget` option, the `acctrpt` command displays the following values in the VSCSI Targets LPAR Accounting report.

Item	Description
CNT	The count of transaction records aggregated per row of accounting report.
CLIENT#	The client partition number.
SERVERID	The server Unit ID.
UNITID	The device logical unit ID.
BYTESIN	The data in (in MB).
BYTESOUT	The data out (in MB).

Based on the `-L vclient` option, the `acctrpt` command displays the following values in the VSCSI Clients LPAR Accounting report.

Item	Description
CNT	The count of transaction records aggregated per row of accounting report.
CLIENT#	The client partition number.
SERVERID	The server Unit ID.

Item	Description
UNITID	The device logical unit ID.
BYTESIN	The data in (in MB).
BYTESOUT	The data out (in MB).

Based on the `-T` option, the `acctrpt` command displays the following values in the Transaction Accounting report.

Item	Description
PROJID	The project name (Project ID).
CNT	The count of transaction records aggregated per row of accounting report.
CLASS	The account class.
GROUP	The application group name.
NAME	The application name.
TRANSACTION	The transaction name
USER	The user name.
RESPONSE	The response time (in milliseconds).
QUEUED	The queued time (in milliseconds).
USER	The CPU time (in milliseconds).

If you specify the **-@** flag , the **acctrprt** command displays workload partition names in the process accounting report and the LPAR accounting report.

Note: Some of the transaction records displayed by `-U`, `-G`, `-P` and `-C` cannot be aggregated. For example, the transaction records that belong to the transaction ID `TRID_agg_proc` cannot be aggregated on group IDs and command names because these transaction records do not have the respective fields. For such records, the `acctrpt` command displays a `*` (asterisk) character in the command name field and a value of `-2` in the group ID field. It is an indication that these records are not aggregated and the caller has to look up for the command name.

Files

Item	Description
`/usr/bin/acctrpt`	Contains the `acctrpt` command.
`/var/aacct/acctdata`	Contains the default accounting data file.

acctwtmp Command

Purpose

Manipulates connect-time accounting records by writing a **utmp** record to standard output.

Syntax

/usr/sbin/acct/acctwtmp "*Reason*"

Description

The **acctwtmp** command is called by the **runacct** command to write a **utmp** record to standard output. The standard output includes the current date and time, plus a *Reason* string of 11 characters or less that you must enter.

Flags

None.

Parameters

Item	Description
Reason	String of 11 characters or less.

Security

Access Control: These commands should grant execute (x) access only to members of the **adm** group.

Files

Item	Description
/usr/sbin/acct/acctwtmp	Contains the **acctwtmp** command.
/var/adm/wtmp	Contains records of date changes that include an old date and a new date.
/usr/include/utmp.h	Contains history records that include a reason, date, and time.

>lacfo Command

Purpose

Manages the tunable parameters of the Advanced Crypto Facility (ACF).

Syntax

```
acfo [-d] | [-d -t tunable_name]
```

```
acfo -R
```

```
acfo -r tunable_name
```

```
acfo -p [-R | -r -t tunable_name] | [-t tunable_name=value]
```

```
acfo -t tunable_name=value
```

```
acfo [-h] | [-h -t tunable_name]
```

Description

The **acfo** command can display or modify current and persistent tunable parameters of ACF. The ACF tunable parameters determine if the nest (NX) acceleration or in-core cryptographic acceleration must be used in kernel-crypto APIs that are provided by Public Key Cryptography Standards (PKCS) #11.

Currently, the NX and in-core crypto acceleration supports only the Advanced Encryption Standard (AES) for the PKCS #11 subsystem.

The NX cryptographic acceleration is supported on POWER7®+ processor-based servers, and later. The in-core cryptographic acceleration is supported on POWER8® processor-based servers, and later. You cannot turn on the in-core cryptographic acceleration if the server does not support it.

The kernel crypto APIs are used by Encrypting File System (EFS), IP security (IPSec), logical volume encryption, kernel extensions, and the user-space applications that use the AIX PKCS #11 API object, `/usr/lib/pkcs11/ibm_pks11.so`.

The PKCS #11 device driver must be active when you run the **acfo** command.

Persistent tunable parameter values are values that are retained by the tunable parameters across the reboot operation. Persistent tunable parameter values are stored in the ODM database. These values are used by the PKCS #11 device drivers when the *CFG_INIT* command is run to initialize the tunable parameters.

The **acfo** command affects only the system-wide tunable parameters, therefore the **acfo** command is not supported in a Workload Partition (WPAR) environment.

Note: The administrator must not modify the NX or in-core cryptographic acceleration settings when several kernel crypto operations are in progress. Use the **-p** parameter of the **acfo** command to modify the acceleration settings permanently, and then restart the logical partition to apply the changes.

Flags

-a
Displays value of all the ACF tunable parameters, one per line.

-d
Displays all ACF tunable parameter names and current values. When you use the **-d** flag with the **-t** flag, the **acfo** command displays the current values of the specified tunable parameters.

-h
Displays help information about the command and its arguments. When you use the **-t** flag with the **-h** flag, the command displays help information for the specific tunable parameters.

-p
Modifies the current values and next boot values of the tunable parameters permanently. If you do not specify the **-p** flag, only the current values of the tunable parameters are changed; the changes are not persistent across the next boot operation.

-R
Resets all tunable parameters to their default values.

-r *tunable*
Resets specified tunable parameter to its default value.

-t *tunable* [=*new_value*]
Displays the current value of the specified tunable parameter or sets the tunable parameter to the specified value.

Tunable parameters

For default values and range of values of AFC tunable parameters, run the **acfo -h -t** *tunable_name* command. The valid tunable parameter names follow:

nx_enabled
Specifies NX crypto acceleration. A value of 1 enables NX crypto acceleration and a value of 0 disables NX crypto acceleration.

min_sz
Specifies the minimum data size (in bytes) that is suitable for NX crypto acceleration. If the acceleration request requires less data than the specified minimum value, the acceleration request

uses the software implementation such as cryptographic software methods that are executed by the general-purpose CPU. This tunable parameter is applicable only for NX crypto acceleration.

in_core_enabled
Specifies in-core crypto acceleration. A value of 1 enables in-core crypto acceleration and a value of 0 disables in-core crypto acceleration. This tunable parameter precedes the *nx_enabled* tunable parameter.

Security

Note: Only a root user can run the **acfo** command.

Examples

1. To display all AFC tunable parameters names and the corresponding current values, run the following command:

```
acfo -d
nx_enabled              : 1.
min_sz                  : 1024.
in_core_enabled         : 0.
```

2. To set minimum data size of AFC NX crypto acceleration to a non-persistent value of 1024 bytes, run the following command:

```
acfo -t min_size=1024
```

3. To turn off NX crypto acceleration permanently, run the following command:

```
acfo -p -t nx_enabled=0
```

aclconvert Command

Purpose

Converts the access control information of a file system object from one type to another.

Syntax

aclconvert [**-R**] [**-I**] **-t** *ACLType File*

Description

The **aclconvert** command converts the access control information (ACL) of the file system object specified by the *File* parameter to another type as specified by *ACLType* argument input to command. The conversion could fail if the target ACL type is not supported by the file system where *File* exists. Also note that the ACL conversion will take place with the help of ACL type specific algorithm and invariably the conversion will be approximate. So the conversion could result in potential loss of access control and it is essential that the user of this command be sure that the converted ACL satisfies the necessary access restrictions. The user might manually review the access control information after the conversion for the file system object to ensure that the conversion was successful and fulfills the requirements of the desired access control.

Flags

Item	Description
-I	Does not display any warning messages.
-R	Recursive option allows the user to convert ACL types for all the file system objects under a directory structure to the desired ACL type.
-t *ACLType*	Specifies the target ACL type to which the File's ACL type will be converted. The conversion will succeed only if the file system in question supports the ACL type requested. If the conversion is lossy, a warning message will be issued. This kind of warning messages can be suppressed using **-I** option. The supported ACL types are ACLX and NFS4.

Exit Status

This command returns the following exit values:

Item	Description
0	The command executed successfully and all requested changes were made.
>0	An error occurred.

Security

Access Control

This command should be a standard user program and have the trusted computing base attribute.

Auditing Events

If the auditing subsystem is properly configured and is enabled, the **aclconvert** command generates the following audit record or event every time the command is run:

Event	Information
FILE_Acl	Lists access controls.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To convert the access control information for the `status` file to AIXC ACL type, type:

   ```
   aclconvert -t AIXC status
   ```

   Conversion takes place and any warning or error message is displayed.
2. To convert the access control information for the all file system objects under directory `dir1` file to AIXC ACL type and ignore any warning messages, type:

   ```
   aclconvert -RI -t AIXC dir1
   ```

   This converts all file system objects under `dir1` to the ACL type AIXC..

Location

/usr/bin/aclconvert

Files

Item	Description
/usr/bin/aclconvert	Contains the **aclconvert** command.

acledit Command

Purpose

Edits the access control information of a file.

Syntax

acledit [**-t** *ACL_type*] [**-v**] *FileObject*

Description

The **acledit** command lets you change the access control information of the file specified by the *FileObject* parameter. The command displays the current access control information and lets the file owner change it with the editor specified by the **EDITOR** environment variable. Before making any changes permanent, the command asks if you want to proceed.

Note: The **EDITOR** environment variable must be specified with a complete path name; otherwise, the **acledit** command will fail. The maximum size of the ACL data is dependent on the ACL type.

The access control information displayed depends on the ACL type associated with the file system object. Information typically includes access control entries displayed for owner and others. Also, file mode bits associated with the object could be displayed.

The following is an example of the access control information of a file:

```
attributes: SUID
base permissions:
    owner  (frank): rw-
    group (system): r-x
    others        : ---
extended permissions:
    enabled
        permit    rw-    u:dhs
        deny      r--    u:chas,    g:system
        specify   r--    u:john,    g:gateway, g:mail
        permit    rw-    g:account, g:finance
```

Note: If the **acledit** command is operating in a trusted path, the editor must have the **trusted process** attribute set.

Flags

Item	Description
-t	This optional input specifies the ACL type in which the ACL data will be stored at the end of the ACL editing process. If no option is specified, then the ACL currently associated with the file system object will be edited in its ACL type format. If an ACL type is specified with this flag, then it is assumed that user is trying to modify the current ACL type and store the ACL in a new ACL type format. When this flag is specified and the ACL type does not match the type that exists currently, it is expected that user will modify the contents of the ACL data to format into the new ACL type specific format before saving. The supported ACL types are ACLX and NFS4.

Item	Description
-v	Displays the ACL information in Verbose mode. Comment lines will be added to explain more details about the ACL associated with the FS object. These comment lines are generated when the command is executed and do not reside anywhere persistently. Hence, any modifications to the same will be lost when **acledit** is exited.

Security

Access Control

This command should be a standard user command and have the **trusted computing base** attribute.

Auditing Events

If the auditing subsystem is properly configured and is enabled, the **acledit** command generates the following audit record or event every time the command is run:

Event	Information
FILE_Acl	Lists access controls.

Files Accessed

Mode	File
x	**/usr/bin/aclget**
x	**/usr/bin/aclput**

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To edit the access control information of the `plans` file, enter:

```
acledit plans
```

Files

Item	Description
/usr/bin/acledit	Contains the **acledit** command.

aclget Command

Purpose

Displays the access control information of a file.

Syntax

aclget [**-o** *OutAclFile*] [**-t** *acl_type*] [**-v**] *FileObject*

Description

The **aclget** command writes the access control information of the file specified by the *FileObject* parameter to standard output or to the file specified by the *OutAclFile* parameter.

The information that you view depends on the ACL type and typically includes the Access Control Entries (ACEs) depicting the access rights of the users in the system, including the owner of the file object.

Flags

Item	Description
-o *OutAclFile*	Specifies that the access control information be written to the file specified by the *OutFile* parameter.
-t *acl type*	Specifies the ACL type of the ACL information being displayed. If this option is not provided the actual ACL data in its original ACL type will be displayed. The supported ACL types are ACLX and NFS4.
-v	Displays the ACL information in Verbose mode. Comment lines will be added to explain more details about the ACL associated with the FS object. These comment lines are generated when the command is executed and do not reside anywhere persistently.

Security

Access Control

This command should be a standard user program and have the **trusted computing base** attribute.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Access Control Lists

Access Control Lists form the core of protection of file system objects. Each file system object is uniquely associated with one piece of data, called ACL, that defines the access rights to the object. ACL could consist of multiple Access Control Entries (ACEs), each defining one particular set of access rights for a user. Typically ACE consists of information such as identification (to whom this ACE applies) and access rights (allow-read, deny-write). Note that ACE might also capture information such as inheritance flags and alarm and audit flags. The format and enforcement of ACL data is entirely dependent on the ACL type in which they are defined. AIX provides for the existence of multiple ACL types on the operating systems. The list of ACLs supported by a file system instance is dependent on the physical file system implementation for that file system instance.

Examples

1. To display the access control information for the `status` file, enter:

   ```
   aclget status
   ```

   An access control list appears, similar to the example in Access Control Lists.
2. To copy the access control information of the `plans` file to the `status` file, enter:

   ```
   aclget plans | aclput status
   ```

   This copies the access control information. In most cases, the ACL type associated with `plans` will be the ACL type of ACL associated with the target status. However, it is possible that the target file system

does not support the ACL type associated with file system object plans. In this case, the operation will fail and an error message is displayed. The target will retain its original associated ACL.

3. To save the access control information of the `plans` file in the `acl1` file to edit and use later, enter:

   ```
   aclget -o acl1 plans
   ```

Files

Item	Description
/usr/bin/aclget	Contains the **aclget** command.

aclgettypes Command

Purpose

Gets ACL types supported by a file system path.

Syntax

aclgettypes *FileSystemPath*

Description

The **aclgettypes** command retrieves the list of ACL types supported for a given file system path and displays the same. The default ACL type for the file system instance concerned will be displayed as the first entry.

The supported ACL types are AIXC and NFS4.

Exit Status

This command returns the following exit values:

Item	Description
0	The command executed successfully and all requested changes were made.
>0	An error occurred.

Security

Access Control

This command should be a standard user program and have the **trusted computing base** attribute.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display ACL types supported by a file system instance that contains path **/home/plan1**, type:

   ```
   aclgettypes /home/plan1
   ```

Location

/usr/bin/aclgettypes

Files

Item	Description
/usr/bin/aclgettypes	Contains the **aclgettypes** command.

aclput Command

Purpose

Sets the access control information of a file.

Syntax

aclput [**-i** *inAclFile*] [**-R**] [**-t** *acl_type*] [**-v**]*FileObject*

Description

The **aclput** command sets the access control information of the file object specified by the *FileObject* parameter. The command reads standard input for the access control information, unless you specify the **-i** flag.

Note: If you are reading from standard input your entries must match the expected format of the access control information or you will get an error message. Use the Ctrl-D key sequence to complete the session.

Access Control List

Access Control Lists form the core of protection for file system objects. Each file system object is uniquely associated with one piece of data, called ACL, that defines the access rights to the object. ACL could consist of multiple Access Control Entries (ACEs), each defining one particular set of access rights for an user. Typically, ACE consists of information such as identification (to whom this ACE applies) and access rights (allow-read, deny-write). ACE might also capture information such as inheritance flags and alarm and audit flags. The format and enforcement of ACL data is entirely dependent on the ACL type in which they are defined. AIX provides for existence of multiple ACL types on the operating system. The list of ACLs supported by a file system instance is dependent on the physical file system implementation for that file system instance.

Flags

Item	Description
-i *inAclFile*	Specifies the input file for access control information. If the access control information in the file specified by the *InFile* parameter is not correct, when you try to apply it to a file, an error message preceded by an asterisk is added to the input file. **Note:** The size of the ACL information depends on the ACL type.
-R	Applys ACL to this directory and its children file system objects recursively.
-t *ACL_type*	Specifies the ACL type of the ACL information being displayed. If this option is not provided the actual ACL data in its original ACL type will be displayed. The supported ACL types are ACLX and NFS4.

Item	Description
-v	Verbose option. This option displays many comment lines as part of the ACL data display. This could help in understanding the details of complex ACL types.

Security

Access Control

This command should be a standard user program and have the **trusted computing base** attribute.

Auditing Events

If the auditing subsystem is properly configured and is enabled, the **aclput** command generates the following audit record or event every time the command is run:

Event	Information
FILE_WriteX acl	Modification to access controls.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To set the access control information for the status file with information from standard input, enter:

   ```
   aclput status
   attributes: SUID
   ```

   and then press the Ctrl-D sequence to exit the session.
2. To set the access control information for the `status` file with information stored in the `acldefs` file, enter:

   ```
   aclput -i acldefs status
   ```
3. To set the access control information for the `status` file with the same information used for the `plans` file, enter:

   ```
   aclget plans | aclput status
   ```
4. To set the access control information for the `status` file with an edited version of the access control information for the `plans` file, you must enter two commands. First, enter:

   ```
   aclget    -o acl plans
   ```

   This stores the access control information for the `plans` file in the `acl` file. Edit the information in the `acl` file, using your favorite editor. Then, enter:

   ```
   aclput -iacl status
   ```

   This second command takes the access control information in the `acl` file and puts it on the `status` file.

Files

Item	Description
/usr/bin/aclput	Contains the **aclput** command.

adb Command

Purpose

Provides a general purpose debug program.

Syntax

adb [**-k**] [**-l** *Directory*] [**-w**] [*ObjectFile* [*CoreFile*]]

Description

The **adb** command provides a debug program for programs. With this debug program, you can examine object and core files and provide a controlled environment for running a program.

Normally, the *ObjectFile* parameter is an executable program file that contains a symbol table. If the *ObjectFile* parameter does not contain a symbol table, the symbolic features of the **adb** command cannot be used, although the file can still be examined. The default for the *ObjectFile* parameter is **a.out**.

The *CoreFile* parameter is a core image file produced by running the *ObjectFile* parameter. The default for the *CoreFile* parameter is **core**.

While the **adb** command is running, it takes standard input and writes to standard output. The **adb** command does not recognize the Quit or Interrupt keys. If these keys are used, the **adb** command waits for a new command.

In general, requests to the **adb** command are in the following form:

[*Address*] [,*Count*] [*Command*] [;]

where *Address* and *Count* are expressions. The default for the *Count* expression is a value of 1. If the *Address* expression is specified, the **.** (period) variable is set to *Address*.

The interpretation of an address depends on the context in which it is used. If a subprocess is being debugged, addresses are interpreted in the usual way in the address space of the subprocess.

Enter more than one command at a time by separating the commands with a **;** (semicolon).

The **adb** debug program allows the use of various:

- expressions
- operators
- subcommands
- variables
- addresses

Note: If the object file does not contain the symbol table, the **adb** command will not be able to show the value of static, automatic, and external variables of a program.

Flags

Item	Description
-k	Causes kernel mapping.
-l *Directory*	Specifies a directory where files to be read with $< or $<< are sought. The default is the **/usr/ccs/bin/adb** file.
-w	Opens the *ObjectFile* and the *CoreFile* parameters for reading and writing. If either file does not exist, this flag creates the file.

Return Values

The **adb** debug program is printed when there is no current command or format. The **adb** command indicates such things as inaccessible files, syntax errors, and abnormal termination of commands. Exit status is a value of 0, unless the last command was unsuccessful or returned non-zero status.

Files

Item	Description
/dev/mem	Provides privileged virtual memory read and write access.
a.out	Provides common assembler and link editor output.
core	Contains an image of a process at the time of an error.

addbib Command

Purpose

Creates or extends a bibliographic database.

Syntax

addbib [**-a**] [**-p** *PromptFile*] *Database*

Description

The **addbib** command uses a series of prompts to guide the user through creating or extending a bibliographic database. The user can define responses to these prompts. All default prompts and instructions are contained in the **refer** message catalog.

The first prompt is `Instructions?`. If the answer is affirmative, you can receive directions.

If the answer is negative or if you press the Enter key, you cannot receive directions. The **addbib** command then prompts for various bibliographic fields, reads responses from the terminal, and sends output records to the database specified by the *Database* parameter.

Pressing the Enter key (a null response) means to omit a particular field. Typing a `-` (minus sign) means to return to the previous field. A trailing backslash allows a field to be continued on the next line. The repeating `Continue?` prompt allows you to resume, to quit the current session, or to edit the database. To resume, type the defined affirmative answer or press the Enter key. To quit the current session, type the defined negative answer.

To edit the database, enter any system text editor (vi, ex, edit, ed).

Flags

Item	Description
-a	Suppresses prompting for an abstract. Prompting for an abstract is the default. Abstracts are ended by pressing a Ctrl-D key sequence.

Item	Description
-p*PromptFile*	Causes the **addbib** command to use a new prompting skeleton, which is defined in the file specified by the *PromptFile* parameter. This file contains prompt strings, a tab, and the key letters written to the specified database. The following are the most common key letters and their meanings. The **addbib** command insulates you from these key letters, since it gives you prompts in English. If you edit the bibliography file later, you need to know this information. **Note:** Except for the **%A** key letter, each field should be given just once. Only relevant fields should be supplied. **%A** Author's name **%B** Book containing article referenced **%C** City (place of publication) **%D** Date of publication **%E** Editor of book containing article referenced **%F** Footnote number or label (supplied by the **refer** command) **%G** Government order number **%H** Header commentary, printed before reference **%I** Issuer (publisher) **%J** Journal containing article **%K** Keywords to use in locating reference **%L** Label field used by **-k** flag of the **refer** command **%M** Bell Labs memorandum (undefined) **%N** Number within volume **%O** Other commentary, printed at end of reference **%P** Page numbers **%Q** Corporate or foreign author (unreversed) **%R** Report, paper, or thesis (unpublished) **%S** Series title **%T** Title of article or book **%V** Volume number **%X** Abstract used by the **roffbib** command, not by the **refer** command **%Y,Z** Ignored by the **refer** command.

Examples

The following is an example of a bibliography file:

```
%A Bill Tuthill
%T Refer - A Bibliography System
%I Computing Services
%C Berkeley
%D 1982
%O UNIX 4.3.5.
```

addrpnode Command

Purpose

Adds one or more nodes to a peer domain definition.

Syntax

addrpnode [**-c**] [**-h**] [**-TV**] *node_name1* [*node_name2* ...]

addrpnode [**-c**] { **-f** | **-F** { *file_name* | **"-"** } } [**-h**] [**-TV**] [**-M**]

addrpnode [**-c**] [**-h**] [**-TV**] *node_name1* [@*host_name1*] [*node_name2* [@*host_name2*] ...]

Description

Before running the addrpnode command:

To set up the proper security environment, run the **preprpnode** command on each node that is to be added to the peer domain.

The **addrpnode** command adds the specified nodes to the online peer domain in which the **addrpnode** command is run. This command must be run on a node that is online to the peer domain in which the new nodes are to be added. Though a node can be defined in multiple peer domains, it can be online only in one peer domain. To add one or more nodes to the peer domain, more than half of the nodes must be online.

To enable the **addrpnode** command to continue when there is an error on one of the nodes, use the **-c** flag.

The **addrpnode** command does not bring the added nodes online in the peer domain. To do so, use the **startrpnode** command.

Flags

-c
Continues processing the command while at least one node can be added to the peer domain.

By default, if the **addrpnode** command fails on any node, it will fail on all nodes. The **-c** flag overrides this behavior, so that the **addrpnode** command runs on the other nodes, even if it fails on one node.

-f | -F { *file_name* | "-" }
Specifies that node names are read from a file or from standard input.

Use **-f** *file_name* or **-F** *file_name* to read the node names from a file. Use **-f "-"** or **-F "-"** to specify STDIN as the input file.

Notes:

- Specify one node name per line. The command ignores any blank characters to the left of the node name.
- Use a number sign (#) to indicate that the remainder of the line (or the entire line if the # is in column 1) is a comment.

By default, all of the nodes that are listed in *file_name*:

- are Group Services group leader candidates.
- are used for quorum decisions.
- have access to the peer domain tiebreaker mechanism.

You can customize node characteristics by using an at sign (@) control character followed by one or more of these special characters:

P | p
Specifies that the node is a Group Services group leader candidate.

Q | q
Specifies that the node is a quorum node.

B | b
Specifies that the node has access to the peer domain tiebreaker mechanism. **B** or **b** can be specified only for quorum nodes.

!
Specifies that the node does not have a certain characteristic. For example, **!Q** indicates that the node is not a quorum node.

When customizing node characteristics, consider the following points (where **x** is **P**, **Q**, or **B**):

- Use only one **@** control character per line, followed immediately by one or more special characters, after the node name and before any comments.
- Do not specify **!QB** for a node; it results an error.
- If you use a node number, add it after the node name and before any comments. The node number can precede or follow the node characteristic specifications.
- If **x** is specified for one or more nodes and **!x** is not specified for any nodes, the nodes that do not have an **x** specified are assumed to have a value of **!x**.
- If **!x** is specified for one or more nodes and **x** is not specified for any nodes, the nodes that do not have an **!x** specified are assumed to have a value of **x**.
- If **x** and **!x** are specified for different nodes in the same node file, all of the nodes in the file must have a specification of **x** or **!x**.

-h
Writes the command usage statement to standard output.

-M
Verifies whether the security compliance mode of the new node matches the domain. If the modes do not match, the node is not added. If the **-M** option is not specified, and the node is using key type which is compatible with the domain, the node is added and its compliance mode is updated to match the domain.

-T
Writes the command trace messages to standard error. For your software service organization use only.

-V
Writes the command verbose messages to standard output.

Parameters

***node_name1* [*node_name2 ...*]**
Specifies the node (or nodes) to be added to the peer domain definition. The node name is the IP address or the long or short version of the DNS host name. The node name must resolve to an IP address.

***node_name1*[@*host_name1*] [*node_name2*[@*host_name2*] ...]**

Specifies the nodes that need to be added to RPD by using the node name along with the host name for each node. The *node_name1* parameter corresponds to a label but the *host_name1* parameter is either the IP address or a long or short version of the DNS host name. The host name must be a valid value that can be contacted or pinged.

If the *HostName* parameter is not specified and only *Name* parameter is specified for the **addrpnode** command, the *HostName* parameter is set as the *Name* parameter. In this case, the *Name* parameter must resolve to IP address or long or short version of the DNS host name.

To add a node to the existing peer domain, use the following command:

```
addrpnode node_name3@host_name3
```

You can also run the **addrpnode -f /home/nodelst** command, where */home/nodelst* has node names as *node_name3@host_name3.in.ibm.com*.

Security

The user of the **addrpnode** command needs write permission for the IBM.PeerDomain resource class and the IBM.PeerNode resource class on each node that is to be added to the peer domain. It is set up by running the **preprpnode** command on each node to be added. Specify the names of all the nodes online in the peer domain with the **preprpnode** command. It gives the online nodes the necessary authority to perform operations on the nodes to be added.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is online in the peer domain in which the new nodes are to be added.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the **-f "-"** or **-F "-"** flag is specified, this command reads one or more node names from standard input.

Standard Output

When the **-h** flag is specified, the command usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To add the nodes node_name2 and node_name3 to the peer domain ApplDomain, where node_name1 is already defined and online on the peer domainApplDomain, run command on node_name1:

```
addrpnode node_name2 node_name3
```

2. To add the nodes node_name2 and node_name3 along with the host names to the peer domain ApplDomain, where node_name1 is already defined and online on the peer domain ApplDomain, run command on node_name1:

```
addrpnode node_name2@host_name2 nodeC_name3@host_name3
```

Location

/opt/rsct/bin/addrpnode

addX11input Command

Purpose

Adds an X11 input extension record into the ODM (Object Data Manager) database.

Syntax

addX11input

Description

The **addX11input** command is used to add an X11 input extension record into the ODM database. When you enter **addX11input** on the command line, the **addX11input** command requests *DeviceName, GenericName,* and *ModuleName* values in turn. The entire record is then added to the ODM database.

The command is a root/system user command. Its action fails with a permissions error if an unauthorized user attempts to add a record.

Error Codes

Item	Description
ODM could not open class	Returned if the X11 Input extension records in the ODM database are not found in the **/usr/lib/objrepos** directory.

admin Command (SCCS)

Purpose

Creates and controls Source Code Control System (SCCS) files.

Syntax

To Create New SCCS Files

admin { **-n -i**[*FileName*] } [**-a** { *User* | *GroupID* }] ... [**-f** *HeaderFlag*[*Value*] ...] [**-r** *SID*] [**-t** *FileName*] [**-m** *ModificationRequestList*] [**-y**[*Comment*]] *File* ...

Note: Do not put a space between a flag and an optional (bracketed) variable.

To Modify Existing SCCS Files

admin [**-a** { *User* | *GroupID* }] ... [**-e** { *User* | *GroupID* }] ... [{ **-d** *HeaderFlag* | **-f** *HeaderFlag*[*Value*] ... }] [**-m** *ModificationRequestList*] [**-t**[*FileName*]] [**-y**[*Comment*]] *File* ...

Note: Do not put a space between a flag and an optional (bracketed) variable.

To Check Damaged SCCS Files

admin -h *File* ...

To Correct Damaged SCCS Files

admin -z *File* ...

Description

The **admin** command creates new Source Code Control System (SCCS) files or changes specified parameters in existing SCCS files.

The **admin** command can change the parameters controlling how the **get** command builds the files that you can edit. The parameters can also set conditions about who can access the file and which releases of the files may be edited.

If the file specified by the *File* parameter exists, the **admin** command modifies the file as specified by the flags. If the file does not exist and you supply the **-i** or **-n** flag, the **admin** command creates a new file and provides default values for unspecified flags.

If you specify a directory name for the *File* parameter, the **admin** command performs the requested actions on all SCCS files in that directory. All SCCS files contain the **s.** prefix before the file name. If you use a **-** (minus sign) for the *File* parameter, the **admin** command reads standard input and interprets each line as the name of an SCCS file. An end-of-file character ends input.

You must have write permission in the directory to create a file. All SCCS file names must have the form **s.***Name.* New SCCS files are created with read-only permission. The **admin** command writes to a temporary x-file, which it calls **x.***Name*. If it already exists, the x-file has the same permissions as the original SCCS file. The x-file is read-only if the **admin** command must create a new file. After successful completion of the **admin** command, the x-file is moved to the name of the SCCS file. This ensures that changes are made to the SCCS file only if the **admin** command does not detect any errors while running.

Directories containing SCCS files should be created with permission code 755 (read, write, and execute permissions for owner, read and execute permissions for group members and others). The SCCS files themselves should be created as read-only files (444). With these permissions, only the owner can use non-SCCS commands to modify SCCS files. If a group can access and modify the SCCS files, the directories should include group write permission.

The **admin** command also uses a temporary lock file (called **z.***Name*), to prevent simultaneous updates to the SCCS file by different users.

You can enter flags and input file names in any order. All flags apply to all the files. Do not put a space between a flag and an optional variable (variable enclosed in bracket). Header flags can be set with the **-f** flag and unset with the **-d** flag. Header flags control the format of the g-file created with the **get** command.

Flags

Item	Description
-a *User* or **-a** *GroupID*	Adds the specified user to the list of users that can make sets of changes (deltas) to the SCCS file. The *User* value can be either a user name or a group ID. Specifying a group ID is the same as specifying the names of all users in that group. You can specify more than one **-a** flag on a single **admin** command line. If an SCCS file contains an empty user list, anyone can add deltas. If a file has a user list, the creator of the file must be included in the list in order for the creator to make deltas to the file. If the *User* or *GroupID* parameter is preceded by an ! (exclamation point), specified users are denied permission to make deltas. For example, enter `-a !User`.
-d *HeaderFlag*	Deactivates the effects of the specified header flag within the SCCS file. You can specify this flag only with existing SCCS files. You can also specify more than one **-d** flag in a single **admin** command. Refer to the list of header flags that follows to learn more about the supported values.
-e *User* or **-e** *GroupID*	Removes the specified user from the list of users allowed to make deltas to the SCCS file. Specifying a group ID is equivalent to specifying all *User* names common to that group. You can specify several **-e** flags on a single **admin** command line.
-f *HeaderFlag*[*Value*]	Activates the specified header flag and value in the SCCS file. You can specify more than one header flag in a single **admin** command. There are 12 header flags. Refer to the list of header flags that follows to learn more about the supported values. Do not put a space between the *HeaderFlag* and *Value* variables.
-h	Checks the structure of the SCCS file and compares a newly computed checksum with the checksum that is stored in the first line of the SCCS file. When the checksum value is not correct, the file has been improperly modified or damaged. This flag helps you detect damage caused by the improper use of non-SCCS commands to modify SCCS files, as well as accidental damage. The **-h** flag prevents writing to the file, so it cancels the effect of any other flags supplied. If an error message is returned indicating the file is damaged, use the **-z** flag to re-compute the checksum. Then test to see if the file is corrected by using the **-h** flag again.

Item	Description
-i[*FileName*]	Gets the text for a new SCCS file from the *FileName* variable. This text is the first delta of the file. If you specify the **-i** flag but omit the file name, the **admin** command reads the text from standard input until it reaches an end-of-file character. If you do not specify the **-i** flag, but you do specify the **-n** flag, the command creates an empty SCCS file. The **admin** command can only create one file containing text at a time. If you are creating two or more SCCS files with one call to the **admin** command, you must use the **-n** flag, and the SCCS files created will be empty. Each line of the file specified by the *FileName* variable cannot contain more than 512 characters. The file name can include MBCS (multibyte character set) characters. Do not put a space between the flag and the *FileName* variable.
-m *ModificationRequestList*	Specifies a list of Modification Request (MR) numbers to be inserted into the SCCS file as the reason for creating the initial delta. A null or empty list can be considered valid, depending on the validation program used. The **v** header flag must be set. The MR numbers are validated if the **v** header flag has a value (the name of an MR number validation program). The **admin** command reports an error if the **v** header flag is not set or if MR validation fails.
-n	Creates a new, empty SCCS file. When the **-n** flag is used without the **-i** flag, the SCCS file is created with control information but without any file data.
-r *SID*	Specifies the SCCS identification string (SID) file version to be created. The *SID* variable accepts a delta with four levels: release, level, branch, and sequence, for example 3.2.5.1. If only release is specified, the **admin** command automatically assumes level 1. If you do not specify the **-r** flag, the initial delta becomes release 1, level 1 (that is, 1.1). For more details on specifying the SID, refer to the SID Determination table described in the **get** command. You can specify the **-r** flag only if you also specify the **-i** or **-n** flag. Use this flag only when creating an SCCS file.
-t [*FileName*]	Takes descriptive text for the SCCS file from the file specified by the *FileName* variable. If you use the **-t** flag when creating a new SCCS file, you must supply a file name. In the case of existing SCCS files: • Without a file name, the **-t** flag removes any descriptive text currently in the SCCS file. • With a file name, the **-t** flag replaces any descriptive text currently in the SCCS file with text in the named file. • The file name can include MBCS (multibyte character set) characters. Do not put a space between the flag and the *FileName* variable.

Item	Description
-y [*Comment*]	Inserts the specified comment into the initial delta in a manner identical to that of the **delta** command. Use this flag only when you create an SCCS file. If you do not specify a comment, the **admin** command inserts a line of the following form: `date and time created YY/MM/DD HH:MM:SS by Login` The comments can include MBCS (multibyte character set) characters. Do not put a space between the flag and the *FileName* variable.
-z	Re-computes the SCCS file checksum and stores it in the first line of the SCCS file (see the **-h** flag). **Attention:** Using the **admin** command with the **-z** flag on a damaged file can prevent future detection of the damage. This flag should only be used if the SCCS file is changed using non-SCCS commands because of a serious error.
File	Specifies the name of the file created or altered by the **admin** command. If a **-** (minus sign) is specified, the **admin** command reads from standard input. An end-of-file character ends standard input.

Header Flags

The following list contains the header flags that can be set with the **-f** flag and unset with the **-d** flag. Header flags control the format of the g-file created with the **get** command.

Item	Description
b	Lets you use the **-b** flag of a **get** command to create branch deltas.
c *Number*	Makes the *Number* variable the highest release number that a **get -e** command can use. The value of the *Number* variable must be greater than 0 and less than or equal to 9999. (The default value is 9999.)
d *SID*	Makes the *SID* variable the default delta supplied to a **get** command.
f *Number*	Makes the *Number* variable the lowest release number that a **get -e** command can retrieve. The *Number* variable must be greater than 0 and less than 9999. (The default value is 1.)
i [*String*]	Treats the following informational message, issued by the **get** or **delta** command, as an error: `There are no SCCS identification keywords in the file. (cm7)` In the absence of this flag, the message is only a warning. The message is issued if no SCCS identification keywords are found in the text retrieved or stored in the SCCS file (refer to the **get** command). If a string is supplied, the keywords must match exactly the given string. The string must contain a keyword and have no embedded newlines.
j	Permits concurrent **get** commands for editing the same SID of an SCCS file. Use of the **j** header flag allows multiple concurrent updates to the same version of the SCCS file.

Item	Description
l*List*	(lowercase L) Locks the releases specified by the *List* variable against editing, so that a **get -e** command against one of these releases fails. The list has the following syntax: `<List> : : = <Range> | <List> , <Range>` `<Range> : : = SID | a` Where character a in the list is equivalent to specifying all releases for the named SCCS file.
m *Module*	Substitutes the *Module* variable for all occurrences of the **59** keyword in an SCCS text file retrieved by a **get** command. The default *Module* variable is the name of the SCCS file without the **s.** prefix. The module name can include MBCS (multibyte character set) characters.
n	Causes the **delta** command to create a null delta in any releases that are skipped when a delta is made in a new release. For example, if you make delta 5.1 after delta 2.7, releases 3 and 4 will be null. Releases 3 and 4 will be created as null delta entries in the delta table of the **s.** file. The resulting null deltas can serve as points from which to build branch deltas. Without this flag, skipped releases do not appear in the SCCS file.
q *Text*	Substitutes the specified text for all occurrences of the keyword in an SCCS text file retrieved by a **get** command.
t *Type*	Substitutes specified type for all keywords in a g-file retrieved by a **get** command.
v [*Program*]	Makes the **delta** command prompt for Modification Request (MR) numbers as the reason for creating a delta. The *Program* variable specifies the name of an MR-number validity-checking program. If the **v** flag is set in the SCCS file, the **-m** flag must also be used, even if its value is null. The program name can include MBCS (multibyte character set) characters.

Locating Damaged SCCS Files

Although SCCS provides some error protection, you may need to recover a file that was accidentally damaged. This damage may result from a system malfunction, operator error, or changing an SCCS file without using SCCS commands.

SCCS commands use the checksum to determine whether a file was changed since it was last used. The only SCCS command that processes a damaged file is the **admin** command when used with the **-h** or **-z** flags. The **-h** flag tells the **admin** command to compare the checksum stored in the SCCS file header against the computed checksum. The **-z** flag tells the command to re-compute the checksum and store it in the file header.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

These examples use an imaginary text file called `test.c` and an editor such as **ed** to edit files.

1. First, create an ordinary SCCS file. To create an empty SCCS file named `s.test.c`, enter:

```
$ admin -n s.test.c
```

Using the **admin** command with the **-n** flag creates an empty SCCS file.

2. To convert an existing text file into an SCCS file, enter:

```
$ admin -itest.c s.test.c
There are no SCCS identification keywords in the file (cm7)
$ ls
s.test.c test.c
```

If you use the **-i** flag, the **admin** command creates delta 1.1 from the specified file. Once delta 1.1 is created, rename the original text file so it does not interfere with SCCS commands:

```
$ mv test.c back.c
```

The message `There are no SCCS identification keywords in the file (cm7)` does not indicate an error. SCCS writes this message when there are no identification keywords in the file. Identification keywords are variables that can be placed in an SCCS file. The values of these variables provide information such as date, time, SID, or file name. See the **get** command for an explanation of identification keywords. If no identification keywords exist, SCCS writes the message. However, if the **i** header flag is set in the **s.** file, this message causes an error condition. This flag is set by the user.

Give the SCCS file any name, beginning with **s.**. In the preceding example, the original file and the SCCS file have the same name, but that is not necessary.

Because you did not specify a release number, the **admin** command gave the SCCS file an SID of 1.1. SCCS does not use the number 0 to identify deltas. Therefore, a file cannot have an SID of 1.0 or 2.1.1.0, for example. All new releases start with level 1.

3. To start the `test.c` file with a release number of 3.1, use the **-r** flag with the **admin** command, as shown below, and enter:

```
$ admin -itest.c -r3 s.test.c
```

To restrict permission to change SCCS files to a specific set of user IDs, list user IDs or group ID numbers in the user list of the SCCS file by using the **-a** flag of the **admin** command. This flag may appear multiple times on the command line. These IDs then appear in the SCCS file header. Without the **-a** flag to restrict access, all user IDs can change the SCCS files.

4. To restrict edit permission to the user ID `dan`, enter:

```
$ admin -adan s.test.c
```

5. Check SCCS files on a regular basis for possible damage. The easiest way to do this is to run the **admin** command with the **-h** flag on all SCCS files or SCCS directories, as follows:

```
$ admin -h s.file1 s.file2 ...
$ admin -h directory1 directory2 ...
```

If the **admin** command finds a file where the computed checksum is not equal to the checksum listed in the SCCS file header, it displays this message:

```
ERROR [s. filename]:
1255-057 The file is damaged. (co6)
```

If a file was damaged, try to edit the file again or read a backup copy. After fixing the file, run the **admin** command with the **-z** flag and the repaired file name:

```
$ admin -z s.file1
```

This operation replaces the old checksum in the SCCS file header with a new checksum based on the current file contents. Other SCCS commands can now process the file.

Files

Item	Description
/usr/bin/admin	Contains the SCCS **admin** command.

aixmibd Daemon

Purpose

Provides the AIX Enterprise Management Information Base (MIB) extension subagent, for use with the Simple Network Management Protocol (SNMP) version 3 agent, that collects data from system for variables defined in the AIX Enterprise Specific MIB.

Syntax

aixmibd [**-f** *FileName*] [**-d** *Level*] [**-a** *Host*] [**-c** *Community*]

Description

The AIX Enterprise MIB extension subagent is a daemon, aixmibd, that collects data from system for variables defined in the AIX Enterprise Specific MIB. The subagent receives SNMP requests and sends data via the SNMP-DPI API for communication with the main AIX snmpd daemon. An Enterprise Management application or other simple application (example snmpinfo command) uses SNMP protocol to get or set AIX MIB objects.

One focus of the subagent is on the data related to the file systems, volume groups, logical volumes, physical volumes, paging space, processes, print queues, print jobs, system users, system groups, users currently logged in, subsystems, subservers, system environment, and various devices.

Another focus of the subagent is on important system traps. Traps, which are also called indications, or notifications, are event reports and are used to decrease the length of time between when the event happens and when it is noticed by a manager so that the event can be handled timely. Traps are generated periodically to report the status change and operating status of the system. From analyzing the data, a manager can determine if a device and the whole system are functioning properly and securely, and make appropriate adjustment. For example, when the **/home** file system reaches the threshold 95% (percent used size), a trap can be generated to report the event to a manager. The manager can respond by sending an email, paging, and so on. To indicate system critical events instantly, a series of traps will be generated by the subagent.

Note: The AIX enterprise subagent should be started by the System Resource Controller (SRC). Entering `aixmibd` at the command line is not recommended.

Flags

Item	Description
-a *Host*	Causes the request to be sent to the specified host. The host can be an IPv4 address, an IPv6 address, or a host name.
-c *Community*	Specifies the community name.

Item	Description
-d *Level*	Specifies the tracing/debug level. The default level is 56. The debug levels are defined as follows: • 8 = DPI level 1 • 16 = DPI level 2 • 32 = Internal level 1 • 64 = Internal level 2 • 128 = Internal level 3 Add the numbers to specify multiple trace levels.
-f *File*	Specifies a non-default configuration file.

Examples

1. In order to cause the **aixmibd** subagent to connect to the SNMP agent on the host 'host1' with the community name 'instrum', enter the following:

```
startsrc -s aixmibd -a "-a host1 -c instrum"
```

2. Because the **aixmibd** subagent is controlled by SRC, it can be activated by **startsrc**. After the **aixmibd** subagent is activated by **startsrc** in this example, the subagent will connect to the SNMP agent on the host nmsu over TCP with default community name 'public':

```
startsrc -s aixmibd -a "-a nmsu"
```

Files

Item	Description
/etc/aixmibd.conf	Contains the configuration file for the **aixmibd** subagent.
/usr/samples/snmpd/aixmibd_security_readme	**/usr/samples/snmpd/aixmibd_security_readme** contains the example configurations for different views and information about related security issues. Also contains information describing how to set the variables in **/etc/aixmibd.conf**.
/usr/samples/snmpd/aixmibd.my	Contains the MIB definitions for the **aixmibd** subagent.

aixpert Command

Purpose

Aids the system administrator in setting the security configuration.

Syntax

`aixpert`

`aixpert -l h|high | m|medium | l|low | d|default | s|sox-cobit [-n -o` *filename* `] [ -a -o` *filename* `] [` **-p** `]`

`aixpert -c [` **-P** `] <`*profile name*`> [`**-r**`] [`**-R**`]`

`aixpert -u [` **-p** `]`

`aixpert -d`

`aixpert [-f` *filename* `] [ -a -o` *filename* `] [` **-p** `]`

`aixpert -t`

`aixpert -c -P` *<profile name>*

Description

The `aixpert` command sets a variety of system configuration settings to enable the desired security level.

Running `aixpert` with the only the `-l` flag set implements the security settings promptly without letting the user configure the settings. For example, running `aixpert -l high` applies all the high-level security settings to the system automatically. However, running `aixpert -l` with the `-n -o` *filename* option saves the security settings to a file specified by the *filename* parameter. The `-f` flag then applies the new configurations.

After the initial selection, a menu is displayed itemizing all security configuration options associated with the selected security level. These options can be accepted in whole or individually toggled off or on. After any secondary changes, `aixpert` continues to apply the security settings to the computer system.

Note: It is recommended that `aixpert` be rerun after any major systems changes, such as the installation or updates of software. If a particular security configuration item is deselected when `aixpert` is rerun, that configuration item is skipped.

Some profiles of the **aixpert** command have shun port rules that create dynamic IP security (IPSec) filter rules and exist for a specified duration. These IPSec filter rules deny all packets that arrive from a specific port of the source host. When fragmented packets arrive at the destination host, the deny filter rules are applied on the fragments based on the source IP, the destination IP, and the protocol, irrespective of the source and destination ports because the IP fragments do not contain the port details. Therefore, these deny rules drop all fragments on all ports, which are received at the destination from all source ports for the specified protocol from the specified source.

If the IP fragments from a specified source must be allowed at the destination, an appropriate **genfilt** rule must be added for that source after the **aixpert** rules are applied. This new rule must be added above the **aixpert** rules so that the **genfilt** rule can take effect. Adding such a rule might make the destination vulnerable to IP fragmentation attacks from the source. Therefore, such rules must be added with diligence. For more information about handling fragments by using IPSec filters, see **genfilt** man page.

Flags

Item	Description
-a	The settings with the associated level security options are written in abbreviated file format to the file specified by the `-o` flag. You must specify the **-o** option when you specify the **-a** option.
-c	Checks the security settings against the previously applied set of rules. If the check against a rule fails, the previous versions of the rule are also checked. This process continues until the check passes, or until all of the instances of the failed rule in the **/etc/security/aixpert/core/appliedaixpert.xml** file are checked.

Item	Description
-f	Applies the security settings in the provided *filename*. For example, the following command writes all of the high-level security options to the **/etc/security/aixpert/core/hls.xml** file: `aixpert -l h -n -o /etc/security/aixpert/core/hls.xml` After removing any unwanted options, you can apply these security settings with the following command: `aixpert -f /etc/security/aixpert/core/hls.xml` When you specify the **-f** option, security settings are consistently applied from system to system by securely transferring and applying an **appliedaixpert.xml** file from system to system. All the successfully applied rules are written to the **/etc/security/aixpert/core/appliedaixpert.xml** file and the corresponding "undo" action rules are written to the **/etc/security/aixpert/core/undo.xml** file.
-l	Sets the system security settings to the level specified with this option. This flag has the following options: **h|high** Specifies high-level security options. **m|medium** Specifies medium-level security options. **l|low** Specifies low-level security options. **d|default** Specifies AIX standards-level security options. **s|sox-cobit** Specifies SOX-COBIT best practices-level security options. If you specify both the **-l** and **-n** flags, the security settings are not implemented on the system; however, they are only written to the file that you specified in the **-o** flag. All the successfully applied rules are written to the **/etc/security/aixpert/core/appliedaixpert.xml** file and the corresponding undo action rules are written to the **/etc/security/aixpert/core/undo.xml** file. **Attention:** When you use the **d|default** option, the option can overwrite the configured security settings that you previously set through the **aixpert** command or independently, and restores the system to its traditional open configuration.
-n	The settings with the associated level security options are written to the file specified by the -o flag. You must specify the **-o** option when you use the **-n** option.
-o	Stores security output to the file pointed to by *filename*. The output file has its read and write permissions set to root as a security precaution. This file should be protected against unwanted access.
-p	Specifies that the output of the security rules is displayed by using verbose output. The **-p** option logs the rules processed into the audit subsystem if the **auditing** option is turned on. This option can be used with any of the **-l**, **-u**, **-c** and **-f** options.

Item	Description
-P	Accepts the profile name as input. This option is used along with the **-c** option. The **-c** option along with the **-P** option is used to check the compatibility of the system is with the profile passed.
-r	Reports existing settings of the system. The output is intended to be used in security or compliance audit reports. The report describes each setting, how it might relate to a regulatory compliance requirement, and whether the check passed or failed.
-R	Produces the same output as the **-r** flag, but also appends a description about each script or program used to implement the configuration setting.
-t	Displays the type of the profile applied on the system.
-u	Undoes the security settings that have been applied.
-d	Displays the document type definition (DTD).

Parameters

Item	Description
filename	The output file that stores the security settings. Root permission is required to access this file.

Security

The `aixpert` command is executable only by root.

Examples

1. To write all of the high-level security options to an output file, use the following command:

   ```
   aixpert -l high -n -o /etc/security/aixpert/plugin/myPreferredSettings.xml
   ```

   After completing this command, the output file can be edited, and specific security roles can be commented out by enclosing them in the standard xml comment string (`<--` begins the comment and `-\>` closes the comment).
2. To apply the security settings from a configuration file, use the following command:

   ```
   aixpert -f /etc/security/aixpert/plugin/myPreferredSettings.xml
   ```
3. To check the security settings that have been applied to the system, and to log the rules that failed into the audit subsystem, use the following command:

   ```
   aixpert –c -p
   ```

Location

Item	Description
`/usr/sbin/aixpert/`	Contains the `aixpert` command.

Files

Item	Description
`/etc/security/aixpert/core/aixpertall.xml`	Contains an xml listing of all possible security settings. Has -r-------- permissions, and requires root security.
`/etc/security/aixpert/core/appliedaixpert.xml`	Contains an xml listing of applied security.
`/etc/security/aixpert/log/aixpert.log`	Contains a trace log of applied security settings. This does not use syslog. The `aixpert` command writes directly to the file. Has -rw------- permissions, and requires root security.
`/etc/security/aixpert/log/firstboot.log`	Contains a trace log of the security settings that were applied during the first boot of a Secure by Default (SbD) installation.
`/etc/security/aixpert/core/undo.xml`	Contains an xml listing of security settings, which can be undone.

aixpertldap Command

Purpose

Uploads or downloads AIX Security Expert XML configuration files to or from a centralized location on a Light Directory Access Protocol (LDAP) server.

Syntax

aixpertldap **-u** **-D** *binddn* **-w** *bindpwd* [**-b** *basedn*] [**-f** *filename*] [**-l** *label*]

aixpertldap **-d** **-D** *binddn* **-w** *bindpwd* [**-b** *basedn*]

aixpertldap [**-?**]

Description

The **aixpertldap** command allows a system administrator to store AIX Security Expert XML configuration files in a centralized location on an LDAP server. By sharing these configuration files, similar systems operating in similar environments can easily download these security policies (XML configuration files), and apply the policies with the **aixpert** command. In this way, systems with similar security requirements are configured the same.

When this command downloads the AIX Security Expert security policy configuration files from the LDAP server, these files are placed in the local **/etc/security/aixpert/ldap** directory. The system administrator can scan these files, choose a relevant file, and apply the security settings specified in the file using the **-f** option of the **aixpert** command.

Tip: With the existing LDAP setup, this command uses the binding distinguished name and the binding password of the running LDAP client to store or retrieve XML configuration files on or from an LDAP server.

Flags

Item	Description
-D *binddn*	Specifies the binding distinguished name to connect to an LDAP server.
-w *bindpwd*	Specifies the binding password to read or write XML configuration files from or to an LDAP server.

Item	Description
-b *basedn*	Specifies the centralized location where the XML configuration files are stored. • If you specify the *basedn* parameter while XML files are being uploaded, the XML files are stored under the location specified by the *basedn* parameter; otherwise the files are stored under the location specified by the default *basedn* value: `cn=aixdata`. For example, if the *basedn* parameter is specified as "`ou=Austin,o=ibm,c=US`", the **aixpertldap** command stores the XML configuration files under the "`ou=aixpert,ou=Austin,o=ibm,c=US`" distinguished name (DN). • If you specify the *basedn* parameter while XML files are being downloaded, the **aixpertldap** command searches under the specific DN for the XML files; otherwise the default *basedn* value (`cn=aixdata`) is used to search the XML files. For example, if the *basedn* parameter is not specified, the **aixpertldap** command searches for XML files under the default *basedn* value: `ou=aixpert, ou=aixdata`.
-d	Downloads the XML configuration files from an LDAP server to the local **/etc/security/aixpert/ldap** directory.
-f *filename*	Specifies the full path of the XML configuration file to be uploaded to an LDAP server. If you do not specify the option, the **/etc/security/aixpert/core/appliedaixpert.xml** file is uploaded to the LDAP server by default. **Restriction**: The **f** and **d** options are mutually exclusive.
-l *label*	Specifies the short description of the content in the XML configuration file that is being uploaded. If you do not this option, the XML file has the host name as the label. For example, if the XML file contains security settings of Accounts department, the label is named AccountsDept. **Restriction**: The **l** and **d** options are mutually exclusive.
-u	Uploads the XML configuration files to an LDAP server.
-?	Displays the usage statement of the command.

Exit Status

Item	Description
0	Success.
1	Failure or partial failure.

Security

Only root users can run the **aixpertldap** command.

Examples

1. To upload the **/home/hussain/netwsec.xml** file under the `ou=aixpert, ou=Bangalore,o=ibm,c=IN` DN with the `NetworkSecurity` label, use the following command:

```
aixpertldap –u –D binddn -w secret –b ou=Bangalore,o=ibm,c=IN
–f /home/hussain/netwsec.xml –l NetworkSecurity
```

2. To download all XML files from the `ou=aixpert, ou=Bangalore,o=ibm,c=IN` DN to the **/etc/security/aixpert/ldap** directory, use the following command:

```
aixpertldap –d –D binddn -w secret –b ou=Bangalore,o=ibm,c=IN
```

3. To download the XML files from the ou=aixpert, cn=aixdata DN, use the following command:

```
aixpertldap -d –D binddn -w secret
```

Files

Item	Description
/etc/security/aixpert/ldap	Stores the downloaded XML configuration files.

aixterm Command

Purpose

Initializes an Enhanced X-Windows terminal emulator.

Syntax

aixterm [**-ah**] [**-ar**] [**-autopush**] [**-b** *NumberPixels*] [**-bd** *Color*] [**-bg** *Color*] [**-bw** *NumberPixels*] [**-cc** *CharRange*:*Value* [**,**...]] [**-cr** *Color*] [**-csd** *CharShape*] [**-cu**] [**-C**] [**-display** *Name*:*Number*] [**-dw**] [**-f0** *Font*] [**-f1** *Font*] [**-f2** *Font*] [**-f3** *Font*] [**-f4** *Font*] [**-f5** *Font*] [**-f6** *Font*] [**-f7** *Font*] [**-f0** *FontSet*] [**-f1** *FontSet*] [**-f2** *FontSet*] [**-f3** *FontSet*] [**-f4** *FontSet*] [**-f5** *FontSet*] [**-f6** *FontSet*] [**-f7** *FontSet*] [**-fb** *Font*] [**-fg** *Color*] [**-fi** *FontSet*] [**-fn** *Font*] [**-fs** *Font*] [**-fullcursor**] [**-geometry** *Geometry*] [**#geometry** *Geometry*] [**-help**] [**-i**] [**-ib** *File*] [**-im** *InputMethod*] [**-j**] [**-keywords**] [**-lang** *Language*] [**-l**] [**-leftscroll**] [**-lf** *File*] [**-ls**] [**-mb**] [**-mc** *Number*] [**-ms** *Color*] [**-mn**] [**-n** *IconName*] [**-name** *Application*] [**-nb** *Number*] [**-nobidi**] [**-nonulls**] [**-nss** *NumShape*] [**-orient** *Orientation*] [**-outline** *Color*] [**-po** *Number*] [**-ps**] [**-pt** *Preedit*] [**-reduced**] [**-rfb** *Font*] [**-rfi** *Font*] [**-rfn** *Font*] [**-rfs** *Font*] [**-rf0** *Font*] [**-rf1** Font] [**-rf2** Font] [**-rf3** Font] [**-rf4** Font] [**-rf5** Font] [**-rf6**Font] [**-rf7** Font] [**-rf0** *FontSet*] [**-rf1** *FontSet*] [**-rf2** *FontSet*] [**-rf3** *FontSet*] [**-rf4** *FontSet*] [**-rf5** *FontSet*] [**-rf6** *FontSet*] [**-rf7** *FontSet*] [**-rv**] [**-rw**] [**-s**] [**-sb**] [**-sf**] [**-si**] [**-sk**] [**-sl** *NumberLines*] [**-sn**] [**-st**] [**-suppress**] [**-symmetric**] [**-T** *Title*] [**-text** *TextType*] [**-ti**] [**-tm** *String*] [**-tn** *TerminalName*] [**-ut**] [**-v**] [**-vb**] [**-W**] [**-xrm** *String*] [**-132**] [**-e** *Command*]

Description

The **aixterm** command provides a standard terminal type for programs that do not interact directly with Enhanced X-Windows. This command provides an emulation for a VT102 terminal or a high function terminal (HFT). The VT102 mode is activated by the **-v** flag.

The **aixterm** command supports the display for up to 16 colors at a time.

The **aixterm** terminal supports escape sequences that perform terminal functions such as cursor control, moving and deleting lines, and **aixterm** private functions.

Many of the special **aixterm** terminal features (like the scroll bar) can be modified under program control through a set of private **aixterm** command escape sequences. You can also use escape sequences to change the title in the title bar.

There are three different areas in the **aixterm** window:

- Scroll bar
- Status line
- Terminal window.

By default, only the terminal window is initially displayed.

The terminal window is the area provided for terminal emulation. When you create a window, a pseudo terminal is allocated and a command (usually a shell) is started.

The **aixterm** command automatically highlights the window border and the text cursor when the mouse cursor enters the window (selected) and unhighlights them when the mouse cursor leaves the window (unselected). If the window is the focus window, the window is highlighted regardless of the location of the mouse cursor. Any window manager, as in the case of the AIXwindows Window Manager (MWM), can cover the **aixterm** border, and the highlight and border color do not show.

The **WINDOWID** environment variable is set to the resource ID number of the **aixterm** window.

When running in an **aixterm** window, the **TERM** environment variable should be **TERM=aixterm**.

The **TERM** environment variable on your home machine determines what the **TERM** environment variable should be on the remote machine (unless it is overridden by your **.profile**).

When you use the **rlogin**, **tn**, or **rsh** commands to login to a different machine, the **TERM** environment variable should be set to **aixterm**. If this operation does not occur, you can perform the following two command line operations:

1. **TERM=aixterm**
2. **export TERM**

If commands (for example, the **vi** command) do not recognize the term type **aixterm** when you login to another system, perform the following one-time operation on the remote system:

1. **su**
2. **cd/tmp**
3. **mkdir** *Xxxxx*
4. **cd** *Xxxxx*
5. **ftp** *LocalSystemName*
6. **cd /usr/share/lib/terminfo**
7. **get ibm.ti**
8. **quit**
9. **TERMINFO=/tmp/***Xxxxx*
10. **export TERMINFO**
11. **tic ibm.ti**
12. **ls**
13. **ls a**
14. **mkdir /usr/share/lib/terminfo/a**
15. **cp a/aixterm* /usr/share/lib/terminfo/a**
16. **cd /tmp**
17. **rm -r /tmp/***Xxxxx*
18. **exit**
19. On the remote machine, enter the following:
    a. **TERM=aixterm**
    b. **export TERM**

Arabic/Hebrew Support

The **aixterm** command supports bidirectional languages such as Arabic and Hebrew. This command can open a window to be used with Arabic/Hebrew applications. You can create an Arabic/Hebrew window by specifying an Arabic or Hebrew locale (**ar_AA**, **Ar_AA**, **iw_IL**, or **Iw_IL**) with the **-lang** flag or by predefining an Arabic or Hebrew locale from SMIT for the system.

The Arabic/Hebrew window supports bidirectional text display. Thus, English and Arabic or Hebrew text can be displayed on the same line. There are different aspects in the Arabic/Hebrew window:

- Screen Orientation

- Text mode
- Character shaping
- Numeric representation
- Status line

Screen Orientation

The screen orientation in an Arabic/Hebrew window can be either left-to-right or right-to-left. The default orientation is left-to-right unless otherwise specified with a flag or in the **.Xdefaults** file. While the window is active, you can reverse the screen orientation using special key combinations. You can reverse the screen orientation according to your needs.

Text Mode

An Arabic/Hebrew window supports two text modes and their corresponding manipulation:

- Implicit
- Visual

In the implicit text mode, characters are stored in same order that they are entered. The text is transformed into its visual form only when it is displayed. In the visual text mode, characters are stored in the same way that they are displayed on the window.

Character Shaping

The Arabic/Hebrew window represents Arabic and Hebrew texts differently, according to its context. Text is represented in one of the following forms:

- Automatic
- Isolated
- Initial
- Middle
- Final

Arabic/Hebrew can also be shaped according to the passthru mode.

Numeric Representation

Numerics can be represented in Arabic numerals, Hindi numerals, or in passthru mode. In implicit text mode, numerals can also be represented according to their contextual form. Thus, Arabic numbers can be displayed in English text or Hindi numbers can be displayed in Arabic text.

Status Line

The Arabic/Hebrew window can display an optional status line that shows the current status of the window. The status line contains the following values:

Value	Current Setting
E	English language
N	National language
SCR->	Left-to-right screen orientation
<-SCR	Right-to-left screen orientation
alef	Auto shape mode
blank	Passthru shaping mode
ghain	Displayed in the currently used shaping mode
I	Implicit text mode
V	Visual text mode

Value	Current Setting
U	Context numbers
A	Arabic numbers
H	Hindi numbers
P	Passthru for numbers

Note: Use the implicit text mode (the default text mode) for more efficient data sorting.

Use the following key combinations in an Arabic/Hebrew window to change certain settings.

Key Combination	Purpose
Alt + Enter	Reverses screen direction.
Alt + Right Shift	Enables Arabic/Hebrew keyboard layer.
Alt + Left Shift	Enables English keyboard layer.

For Implicit Mode only:

Item	Description
Alt + Kpd*	Adjusts the column heading.

For Visual Mode only:

Item	Description
Alt + Kpd 1	Shapes characters in their initial form.
Alt + Kpd 2	Shapes characters in their isolated form.
Alt + Kpd 3	Shapes characters in their passthru form.
Alt + Kpd 4	Shapes characters automatically (Valid also for Implicit).
Alt + Kpd 7	Shapes characters in their middle form.
Alt + Kpd 8	Shapes characters in their final form.
Shift + Kpd /	Toggles the Push Mode (Push/End Push).
Alt + Kpd /	Toggles the Autopush function.

Using the aixterm Command Data-Stream Support

The following is a list of the escape sequences supported by the **aixterm** command.

Some escape sequences activate and deactivate an alternate screen buffer that is the same size as the display area of the window. This capability allows the contents of the screen to be saved and restored. When the alternate screen is activated, the current screen is saved and replaced with the alternate screen. Saving lines scrolled off of the window is disabled until the original screen is restored.

The following table uses these abbreviations in the right hand column:

Xv
Supported by the **aixterm** command running in VT100 mode.

Xh
Supported by the **aixterm** command running in HFT mode.

H
Found in the HFT data stream.

V
Found in the VT100 data stream.

Item	Description
BEL	**Function (single-byte control)** Bell **Data Stream** 0x07 **Support** Xv, Xh, H, V
BS	**Function (single-byte control)** Backspace **Data Stream** 0x08 **Support** Xv, Xh, H, V
HT	**Function (single-byte control)** Horizontal tab **Data Stream** 0x09 **Support** Xv, Xh, H, V
LF	**Function (single-byte control)** Linefeed **Data Stream** 0x0A **Support** Xv, Xh, H, V
VT	**Function (single-byte control)** Vertical tab **Data Stream** 0x0B **Support** Xv, Xh, H, V
FF	**Function (single-byte control)** Form feed **Data Stream** 0x0C **Support** Xv, Xh, H, V
CR	**Function (single-byte control)** Carriage return **Data Stream** 0x0D **Support** Xv, Xh, H, V

Item	Description
SO	**Function (single-byte control)** Shift out **Data Stream** 0x0E **Support** Xv, Xh, H, V
SI	**Function (single-byte control)** Shift in **Data Stream** 0x0F **Support** Xv, Xh, H, V
DCI	**Function (single-byte control)** Device control 1 **Data Stream** 0x11 **Support** H, V
DC3	**Function (single-byte control)** Device control 3 **Data Stream** 0x13 **Support** H, V
CAN	**Function (single-byte control)** Cancel **Data Stream** 0x18 **Support** H, V
SUB	**Function (single-byte control)** Substitute (also cancels) **Data Stream** 0x1A **Support** H, V
ESC	**Function (single-byte control)** Escape **Data Stream** 0x1B **Support** Xv, Xh, H, V

Item	Description
SS4	**Function (single-byte control)** Single Shift 4 **Data Stream** 0x1C **Support** H
SS3	**Function (single-byte control)** Single Shift 3 **Data Stream** 0x1D **Support** H
SS2	**Function (single-byte control)** Single Shift 2 **Data Stream** 0x1E **Support** H
SS1	**Function (single-byte control)** Single Shift 1 **Data Stream** 0x1F **Support** H
cbt	**Function (single-byte control)** cursor back tab **Data Stream** ESC [Pn Z **Support** Xv, Xh, H
cha	**Function (single-byte control)** cursor horizontal absolute **Data Stream** ESC [Pn G **Support** Xv, Xh, H
cht	**Function (single-byte control)** cursor horizontal tab **Data Stream** ESC [Pn I **Support** H

Item	Description
ctc	**Function (single-byte control)** cursor tab stop control **Data Stream** ESC [Pn W **Support** H
cnl	**Function (single-byte control)** cursor next line **Data Stream** ESC [Pn E **Support** H
cpl	**Function (single-byte control)** cursor preceding line **Data Stream** ESC [Pn F **Support** Xv, Xh, H
cpr	**Function (single-byte control)** cursor position report **Data Stream** ESC [Pl; Pc R **Support** Xv, Xh, H, V
cub	**Function (single-byte control)** cursor backward **Data Stream** ESC [Pn D **Support** Xv, Xh, H, V
cud	**Function (single-byte control)** cursor down **Data Stream** ESC [Pn B **Support** Xv, Xh, H, V
cuf	**Function (single-byte control)** cursor forward **Data Stream** ESC [Pn C **Support** Xv, Xh, H, V

Item	Description
cup	**Function (single-byte control)** cursor position **Data Stream** ESC [Pl; PC H **Support** Xv, Xh, H, V
cuu	**Function (single-byte control)** cursor up **Data Stream** ESC [Pn A **Support** Xv, Xh, H, V
cvt	**Function (single-byte control)** cursor vertical tab **Data Stream** ESC [Pn Y **Support** H
da1	**Function** Device attributes • request (host to vt100) • response (vt100 to host) **Data Stream** • For a request, ESC [c • For a request, ESC [0 c • For a response, ESC [? 1 ; 2 c **Support** Xv, Xh, V
dch	**Function (single-byte control)** delete character **Data Stream** ESC [Pn P **Support** Xv, Xh, H
decaln	**Function (single-byte control)** screen alignment display **Data Stream** ESC # 8 **Support** Xv, Xh, V

Item	Description
deckpam	**Function (single-byte control)** keypad application mode **Data Stream** ESC = **Support** Xv, V
deckpnm	**Function (single-byte control)** keypad numeric mode **Data Stream** ESC > **Support** Xv, V
decrc	**Function (single-byte control)** restore cursor & attributes **Data Stream** ESC 8 **Support** Xv, Xh, V
decsc	**Function (single-byte control)** save cursor & attributes **Data Stream** ESC 7 **Support** Xv, Xh, V
decstbm	**Function (single-byte control)** set top & bottom margins **Data Stream** ESC [Pt; Pb r **Support** Xv, Xh, V
dl	**Function (single-byte control)** delete line **Data Stream** ESC [Pn M **Support** Xv, Xh, H

Item	Description
dsr	**Function (single-byte control)** device status report **Data Stream** ESC [Ps n **Support** • 0 response from vt100: ready—Xv, Xh, V • 5 command from host: please report status—Xv, Xh, V • 6 command from host: report active position—Xv, Xh, H, V • 13 error report sent from virtual terminal to host—H
dmi	**Function (single-byte control)** disable manual input **Data Stream** ESC ` (back quote) **Support** H
emi	**Function (single-byte control)** enable manual input **Data Stream** ESC b **Support** H
ea	**Function (single-byte control)** erase area **Data Stream** ESC [Ps O **Support** • 0 erase to end of area—Xv, Xh, H • 1 erase from area start—Xv, Xh, H • 2 erase all of area—Xv, Xh, H
ed	**Function (single-byte control)** erase display **Data Stream** ESC [Ps J **Support** • 0 erase to end of display—Xv, Xh, H, V • 1 erase from display start—Xv, Xh, H, V • 2 erase all of display—Xv, Xh, H, V

Item	Description
ef	**Function (single-byte control)** erase field-e,s,all **Data Stream** ESC [Ps N **Support** • 0 erase to end of field—Xv, Xh, H • 1 erase from field start—Xv, Xh, H • 2 erase all of field—Xv, Xh, H
el	**Function (single-byte control)** erase line **Data Stream** ESC [Ps K **Support** • 0 erase to end of line—Xv, Xh, H, V • 1 erase from line start—Xv, Xh, H, V • 2 erase all of line—Xv, Xh, H, V
ech	**Function (single-byte control)** erase character **Data Stream** ESC [Pn X **Support** Xv, Xh, H
hts	**Function (single-byte control)** horizontal tab stop **Data Stream** ESC H **Support** Xv, Xh, H, V
hvp	**Function (single-byte control)** horizontal and vertical position **Data Stream** ESC [Pl; Pc f **Support** Xv, Xh, H, V
ich	**Function (single-byte control)** insert character **Data Stream** ESC [Pn @ **Support** Xv, Xh, H

Item	Description
il	**Function (single-byte control)** insert line **Data Stream** ESC [Pn L **Support** Xv, Xh, H
ind	**Function (single-byte control)** index **Data Stream** ESC D **Support** Xv, Xh, H, V
ls2	**Function (single-byte control)** lock shift G2 **Data Stream** ESC n **Support** Xv
ls3	**Function (single-byte control)** lock shift G2 **Data Stream** ESC o **Support** Xv
nel	**Function (single-byte control)** next line **Data Stream** ESC E **Support** Xv, Xh, H, V
ksi	**Function (single-byte control)** keyboard status information **Data Stream** ESC [Ps p **Support** H
pfk	**Function (single-byte control)** PF key report **Data Stream** ESC [Pn q **Support** Xh, H

Item	Description
rcp	**Function (single-byte control)** restore cursor position **Data Stream** ESC [u **Support** Xv, Xh, H
ri	**Function (single-byte control)** reverse index **Data Stream** ESC M **Support** Xv, Xh, H, V
ris	**Function (single-byte control)** reset to initial state **Data Stream** ESC c **Support** Xv, Xh, H, V
rm	**Function (single-byte control)** reset mode, restore mode, save mode **Data Stream** • reset mode, ANSI specified modes (see sm)—ESC [Ps;...;Ps • reset mode, other private modes and XTERM private modes (see sm)—ESC [? Ps;...;Ps l • restore mode, other private modes and XTERM private modes (see sm)—ESC [? P;...;Ps r • save mode, other private modes and XTERM private modes (see sm)—ESC [? Ps;...;Ps s

Item	Description
sapv	**Function** select alternate presentation variant • 0 set default values for BIDI • 1 set Arabic numeric shapes • 2 set Hindi numeric shapes • 3 set symmetric swapping mode for directional characters • 5 the following graphic character is presented in its isolated form (Arabic only) • 6 the following graphic character is presented in its initial form (Arabic only) • 7 the following graphic character is presented in its middle form (Arabic only) • 8 the following graphic character is presented in its final form (Arabic only) • 13 set Special shaping mode • 14 set standard shaping mode • 15 reset symmetric mode • 18 Passthru (everything) • 19 Passthru (everything except numbers) • 20 Contextual numbers (device dependent) • 21 lock 5, 6, 7, 8 • 22 unlock • 23 set the nonull mode • 24 reset the nonull mode • Values 5-8 affect only the following character unless used with values 21 or 22 **Data Stream** ESC [Psl;...Psn] **Support** Xh
scp	**Function (single-byte control)** save cursor position **Data Stream** ESC [s **Support** Xv, Xh, H

Item	Description
scs	**Function (single-byte control)** select character set • United Kingdom Set • ASCII Set (USASCII) • special graphics **Data Stream** United Kingdom Set: • ESC (A (GO) • ESC) A (G1) • ESC * A (G2) • ESC + A (G3) ASCII Set (USASCII): • ESC (B (GO) • ESC) B (G1) • ESC * B (G2) • ESC + B (G3) special graphics: • ESC (0 (GO) • ESC) 0 (G1) • ESC * 0 (G2) • ESC + 0 (G3) **Support** Xv, V
sd	**Function (single-byte control)** scroll down **Data Stream** ESC [Pn T **Support** H
sl	**Function (single-byte control)** scroll left **Data Stream** ESC [Pn Sp @ **Support** H

Item	Description
spd	**Function (single-byte control)** select screen direction • 0 turn screen to left-to-right, set to Latin keyboard • 1 turn screen direction to right-to-left set to National keyboard **Data Stream** ESC [Ps1;1 S **Support** Xh
sr	**Function (single-byte control)** scroll right **Data Stream** ESC [Pn Sp A **Support** H
srs	**Function (single-byte control)** select reversed string • 0 end push • 1 start push **Data Stream** ESC [Ps[**Support** Xh
ss2	**Function (single-byte control)** single shift G2 **Data Stream** ESC N **Support** Xv
ss3	**Function (single-byte control)** single shift G3 **Data Stream** ESC O **Support** Xv
su	**Function (single-byte control)** scroll up **Data Stream** ESC [Pn S **Support** Xv, Xh, H

Item	Description
sgr	**Function (single-byte control)** set graphic rendition **Data Stream** ESC [Ps m **Support** • 0 normal—Xv, Xh, H, V • 1 bold—Xv, Xh, H, V • 4 underscore—Xv, Xh, H, V • 5 blink (appears as bold)—Xv, Xh, H, V • 7 reverse—Xv, Xh, H, V • 8 invisible—Xh, H • 10..17 fonts—Xh, H • 30..37 foreground colors—Xh, H • 40..47 background colors—Xh, H • 90..97 foreground colors—Xh, H • 100..107 background colors—Xh, H
sg0a	**Function (single-byte control)** set GO character set **Data Stream** ESC (< **Support** Xh, H
sg1a	**Function (single-byte control)** set G1 character set **Data Stream** ESC) < **Support** Xh, H

Item	Description
sm	**Function (single-byte control)** set mode • ANSI specified modes • Other private modes **Data Stream** • ANSI specified modes—ESC [Ps;...;Ps h • Other private modes—ESC [? Ps;...;Ps h **Support** • (ANSI) 4 IRM insert mode—Xv, Xh, H • (ANSI) 12 SRM send/rec mode—H • (ANSI) 18 TSM tab stop mode—H • (ANSI) 20 LNM linefeed/newline—Xv, Xh, H, V • 1 normal/application cursor—Xv, V • 3 80/132 columns—Xv, Xh, V • 4 smooth/jump scroll—Xv, Xh, V • 5 reverse/normal video—Xv, Xh, V • 6 origin/normal—Xv, Xh, V • 7 on/off autowrap—Xv, Xh, H, V • 8 on/off autorept—Xv, Xh, V • 21 CNM CR-NL—H • (XTERM) 40 132/80 column mode—Xv, Xh • (XTERM) 41 curses(5) fix—Xv, Xh • (XTERM) 42 hide/show scroll bar—Xv, Xh • (XTERM) 43 on/off save scroll text—Xv, Xh • (XTERM) 44 on/off margin bell—Xv, Xh • (XTERM) 45 on/off reverse wraparound—Xv, Xh • (XTERM) 47 alternate/normal screen buffer—Xv, Xh • (XTERM) 48 reverse/normal status line—Xv, Xh • (XTERM) 49 page/normal scroll mode—Xv, Xh
tbc	**Function (single-byte control)** tabulation clear **Data Stream** ESC [Ps g (default Ps =0) **Support** • 0 clear horizontal tab stop at active position—Xv, Xh, H, V • 1 vertical tab at line indicated by cursor—H • 2 horizontal tabs on line—H • 3 all horizontal tabs—Xv, Xh, H, V • 4 all vertical tabs—H

Item	Description
VTD	**Function (single-byte control)** virtual terminal data **Data Stream** ESC [x **Support** Xv, Xh, H
VTL	**Function (single-byte control)** virtual terminal locator report **Data Stream** ESC [y **Support** Xh, H
VTR	**Function (single-byte control)** vt raw keyboard input **Data Stream** ESC [w **Support** Xh, H
vts	**Function (single-byte control)** vertical tab stop **Data Stream** ESC I **Support** H
xes	**Function (single-byte control)** erase status line **Data Stream** ESC [? E **Support** Xv, Xh
xrs	**Function (single-byte control)** return from status line **Data Stream** ESC [? F **Support** Xv, Xh
xhs	**Function (single-byte control)** hide status line **Data Stream** ESC [? H **Support** Xv, Xh

Item	Description
xss	**Function (single-byte control)** show status line **Data Stream** ESC [? S **Support** Xv, Xh
xgs	**Function (single-byte control)** go to column of status line **Data Stream** ESC [? Ps T **Support** Xv, Xh
xst	**Function (single-byte control)** set text parameters • 0 change window name and title to Pt • 1 sets only the icon name • 2 sets only the title name • Everything between ESC-P and ESC\ is ignored. aixterm will work as usual after the ESC\. **Data Stream** ESC] Ps ; Pt \007 **Support** Xv, Xh

Copy, Paste, and Re-execute Functions

When you create a terminal window, the **aixterm** command allows you to select text and copy it within the same window or other windows by using copy, paste, and re-execute button functions. These text functions are available in HFT and VT102 emulations. The selected text is highlighted while the button is pressed.

The copy, paste, and re-execute button functions perform as follows:

Item	Description
Copy	The left button is used to save text into the cut buffer. The **aixterm** command does a text cut, not a box cut. Move the cursor to beginning of the text, hold the button down while moving the cursor to the end of the region, and release the button. The selected text is highlighted and saved in the global cut buffer and made the **PRIMARY** selection when the button is released. • Double clicking selects by words. • Triple clicking selects by lines. • Quadruple clicking goes back to characters, and so on. Multiple clicking is determined from the time the button is released to the time the button is pressed again, so you can change the selection unit in the middle of a selection. The right button extends the current selection. If you press this button while moving closer to the right edge of the selection than the left, it extends or contracts the right edge of the selection. If you contract the selection past the left edge of the selection, the **aixterm** command assumes you really meant the left edge, restores the original selection, and extends or contracts the left edge of the selection. Extension starts in the selection unit mode that the last selection or extension was performed in; you can multiple click to cycle through them.
Paste	Pressing both buttons at once (or the middle button on a three-button mouse) displays (pastes) the text from the **PRIMARY** selection or from the cut buffer into the terminal window that contains the mouse cursor, inserting it as keyboard input.
Re-execute	Pressing the Shift key and the left mouse button takes the text from the cursor (at button release) through the end of the line (including the new line), saves it in the global cut buffer and immediately retypes the line, inserting it as keyboard input. The selected text is highlighted. Moving the mouse cursor off of the initial line cancels the selection. If there is no text beyond the initial cursor point, the **aixterm** command sounds the bell, indicating an error.

By cutting and pasting pieces of text without trailing new lines, you can take text from several places in different windows and form a command to the shell. For example, you can take output from a program and insert it into your favorite editor. Since the cut buffer is globally shared among different applications, you should regard it as a file whose contents you know. The terminal emulator and other text programs should treat it as if it were a text file, that is, the text is delimited by new lines.

Menu Usage

The **aixterm** command has two different menus:

- Options
- Modes

Each menu pops up under the correct combinations of key and button presses. Most menus are divided into two sections that are separated by a horizontal line. The top portion contains various modes that can be altered. A check mark is displayed next to a mode that is currently active. Selecting one of these modes toggles its state. The bottom portion of the menu provides the command entries; selecting one of these performs the indicated function.

The Options menu open when the Ctrl key and the left mouse button are pressed simultaneously while the mouse cursor is in a window. The menu contains items that apply to all emulation modes.

The Modes menu sets various modes for each emulation mode. The menu is activated by pressing the Ctrl key and the middle mouse button at the same time, while the mouse cursor is in the window. In the command section of this menu, the soft reset entry resets the scroll regions. This is convenient when a program leaves the scroll regions set incorrectly. The full reset entry clears the screen, resets tabs to every eight columns, and resets the terminal modes (such as wrap and smooth scroll) to their initial states

after the **aixterm** command finishes processing the command-line options. When the Auto Linefeed option is turned on, a carriage return is added when a carriage return, vertical tab, or form feed is received. The shells generally do this for the linefeed, but not for the vertical tab or form feed.

Scroll Bar

The **aixterm** command supports an optional scroll bar composed of a scroll button that displays at the top of the scroll bar and a scroll region that displays at the bottom. The scroll bar is hidden until you request it to display.

The scroll region displays the position and amount of text currently showing in the window (highlighted) relative to the amount of text actually saved in the scrolling buffer. As more text is saved in the scrolling buffer (up to the maximum), the size of the highlighted area decreases.

The scroll button causes the window to scroll up and down within the saved text. Clicking the right button moves the window position up (the text scrolls downward); clicking the left button moves the window position down (the text scrolls upward). The amount of scrolling is modified by the Shift and Ctrl keys. If neither key is pressed, the window scrolls a single line at a time. Pressing the Shift key causes the text to scroll a full window at a time, minus one line. Pressing the Ctrl key causes the text to be positioned at the extreme top or bottom of the file.

Character Classes

Clicking the left mouse button (the copy function) twice in rapid succession causes all characters of the same class (that is, letters, white space, punctuation, and so on) to be selected. Because people have different preferences for what should be selected (for example, if file names be selected as a whole or only the separate subnames), you can override the default mapping by using the **charClass** (class **CharClass**) resource.

The **charClass** resource is a list of *CharRange***:***Value* pairs where the range is either a single number or a low-to-high number in the range of 0 to 127, corresponding to the ASCII code for the character or characters to be set. The value is arbitrary, although the default table uses the character number of the first character occurring in the set.

The default table is as follows:

```
static int charClass[128] = {
/* NUL  SOH  STX  ETX  EOT  ENQ  ACK  BEL */
   32,   1,   1,   1,   1,   1,   1,   1,
/*  BS   HT   NL   VT   NP   CR   SO   SI */
    1,  32,   1,   1,   1,   1,   1,   1,
/* DLE  DC1  DC2  DC3  DC4  NAK  SYN  ETB */
    1,   1,   1,   1,   1,   1,   1,   1,
/* CAN   EM  SUB  ESC   FS   GS   RS   US */
    1,   1,   1,   1,   1,   1,   1,   1,
/*  SP   !    "    #    $    %    &    ' */
   32,  33,  34,  35,  36,  37,  38,  39,
/*  (    )    *    +    ,    -    .    / */
```

```
   40,  41,  42,  43,  44,  45,  46,  47,
/*  0    1    2    3    4    5    6    7 */
   48,  48,  48,  48,  48,  48,  48,  48,
/*  8    9    :    ;    <    =    >    ? */
   48,  48,  58,  59,  60,  61,  62,  63,
/*  @    A    B    C    D    E    F    G */
   64,  48,  48,  48,  48,  48,  48,  48,
/*  H    I    J    K    L    M    N    O */
   48,  48,  48,  48,  48,  48,  48,  48,
/*  P    Q    R    S    T    U    V    W */
   48,  48,  48,  48,  48,  48,  48,  48,
/*  X    Y    Z    [    \    ]    ^    _ */
   48,  48,  48,  91,  92,  93,  94,  48,
/*  `    a    b    c    d    e    f    g */
   96,  48,  48,  48,  48,  48,  48,  48,
/*  h    i    j    k    l    m    n    o */
   48,  48,  48,  48,  48,  48,  48,  48,
/*  p    q    r    s    t    u    v    w */
   48,  48,  48,  48,  48,  48,  48,  48,
/*  x    y    z    {    |    }    ~   DEL */
   48,  48,  48, 123, 124, 125, 126,   1};
```

For example, the string "33:48,37:48,45-47:48,64:48" indicates that the ! (exclamation mark), % (percent sign), - (dash), . (period), / (slash), and & (ampersand) characters should be treated the same way as characters and numbers. This is very useful for cutting and pasting electronic mailing addresses and UNIX file names.

Key Translations

It is possible to rebind keys (or sequences of keys) to arbitrary strings for input. Changing the translations for events other than key and button events is not expected, and causes unpredictable behavior.

The actions available for key translations are as follows:

Item	Description
insert()	Processes the key in the normal way (that is, inserts the ASCII character code corresponding to the keysym found in the keyboard mapping table into the input stream).
string(*String***)**	Rebinds the key or key sequence to the string value; that is, inserts the string argument into the input stream. Quotation marks are necessary if the string contains white space or non-alphanumeric characters. If the string argument begins with the characters ``0x,'' it is interpreted as a hex character constant and the corresponding character is sent in the normal way.
keymap(*Name***)**	Takes a single string argument naming a resource to be used to dynamically define a new translation table; the name of the resource is obtained by appending the string Keymap to *Name*. The keymap name **None** restores the original translation table (the very first one; a stack is not maintained). Uppercase and lowercase is significant.
insert-selection(*Name*[,*Name*]...**)**	Retrieves the value of the first (leftmost) named selection that exists and inserts the value into the input stream. The *Name* parameter is the name of any selection, for example, **PRIMARY** or **SECONDARY**. Uppercase and lowercase is significant.

For example, a debugging session might benefit from the following bindings:

```
*aixterm.Translations: #override <Key>F13: keymap(dbx)
*aixterm.dbxKeymap.translations:\
<Key>F14: keymap(None) \n\
<Key>F17: string("next") string(0x0d) \n\
<Key>F18: string("step") string(0x0d) \n\
<Key>F19: string("continue") string(0x0d) \n\
<Key>F20: string("print") insert-selection(PRIMARY)
```

Key and Button Bindings

The key and button bindings for selecting text, pasting text, and activating the menus are controlled by the translation bindings. In addition to the actions listed in the Key Translations section, the following actions are available:

Item	Description
mode-menu()	Posts one of the two mode menus, depending on which button is pressed.
select-start()	Deselects any previously selected text and begins selecting new text.
select-extend()	Continues selecting text from the previous starting position.
start-extend()	Begins extending the selection from the farthest (left or right) edge.

Item	Description
select-end(*Name*[**,***Name*]...**)**	Ends the text selection. The *Name* parameter is the name of a selection into which the text is to be copied. The **aixterm** command asserts ownership of all the selections named. Uppercase and lowercase is significant.
ignore()	Quietly discards the key or button event.
bell([*Volume*]**)**	Rings the bell at the specified volume increment above or below the base volume.

The default bindings are:

```
static char defaultTranslations =
"                     <KeyPress>:  insert() \n\
~Shift Ctrl   ~Meta  <Btn1Down>:   mode-menu(options) \n\
~Shift Ctrl   ~Meta  <Btn2Down>:   mode-menu() \n\
~Shift Ctrl   ~Meta  <Btn3Down>:   mode-menu(modes) \n\
~Shift ~Ctrl  ~Meta  <Btn1Down>:   select-start() \n\
~Shift ~Ctrl  ~Meta  <Btn1Motion>: select-extend() \n\
~Shift ~Ctrl  ~Meta  <Btn1Up>:     select-end(PRIMARY)\n\
~Shift ~Ctrl  ~Meta  <Btn2Down>:   ignore() \n\
~Shift ~Ctrl  ~Meta  <Btn2Up>:     insert-selection(PRIMARY)\n\
~Shift ~Ctrl  ~Meta  <Btn3Down>:   start-extend() \n\
~Shift ~Ctrl  ~Meta  <Btn3Motion>: select-extend() \n\
~Shift ~Ctrl  ~Meta  <Btn3Up>:     select-end(PRIMARY)\n\
Shift ~Ctrl   ~Meta  <Btn1Down>:   reexecute() \n\
Shift ~Ctrl   ~Meta  <Btn1Motion>: select-extend() \n\
Shift ~Ctrl   ~Meta  <Btn1Up>:     select-end(PRIMARY)\n\
Shift ~Ctrl   ~Meta  <Btn2Down>:   select-start() \n\
Shift ~Ctrl   ~Meta  <Btn2Motion>: select-extend() \n\
Shift ~Ctrl   ~Meta  <Btn2Up>:     select-end(PRIMARY)\n\
Shift ~Ctrl   ~Meta  <Btn3Down>:   ignore() \n\
Shift ~Ctrl   ~Meta  <Btn3Up>:     insert-selection(PRIMARY)\n\
Shift Ctrl    ~Meta  <BtnDown>:    size(toggle) \n\
Shift Ctrl    ~Meta  <BtnUp>:      ignore() \n\
                     <BtnDown>:    bell(0) \n\
                     <BtnUp>:      bell(0) \n\
";
```

aixterm Command Internationalization (I18N)

To run an aixterm with a different keyboard layout than the X server's (such as a French keyboard layout on a Swiss German X server), run the following commands:

1. Change the X server to a French keyboard:

   ```
   xmodmap /usr/lpp/X11/defaults/xmodmap/Fr_FR/keyboard
   ```

2. Set the locale environment variable to Fr_FR using one of the following:
   - For Korn shells: `export LANG=Fr_FR`
   - For C shells: `setenv LANG Fr_FR`
   - For Bourne shells: `LANG=Fr_FR; export LANG`
3. Start an aixterm terminal emulator:

   ```
   aixterm &
   ```

4. Reset the X server's keyboard file to its original language:

   ```
   xmodmap /usr/lpp/X11/defaults/xmodmap/Gr_SW/keyboard
   ```

The **aixterm** command continues to use the keyboard layout that the X server was using when the aixterm started. It ignores **KeymapNotify** by default.

The **aixterm** command uses the Input Method to convert the X server's keysyms into either printable characters or nonprintable escape strings such as function keys. The Input Method uses its own keymap files, in **/usr/lib/nls/loc**, to convert X keysyms into code points for the printable characters,

and escape strings for nonprintable characters. There is a keymap file for each language and one keymap file for escape sequences. The escape sequences are in **C@outbound.imkeymap**; the source is **C@outbound.imkeymap.src**. The other keymap files begin with the locale name and look like: **locale.imkeymap** and **locale.codeset.imkeymap**. For example:

Item	Description
US English in codeset IBM-850	En_US.IBM-850.imkeymap
US English in codeset ISO8859-1	en_US.ISO8859-1.imkeymap
Turkish in codeset ISO8859-9	tr_TR.ISO8859-9.imkeymap
Japanese in codeset IBM-943	Ja_JP.IBM-943.imkeymap
Japanese in codeset EUC(JP)	ja_JP.IBM-eucJP.imkeymap

The following dependencies apply:

- You can change the locale by entering the following SMIT fast path: `smit mle_sel_menu`. You can also change the locale temporarily by modifying the LANG environment variable.
- You can change the system keyboard definition by selecting the following SMIT menu items: System Environments, Manage Language Environment, and Change the Keyboard Map for the Next System Restart.
- Codeset depends on the locale (LC_ALL, LANG environment variables).
- Default fonts and font sets depend on the codeset and locale. Using a font that does not match the codeset may produce incorrect output.
- Input Method depends on the locale. The Input Method for the locale should be installed. The Input Method maps Keysyms to a codeset.
- Compose keys (dead keys) depend on the Input Method and X keyboard mapping. An incorrect input method or X keyboard mapping may produce incorrect input.
- Error messages and menu contents depend on the locale and a correct font or fontset. The message catalogs for the locale should be installed. The default messages are English. An incorrect font or fontset can result in garbled menu text and messages.
- Text display depends on the locale and a correct font or fontset. An incorrect font or fontset can result in garbled text. Changing the locale (LC_ALL, LANG environment variables) in an aixterm does not change the codeset that the aixterm displays. If the codeset of the new locale differs from the codeset of **aixterm**, incorrect output (garbled text) may be displayed.
- The X keyboard mapping depends on the system keyboard definition. Xinit sets the X keyboard mapping to match the system keyboard definition. The mapping is changed with **xmodmap**. The X keyboard mapping maps key presses to Keysyms.

Availability of Characters in aixterm

ASCII characters 32 (0x20) to 126 (0x7e) are available in most of the codesets and fonts. Characters (bytes) 0 (0x00) to 31 (0x1f) are treated as control sequences and unprintable characters. Other characters 127 (0x7f) to 255 (0xff) vary with codeset and fonts. Using a font that does not match the codeset the aixterm is started in leads to unpredictable results. For example, box characters (line drawing) are available in **aixterm** vt100 mode with the default vtsingle font. If you use a different font, other characters may be displayed instead. Another example is using a ISO8859-1 font while running in the IBM-850 codeset. Trying to display box characters (line drawing) generates accented characters. Trying to display accented characters generates different accented characters or blanks.

Key Assignments for Bidirectional Languages

In addition to the above key and button bindings, the following key assignments for bidirectional languages are supported by the **aixterm** command:

Item	Description
scr-rev()	Reverses the screen orientation and sets the keyboard layer to the default language of the new orientation.
ltr-lang()	Enables the English keyboard layer.
rtl-lang()	Enables the Arabic/Hebrew keyboard layer.
col-mod()	Enables the column heading adjustment which handles each word as a separate column.
auto-push()	Toggles the Autopush function. This function handles mixed left-to-right and right-to-left text. When you enable the Autopush function, reversed segments are automatically initiated and terminated according to the entered character or the selected language layer. Thus, you are relieved of manually invoking the Push function.
chg-push()	Toggles the Push mode. This mode causes the cursor to remain in its position and pushes the typed characters in the direction opposed to the field direction.
shp-in()	Shapes Arabic characters in their initial forms.
shp-is()	Shapes Arabic characters in their isolated forms.
shp-p()	Shapes Arabic characters in their passthru forms.
shp-asd()	Shapes Arabic characters in their automatic forms.
shp-m()	Shapes Arabic characters in their middle forms.
shp-f()	Shapes Arabic characters in their final forms.

The BIDI bindings (for Arabic/Hebrew) are:

```
~Shift ~Ctrl Mod1   <Key>Return:         scr-rev() \n\
~Shift ~Ctrl Mod2   <Key>Return:         scr-rev() \n\
~Shift ~Ctrl Mod1   <Key>Shift_L: ltr-lang() \n\
~Shift ~Ctrl Mod2   <Key>Shift_L: ltr-lang() \n\
~Shift ~Ctrl Mod1   <Key>Shift_R: rtl-lang() \n\
~Shift ~Ctrl Mod2   <Key>Shift_R: rtl-lang() \n\
~Shift ~Ctrl Mod1   <Key>KP_Multiply:    col-mod() \n\
~Shift ~Ctrl Mod2   <Key>KP_Multiply:    col-mod() \n\
~Shift ~Ctrl Mod1   <Key>KP_Divide:      auto-push() \n\
~Shift ~Ctrl Mod2   <Key>KP_Divide:      auto-push() \n\
~Shift ~Ctrl ~Meta  <Key>KP_Divide:      chg-push() \n\
~Shift ~Ctrl Mod1   <Key>KP_1:           shp-in() \n\
~Shift ~Ctrl Mod2   <Key>KP_2:           shp-in() \n\
~Shift ~Ctrl Mod1   <Key>KP_1:           shp-is() \n\
~Shift ~Ctrl Mod1   <Key>KP_2:           shp-is() \n\
~Shift ~Ctrl Mod1   <Key>KP_3:           shp-p() \n\
~Shift ~Ctrl Mod2   <Key>KP_3:           shp-p() \n\
~Shift ~Ctrl Mod1   <Key>KP_4:           shp-asd() \n\
~Shift ~Ctrl Mod2   <Key>KP_4:           shp-asd() \n\
~Shift ~Ctrl Mod1   <Key>KP_7:           shp-m() \n\
~Shift ~Ctrl Mod2   <Key>KP_7:           shp-m() \n\
~Shift ~Ctrl Mod1   <Key>KP_8:           shp-f() \n\
~Shift ~Ctrl Mod2   <Key>KP_8:           shp-f() \n\
```

You can change these values in the **.Xdefaults** file. For example, if you want to use Ctrl+Shift to change language layer, you can add the following line in the **.Xdefaults** file:

```
Translations:   Ctrl<Key>Shift_R: rtl-lang() \n\
                Ctrl<Key>Shift_L: ltr-lang()
```

Flags

A flag takes on the opposite value if the - (minus sign) is changed to a + (plus sign). The following options override those set in the **.Xdefaults** file:

Item	Description
-ah	Highlights the cursor at all times.
-ar	Turns on the autoraise mode of **aixterm**, which automatically raises the window (after a delay determined by the **.Xdefaults** keyword **autoRaiseDelay**) when the mouse cursor enters the window. The default is off. This flag can be turned on and off from the Options menu.
- autopush	Enables the Autopush function for the visual text type.
-b *NumberPixels*	Specifies the width in pixels of an inner border. The inner border is the distance between the outer edge of the characters and the window border. The default is 2.
-bd *Color*	Specifies the color of the highlighted border on color displays. The default is black.
-bg *Color*	Specifies the color of the window background on color displays. The default is white.
-bw *NumberPixels*	Specifies the width of the window border in pixels. The default is 2 pixels. Some window managers can override this option.
-C	Intercepts console messages.
-cc*CharRange***:***Value***,**...	Changes the types of characters that are part of a word. For example, the string `-cc 48-52:3` would make the characters 01234 one word and 56789 a different word. The `:3` defines a word group number 3. By default, numbers are in class 48. The character classes are used by cut and paste.
-cr *Color*	Determines the color of the text cursor on color displays. The default is the foreground color.
-csd *CharShape*	Specifies the default shape of Arabic text. The *CharShape* variable can be one of the following options: **automatic** Shapes the characters automatically. **passthru** Does not shape the characters. The characters are displayed in the same way that they are entered. **isolated** Displays the characters in their isolated form (valid in visual mode only). **initial** Displays the characters in their initial form (valid in visual mode only). **middle** Displays the characters in their middle form (valid in visual mode only). **final** Displays the characters in their final form (valid in visual mode only).

Item	Description
-cu	Causes certain curses applications to display leading tabs correctly. The default is off. This flag can be turned on and off from the Modes menu.
-display *Name***:***Number*	Identifies the host name and X Server display number where the **aixterm** command is to run. By default, **aixterm** gets the host name and display number from the **DISPLAY** environment variable.
-dw	Causes the mouse cursor to move (warp) automatically to the center of the **aixterm** window when the **aixterm** icon window is deiconified. The default is off.
-e *Command*	Specifies a command to be executed in the window. This flag runs the command; it does not start a shell. If this flag is used, the command and its arguments (if any) must be displayed last on the **aixterm** command line. When the command exits, the **aixterm** command exits.
-f0 *Font*	Specifies the name of the default font on the command line. Also specifies the name of the font placed in position 0 in the font table. This flag is similar to the **-fn** flag. For example, to specify a default font on the command line, enter the following: `aixterm -f0 rom11`
-f1 *Font*	Specifies the name of the font placed in position 1 in the font table. This flag is similar to the **-fb** flag.
-f2 *Font*	Specifies the name of the font placed in position 2 of the font table. This flag is similar to the **-fi** flag.
-f3 *Font*	Specifies the name of the font placed in position 3 of the font table.
-f4 *Font*	Specifies the name of the font placed in position 4 of the font table.
-f5 *Font*	Specifies the name of the font placed in position 5 of the font table.
-f6 *Font*	Specifies the name of the font placed in position 6 of the font table.
-f7 *Font*	Specifies the name of the font for position 7 in the font table.
—f0 *FontSet*	Specifies the name of the font set for position 0 in the font table. This flag is similar to the **-fn** flag.
—f1 *FontSet*	Specifies the name of the font set for position 1 in the font table. This flag is similar to the **-fb** flag.
—f2 *FontSet*	Specifies the name of the font set for position 2 in the font table. This flag is similar to the **-fi** flag.
—f3 *FontSet*	Specifies the name of the font set for position 3 in the font table.
—f4 *FontSet*	Specifies the name of the font set for position 4 in the font table.
—f5 *FontSet*	Specifies the name of the font set for position 5 in the font table.
—f6 *FontSet*	Specifies the name of the font set for position 6 in the font table.

Item	Description
—f7 *FontSet*	Specifies the name of the font set for position 7 in the font table.
-fb *Font*	Specifies the name of the bold font. This font must be the same height and width as the normal font.
-fi *FontSet*	Specifies the name of the italic font set.
-fg *Color*	Determines the foreground color of the text on color displays. The default is black.
-fn *Font*	Specifies the name of a normal full-text font set. Any fixed-width font set can be used. In HFT emulation, the default is **Rom14.500** for a large display or **Rom10.500** for a small display. In VT102 emulation, the default is **vtsingle**. To specify a font set in the resource file, use **aixterm.Fontset** *FontSet*.
-fs *Font*	Specifies the name of the special graphics font.
-fullcursor	Uses a full block cursor instead of the default underscore cursor.
-geometry *Geometry*	Specifies the location and dimensions of a window. The default is **80x25+0+0**. Some window managers (such as the **mwm** command) can override these defaults.
#geometry*Geometry*	Specifies the location of an icon window. If specified, width and height are ignored. Width and height are taken from the size of the bitmap and the length of the title. The window manager can override the location of the icon. **Note:** When you use one of these values as part of an **sh** (shell) command, enclose the value in "" (double quotation marks). Normally, #(the pound sign) indicates a comment in a shell script.
-help	Lists the available option flags.
-i	Displays the icon window rather than the normal window when the window is opened. The default is false. **Note:** This flag does not work unless the window manager has started.
-ib *File*	Specifies name of the bitmap file to read for use as the icon bitmap file instead of the default bitmap file. You can access a **/usr/include/X11/bitmaps** file from an operating system shell to see a sample bitmap file.
-im *InputMethod*	Specifies a modifier string that identifies the input method to be used by the **aixterm** command.
-j	Causes the **aixterm** command to move multiple lines up at once (jump scroll) if many lines are queued for display. The default is false. This flag can be turned on and off from the Modes menu.
-keywords	Lists the **.Xdefaults** keywords.
-lang *Language*	Specifies the language to be used under the **aixterm** command. The language should follow the format for the locale, as used by the **setlocale** function.

Item	Description
-l	Causes the **aixterm** command to append output from the window to the end of the **logfile** file. The default is false. This flag can be turned on and off from the Options menu. This does not override **LogInhibit** in the **.Xdefaults** file.
-leftscroll	Places the scroll bar on the left when it is displayed. The default is on the right side of the text window.
-lf *File*	Specifies the file where the output is saved, instead of the default **AixtermLog.***XXXXXX* file, where *XXXXXX* is the process ID of the **aixterm** command. The file is created in the directory where the **aixterm** command is started, or in the home directory for a login **aixterm** command. If the file name begins with a | (pipe symbol), the rest of the string is interpreted as a command to be executed by the shell, and a pipe is opened to the process. This flag must be used in conjunction with the **-l** flag to work effectively.
-ls	Causes the shell run under the **aixterm** command to be a login shell. The user's **.login** or **.profile** file is read, and the initial directory is usually the home directory. The default is false.
-mb	Turns on the right margin bell. The default is false. This flag can be turned on and off from the Modes menu.
-mc *Number*	Determines the multiple-click time. This is used by the cut and paste button functions.
-mn	Ignores the **XMappingNotify** event. The **-mn** flag is the default.
-ms *Color*	Determines the color of the mouse cursor on color displays. The default is the foreground color.
-n *IconName*	Specifies the icon name for use by the **aixterm** command.
-name *Application*	Specifies the application name to use for the **.Xdefaults** file.
-nb *Number*	Specifies the right margin distance at which the margin bell rings. The default is 10 spaces from the right edge of the window.
-nobidi	Disables the Arabic/Hebrew functions such as screen reverse, while maintaining an Arabic/Hebrew locale.
- nonulls	Enables a Nonulls mode in which nulls within a line are replaced by spaces.
-nss *NumShape*	Specifies the default shape of numerals. The *NumShape* variable can be one of the following options: **bilingual** Displays numerals according to the surrounding text. For example, Arabic numerals are displayed within Arabic text and English numerals within English text. **hindi** Displays numerals in Hindi. **arabic** Displays numerals in Arabic. **passthru** Displays numerals the same way they are entered.

Item	Description
- orient *Orientation*	Specifies the default screen orientation. The orientation can be one of the following options: **LTR** Left-to-right screen orientation **RTL** Right-to-left screen orientation
-outline *Color*	Determines the color of the outline attribute (Keisen) on color displays. The default is the foreground color. The outline attribute for a character is similar to other character attributes such as bold or reverse video. The outline attribute is displayed as a box drawn to enclose a character or group of characters.
-po *Number*	Specifies the number of lines from the previous screen that display on the screen when the window scrolls one page. The default is 1 line.
-ps	Turns on the page scroll mode. After a page of lines is displayed, the **aixterm** command stops displaying new lines and the text cursor is no longer displayed. Pressing the Enter key displays one new line. Pressing the Spacebar key or a character key displays a new page. The default is false.
-pt *Preedit*	Specifies the pre-edit type for text composing. The possible pre-edit types are: **over** Places the pre-edit window over the spot of character composition. **off** Places the pre-edit window off the spot of character composition in the status area. **root** Composes character outside of the current window tree. **none** Specifies that the input method has no pre-edit area.
-reduced	Causes the **aixterm** command to begin in reduced mode.
-rfb *Font*	Specifies the name of the reduced bold font. This font must be the same width and height as the reduced normal font.
-rfi *Font*	Specifies the name of the reduced italic font. This font must be the same width and height as the reduced normal font.
-rfn *Font*	Specifies the name of the reduced normal font.
-rfs *Font*	Specifies the name of the reduced special graphics font.
-rf0 *Font*	Specifies the name of the reduced font placed in position 0 in the font table. This flag is similar to the **-rfn** flag.
-rf1 *Font*	Specifies the name of the reduced font placed in position 1 in the font table. This flag is similar to the **-rfb** flag.
-rf2 *Font*	Specifies the name of the reduced font placed in position 2 in the font table. This flag is similar to the **-rfi** flag.

Item	Description
-rf3 *Font*	Specifies the name of the reduced font placed in position 3 in the font table.
-rf4 *Font*	Specifies the name of the reduced font placed in position 4 in the font table.
-rf5 *Font*	Specifies the name of the reduced font placed in position 5 in the font table.
-rf6 *Font*	Specifies the name of the reduced font placed in position 6 in the font table.
-rf7 *Font*	Specifies the name of the reduced font placed in position 7 in the font table.
—rf0 *FontSet*	Specifies the name of the reduced fontset placed in position 0 in the font table. This flag is similar to the **-rfn** flag.
—rf1 *FontSet*	Specifies the name of the reduced fontset placed in position 1 in the font table. This flag is similar to the **-rfb** flag.
—rf2 *FontSet*	Specifies the name of the reduced fontset placed in position 2 in the font table. This flag is similar to the **-rfi** flag.
—rf3 *FontSet*	Specifies the name of the reduced fontset placed in position 3 in the font table.
—rf4 *FontSet*	Specifies the name of the reduced fontset placed in position 4 in the font table.
—rf5 *FontSet*	Specifies the name of the reduced fontset placed in position 5 in the font table.
—rf6 *FontSet*	Specifies the name of the reduced fontset placed in position 6 in the font table.
—rf7 *FontSet*	Specifies the name of the reduced fontset placed in position 7 in the font table.
-rv	Reverses the foreground and background colors. This becomes the normal video mode. This flag can be turned on and off from the Modes menu.
-rw	Turns on the reverse-wraparound mode. The default is false. This mode allows the cursor to wraparound from the leftmost column to the rightmost column of the previous line. This can be useful in the shell to allow erasing characters backwards across the previous line. This flag can be turned on and off from the Modes menu.
-s	Turns off synchronous scrolling on the display. The default is true. When this flag is specified, the **aixterm** command no longer attempts to keep the screen current while scrolling and can run faster when network latencies are very high.
-sb	Causes the scroll bar to display. This flag can be turned on and off from the Modes menu. The default is off.
-sf	Generates the Sun function keycodes for programmed-function (PF) keys in VT102 mode.

Item	Description
-si	Specifies that while using the scroll bar to review previous lines of text, the window is normally repositioned automatically at the bottom of the scroll region before output to the screen is processed. The default is true. This flag disables window repositioning on output.
-sk	Causes the window to be repositioned automatically in the normal position at the bottom of the scroll region when a key is pressed. The default is false. This flag is intended for use with the scroll bar to review previous lines of text. Pressing a key also creates output, which is affected by the **-si** flag. This flag can be turned on and off from the Scrollbar menu.
-sl *NumberLines*	Specifies the maximum number of lines to save that scroll off of the top of the window. The default is 64.
-sn	Displays the status line to be displayed in normal video (the status line is still enclosed in a box). By default, the status line is displayed in reverse-video relative to the rest of the window. This flag can be turned on and off from the Modes menu.
-st	Displays the status line on startup. The default is false.
-suppress	Specifies that the preediting function in the input method **IMIoctl** call is suppressed.
- symmetric	Enables the Symmetric Swapping mode for handling bidirectional character pairs such as <> and ().
-T *Title*	Sets the title bar name, but not the icon name. If the **-n** option is not specified, or the icon name is not a specified keyword in the **.Xdefaults** file, the title is used as the icon name.
-text *TextType*	Specifies the type of data stream. The *TextType* variable can be one of the following options: **- implicit** Characters are stored in key stroke order. **- visual** Characters are stored the same way that they are displayed. You can use the Autopush mode or Push mode with different shape types.
-ti	Displays the title to the right of the bitmap in the icon window. By default, the title is displayed under the bitmap (if the window manager allows it).
-tm *String*	Specifies a series of terminal setting keywords followed by the characters that should be bound to those functions. Allowable keywords include: **intr**, **quit**, **erase**, **kill**, **eof**, **eol**, **start**, **stop**, **susp**, **dsusp**, **rprnt**, **flush**, **weras**, and **lnext**.

Item	Description
-tn *TerminalName*	Specifies the terminal environment variable. Use the **-tn** flag to change the terminal environment variable only. The terminal environment variable should not be changed to match the terminal in which the X Server is running. The **aixterm** command has no direct access to the terminal where the X Server is running.
-ut	Disables the addition of the login ID to **/etc/utmp**.
-v	Enables VT102 emulation. By default, HFT is emulated. **Note:** The keyboard map is needed for this mode.
-vb	Enables the visual bell mode. The visual bell flashes the window on receipt of the Ctrl-G key combination instead of ringing the bell. The default is false.
-W	Causes the mouse cursor to move (warp) to the middle of the **aixterm** window when the window is created. The default is false.
-xrm *String*	Sets the resource string. For example, `aixterm.foreground: blue`
-132	Causes the sm/rm escape sequences to be recognized and the **aixterm** window to be resized as specified. Normally, the sm/rm escape sequences that switch between the 80-column and 132-column modes are ignored. The default is false. This flag can be turned on and off from the Modes menu.

.Xdefaults Keywords

Use the following keywords to set the defaults for the **aixterm** command.

Item	Description
alwaysHighlight	If true, always highlights the cursor, even when the mouse pointer is outside the window.
autoRaise	If true, raises the **aixterm** window automatically (after a delay of **autoRaiseDelay**) when the mouse cursor enters the window. The default is false. Window managers can override this option.
autoRaiseDelay	If **autoRaise** is true, specifies the number of seconds to delay before automatically raising a window. The default is 2 seconds. Window managers can override this option.
background	Specifies the color of the window background on color displays. The default is a white background.
boldFontSet	Specifies the name of a bold font. This font must have the same height and width as the normal sized font.
borderColor	Specifies the color of the window border. Window managers can override this option.
borderWidth	Specifies the width of the window border in pixels. The default is 2 pixels.
c132	If true, specifies that the sm/rm escape sequences to resize the **aixterm** window between 80 and 132 columns be recognized. The default is false.
charClass	Specifies the character class.

Item	Description
charShape	If set to automatic, the characters are shaped automatically. If set to passthru, the characters do not exert any shaping. If set to isolated, the characters are displayed in isolated shape. If set to initial, the characters are displayed in initial shape. If set to final, the characters are displayed in final shape.
console	If set to true, the **aixterm** command intercepts console messages. The default is false.
curses	If true, causes certain curses applications to display leading tabs correctly. The default is false.
cursorColor	Specifies the color of the text cursor on color displays. The default is the foreground color.
deiconifyWarp	If true, moves or warps the mouse to the center of the window when replacing the **aixterm** icon window with the **aixterm** window. The default is false.
expandTail	The "seen", "sheen", "sad", "dad" Arabic characters and their tails are displayed as two characters.
fASD	Enables the automatic shaping function.
fAutoPush	Enables the Autopush function.
fEndPush	Enables the End Push function.
fLTR	Enables the LTR screen orientation.
font0	Specifies the name of the font placed in position 0 in the font table. This flag is similar to the **-fn** flag.
font1	Specifies the name of the font placed in position 1 in the font table. This flag is similar to the **-fb** flag.
font2	Specifies the name of the font placed in position 2 of the font table. This flag is similar to the **-fi** flag.
font3	Specifies the name of the font placed in position 3 of the font table.
font4	Specifies the name of the font placed in position 4 of the font table.
font5	Specifies the name of the font placed in position 5 of the font table.
font6	Specifies the name of the font placed in position 6 of the font table.
font7	Specifies the name of the font for position 7 in the font table.
fontSet	Specifies the name of the normal sized text font used in the body of the **aixterm** window.
fontSet0	Specifies the name of the font set for position 0 in the font table. This flag is similar to the **-fn** flag.
fontSet1	Specifies the name of the font set for position 1 in the font table. This flag is similar to the **-fb** flag.
fontSet2	Specifies the name of the font set for position 2 in the font table. This flag is similar to the **-fi** flag.
fontSet3	Specifies the name of the font set for position 3 in the font table.
fontSet4	Specifies the name of the font set for position 4 in the font table.
fontSet5	Specifies the name of the font set for position 5 in the font table.
fontSet6	Specifies the name of the font set for position 6 in the font table.

Item	Description
fontSet7	Specifies the name of the font set for position 7 in the font table.
foreground	Specifies the color for the text displayed inside the body of the window on color displays. The default is black.
fPush	Enables the Push function.
fRTL	Enables the RTL screen orientation.
fScrev	Enables the Screen Reverse function.
fShapeF	Enables the Final Shape function.
fShapeIN	Enables the Initial Shape function.
fShapeIS	Enables the Isolated Shape function.
fShapeM	Enables the Middle Shape function.
fShapeP	Enables the Passthru shape function.
fullCursor	Displays the full cursor. The default is an underscore cursor.
geometry	Specifies the location or dimensions of the window.
iconBitmap	Reads the bitmap file name and uses the resulting bitmap as the icon.
iconGeometry	Specifies the location of the icon window.
iconName	Specifies the icon name.
iconStartup	If true, causes the **aixterm** command to start by displaying an icon window rather than the normal window.
inputMethod	Specifies the input method to be used by the **aixterm** command.
internalBorder	Specifies the number of pixels between the text characters and the window border. The default is 2 pixels.
italicFontSet	Specifies the name of the italic font set.
jumpScroll	If true, enables jump scroll. The default is false.
language	Specifies the language to be used under the **aixterm** command. The language should follow the format for the locale, as used by the **setlocale** function.
logFile	If **logging** is true, specifies the file in which the log is written. The default is **AixtermLog.***XXXXXX*, where *XXXXXX* is a unique ID of the **aixterm** command.
logging	If true, appends all input from the pseudo tty to the logfile. The default is false.
logInhibit	If true, prevents a user or an application program from enabling logging. This overrides any values set for **logging**.
loginShell	If true, indicates that the **aixterm** command should start as a login shell. The default is false.
mappingNotify	If set to false, ignores the **XMappingNotify** event. The default is false.
marginBell	If true, enables the right margin bell. The default is false.
multiClickTime	Specifies the number of milliseconds between button clicks when cutting and pasting. The default is 250 milliseconds.
multiScroll	If true, allows asynchronous scrolling.

Item	Description
nMarginBell	Specifies the distance from the right edge of the window where the margin bell rings. The default is 10 spaces from the right edge of the window.
noNulls	Replaces nulls with spaces within a line.
numShape	If set to bilingual, the numbers are shaped according to context. If set to hindi, the numbers are represented in Arabic. If set to arabic, the numbers are represented in English. If set to passthru, the numbers are represented as they are.
orientation	If set to LTR, left-to-right is set as the default screen orientation. If set to RTL, right-to-left is set as the default screen orientation.
outline	Determines the color of the outline attribute (Keisen) on color displays. The default is the foreground color. The outline attribute for a character is similar to other character attributes such as bold or reverse video. The outline attribute is displayed as a box drawn to enclose a character or group of characters.
pageOverlap	Specifies the number of lines from the previous screen that remain on the screen when the terminal scrolls one page. In page scroll mode, a page is the number of lines in the scrolling region minus the page overlap. The default is 1 line.
pageScroll	If true, enables the page scroll mode. The default is false. After a page of lines displays, **aixterm** stops displaying new lines and the text cursor disappears. Pressing the Enter key displays one new line. Pressing the Spacebar key or a character key displays a new page.
preeditType	Specifies the pre-edit type for text composing. The possible pre-edit types are:
over	Places the pre-edit window over the spot of character composition.
off	Places the pre-edit window off the spot of character composition in the status area.
root	Composes character outside of the current window tree.
none	Specifies that the input method has no pre-edit area.
pointerColor	Specifies the color of the mouse cursor on color displays. The default is the foreground color.
pointerShape	Specifies the shape of the mouse cursor to be used in an **aixterm** window. The default is **XC_xterm**. The cursors are listed in the **/usr/include/X11/cursorfont.h** file.
reducedBoldFontSet	Specifies the name of the reduced fontset placed in position 1 in the font table.
reducedFont0	Specifies the name of the reduced font placed in position 0 in the font table.
reducedFont1	Specifies the name of the reduced font placed in position 1 in the font table.
reducedFont2	Specifies the name of the reduced font placed in position 2 in the font table.

Item	Description
reducedFont3	Specifies the name of the reduced font placed in position 3 in the font table.
reducedFont4	Specifies the name of the reduced font placed in position 4 in the font table.
reducedFont5	Specifies the name of the reduced font placed in position 5 in the font table.
reducedFont6	Specifies the name of the reduced font placed in position 6 in the font table.
reducedFont7	Specifies the name of the reduced font placed in position 7 in the font table.
reducedFontSet	Specifies the name of the reduced fontset placed in position 0 in the font table.
reducedFontSet0	Specifies the name of the reduced fontset placed in position 0 in the font table.
reducedFontSet1	Specifies the name of the reduced fontset placed in position 1 in the font table.
reducedFontSet2	Specifies the name of the reduced fontset placed in position 2 in the font table.
reducedFontSet3	Specifies the name of the reduced fontset placed in position 3 in the font table.
reducedFontSet4	Specifies the name of the reduced fontset placed in position 4 in the font table.
reducedFontSet5	Specifies the name of the reduced fontset placed in position 5 in the font table.
reducedFontSet6	Specifies the name of the reduced fontset placed in position 6 in the font table.
reducedFontSet7	Specifies the name of the reduced fontset placed in position 7 in the font table.
reducedItalicFontSet	Specifies the name of the reduced fontset placed in position 2 in the font table.
reducedSpecialFont	Specifies the name of the reduced special graphics font.
reducedStartup	Causes the **aixterm** command to begin in reduced mode.
reverseVideo	If true, reverses the foreground and background color. The default is false.
reverseWrap	If true, sets reverse-wraparound mode, which allows the cursor to wrap from the leftmost column to the rightmost column of the previous line. The default is false.
rtArrow	The Right Arrow key is handled as a movement key.
saveLines	Specifies the maximum number of lines to save when lines scroll off the top of a window. The default is 64 lines.
scrollBar	If true, displays the scroll bar during startup.
scrollInput	Specifies whether output to the terminal automatically causes the scroll bar to go to the bottom of the scrolling region. The default is true.

Item	Description
scrollKey	If true, repositions the window at the bottom of the scroll region (normal position) when a key is pressed while using the scroll bar to review previous lines of text. The default is false. Pressing a key also creates input, which is affected by the **scrollInput** keyword.
scrollPosition	If left, positions the scroll bar to the left side of the screen. The default is right.
signalInhibit	If true, specifies that the signals should not be listed. The default is false.
specialFont	Specifies the name of the special graphics font.
statusLine	If true, displays the status line on startup. The default is false.
statusNormal	If true, displays the status line in normal video (the status line is still enclosed in a box). By default, the status line is in reverse-video relative to the rest of the window.
sunFunctionKeys	If true, the PF keys generate Sun function keycodes when in the VT102 mode. The default is false.
suppress	If true, specifies that the pre-editing function in the input method **IMIoctl** call is suppressed.
symmetric	Enables symmetric character swapping.
termName	Specifies the terminal environment variable, **$TERM**. Use the **termName** keyword to change the terminal environment variable only. The terminal environment variable should not be changed to match the terminal in which the X Server is running. The **aixterm** command has no direct access to the terminal where the X Server is running.
textType	If set to implicit, the data stream type is set to implicit. If set to visual, the data stream type is set to visual.
textUnderIcon	If False, displays the title of the icon window at the right of the bitmap in the icon window. By default, the title is displayed under the bitmap.
title	Specifies the title to show in the title bar. The default is **aixterm**.
ttyModes	Specifies the tty settings.
translations	Specifies the key and button translations to be supplied.
utmpInhibit	If False, adds the login ID to the **/etc/utmp** file. The default is false**.**
visualBell	If true, enables the visual bell mode which flashes the window on receipt of a Ctrl-G key sequence. The default is false.
vt102	If true, enables VT102 mode. The default is emulation.
warp	If true, automatically warps (moves) the mouse cursor to the center of a newly created **aixterm** window. The default is false.

Example

The following example can be used to create an **aixterm**, specifying the size and location of the window, using a font other than the default, and also specifying the foreground color that is used in text. The **aixterm** command then runs a command in that window.

```
aixterm -geometry 20x10+0+175 -fn Bld14.500 -fg DarkTurquoise -e
/tmp/banner_cmd &
```

The **aixterm** command is NOT an X Toolkit based application. Because of this, the **aixterm** command gets resource files as follows:

- **System defaults** from the first of these it finds:

```
$XFILESEARCHPATH %T=app-defaults %N=Xdefaults %L=$LANG
$XFILESEARCHPATH %T=app-defaults %N=Xdefaults %L=
/usr/lpp/X11/defaults/$LANG/Xdefaults
/usr/lpp/X11/defaults/Xdefaults
/usr/lib/X11/$LANG/app-defaults/Xdefaults
/usr/lib/X11/app-defaults/Xdefaults
/usr/lpp/X11/defaults/app-defaults/Xdefaults
```

- **Application system defaults** from the first of these it finds:

```
$XFILESEARCHPATH %T=app-defaults %N=Aixterm %L=$LANG
$XFILESEARCHPATH %T=app-defaults %N=Aixterm %L=
$XFILESEARCHPATH %T=app-defaults %N=aixterm %L=$LANG
$XFILESEARCHPATH %T=app-defaults %N=aixterm %L=
/usr/lpp/X11/defaults/$LANG/Aixterm
/usr/lpp/X11/defaults/Aixterm
/usr/lib/X11/$LANG/app-defaults/Aixterm
/usr/lib/X11/app-defaults/Aixterm
/usr/lib/X11/defaults/app-defaults/Aixterm
/usr/lpp/X11/defaults/$LANG/aixterm
/usr/lpp/X11/defaults/aixterm
/usr/lib/X11/$LANG/app-defaults/aixterm
/usr/lib/X11/app-defaults/aixterm
/usr/lib/X11/defaults/app-defaults/aixterm
```

- **User application defaults** from the first of these it finds:

```
$XUSERFILESEARCHPATH %T=app-defaults %N=Aixterm %L=$LANG
$XUSERFILESEARCHPATH %T=app-defaults %N=Aixterm %L=
$XUSERFILESEARCHPATH %T=app-defaults %N=aixterm %L=$LANG
$XUSERFILESEARCHPATH %T=app-defaults %N=aixterm %L=
$XAPPLRESDIR/$LANG/Aixterm
$XAPPLRESDIR/Aixterm
$XAPPLRESDIR/$LANG/aixterm
$XAPPLRESDIR/aixterm
$HOME/$LANG/Aixterm
$HOME/Aixterm
$HOME/$LANG/aixterm
```

- **User defaults** from the first of these it finds:

```
dpy->xdefaults          (A.K.A. "RESOURCE_MANAGER" property)
$HOME/$LANG/.Xdefaults
$HOME/.Xdefaults
```

- **Host defaults** from the first of these it finds:

```
$XENVIRONMENT
$HOME/$LANG/.Xdefaults-hostname
$HOME/.Xdefaults-hostname
```

> **Note:** XFILESEARCHPATH and XUSERFILESEARCHPATH support is limited to the %T, %N and %L substitution strings. Also, $LANG is actually whatever the result of the setlocale(LC_CTYPE,NULL) call is.

ali Command

Purpose

Lists mail aliases and their addresses.

Syntax

ali [**-alias** *File*] [**-list** | **-nolist**] [**-normalize** | **-nonormalize**] [**-user** *User* | **-nouser**] [*Alias ...*]

Description

The **ali** command lists mail aliases and their addresses. By default, this command searches the **/etc/mh/MailAliases** file and writes to standard output each alias and its address defined in the file. To specify an alternate mail aliases file, use the **-alias** *File* flag.

If you specify the **-user** flag, the **ali** command searches the alias files for the user name and writes to standard output the aliases that contain this user name.

Flags

Item	Description
-alias *File*	Specifies the mail alias file to be searched. The default is the **/etc/mh/MailAliases** file.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-list	Displays each address on a separate line.
-nolist	Displays addresses on as few lines as possible. This flag is the default.
-nonormalize	Prevents conversion of local host nicknames to official host names. This is the default.
-normalize	Converts local host nicknames to their official host names.
-nouser	Lists the address for an alias. This flag is the default.
-user *User*	Lists the aliases that contain the specified user. When the **-user** and **-nonormalize** flags are used together, the result may be a partial list of aliases that contain the specified user.

Examples

1. To display a list of all aliases and their addresses in the **/etc/mh/MailAliases** file, enter:

   ```
   ali
   ```

2. To list the names and addresses of the `mygroup` alias, enter:

   ```
   ali   mygroup
   ```

   A list similar to the following is displayed on your local system:

   ```
   mike@mercury   george@helium   vicky@venus
   ```

Files

Item	Description
$HOME/.mh_profile	Contains the MH user profile.
/etc/group	Contains a list of groups.
/etc/passwd	Contains a list of users.
/etc/mh/MailAliases	Contains the default mail alias file.

Item	Description
/usr/bin/ali	Contains the **ali** command.

alias Command

Purpose

Defines or displays aliases.

Syntax

alias [**-t**] [**-x**] [*AliasName* [=*String*]] ...

Description

The **alias** command creates or redefines alias definitions or writes existing alias definitions to standard output.

If no flags or parameters are supplied, all existing alias definitions are written to standard output. You can display a specific alias definition by using the *AliasName* parameter.

Create a new alias by using the *AliasName*=*String* parameter pair. When the shell encounters an alias on the command line or in a shell script, it substitutes the definition supplied by the string. The *String* variable can contain any valid shell text. Enclose the value of the *String* variable in single quotes if the string contains spaces. If the *AliasName* parameter is not a valid name, the **alias** command displays an error message.

If you specify the **-t** flag, the shell displays aliases that are *tracked*. A tracked command uses the full path name of the command. A tracked command can become undefined when the value of the **PATH** environment variable is reset, but aliases created with the **-t** flag remain tracked.

If you specify the **-x** flag, the shell displays aliases that are *exported*. An exported alias is active in all shells.

An alias definition affects the current shell environment and the execution environments of any subshells. The alias definition affects neither the parent process of the current shell nor any utility environment invoked by the shell.

Flags

Item	Description
-t	Sets or displays all existing tracked aliases. If this flag is used with the *AliasName* parameter, the new alias is tracked and the alias definition includes the full path name obtained by doing a path search. When the value of the **PATH** environment variable is reset, the alias definition becomes undefined but remains tracked.
-x	Displays all existing exported alias definitions. If this flag is used with the *AliasName* parameter, the new alias is exported. Exported alias are not defined across separate invocations of the shell. You must put alias definitions in your environment file to have aliases defined for separate shell invocations.

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	One of the specified alias name did not have an alias definition, or an error occurred.

Examples

1. To change the **ls** command so that it displays information in columns and annotates the output, enter:

```
alias ls='ls -CF'
```

2. To create a command for repeating previous entries in the command history file, enter:

```
alias r='fc -s'
```

3. To use 1KB units for the **du** command, enter:

```
alias du=du\ -k
```

4. To create a command to display all active processes for user Dee, enter:

```
alias psc='ps -ef | grep Dee'
```

5. To see the full path name of the **ls** command, enter:

```
alias -t ls
```

   The screen displays `ls=/usr/bin/ls`.

Files

Item	Description
/usr/bin/ksh	Contains the Korn shell **alias** built-in command.
/usr/bin/alias	Contains the **alias** command.

alog Command

Purpose

Creates and maintains fixed-size log files created from standard input.

Syntax

To Show the Contents of a Log File

alog -f *LogFile* [**-o**] **To Log Data to a Specified Log File**

alog -f *LogFile* | [[**-q**] [**-s** *Size*]]

To Display the Verbosity Value of a Specified Log Type

alog -t *LogType* **-V**

To Change the Attributes of a Specified Log Type

alog -C -t *LogType* [**-f** *LogFile*] [**-s** *Size*] [**-w** *Verbosity*]

To Display the Current Attributes of a Specified Log Type

alog -L [**-t** *LogType*]

To Display the Usage of the alog Command

alog -H

Description

The **alog** command reads standard input, writes to standard output, and copies the output into a fixed-size file. This file is treated as a circular log. If the file is full, new entries are written over the oldest existing entries.

The **alog** command works with log files that are specified on the command line or with logs that are defined in the alog configuration database. Logs that are defined in the alog configuration database are identified by *LogType*. The **File**, **Size**, and **Verbosity** attributes for each defined *LogType* are stored in the alog configuration database with the *LogType*. You can add a new *LogType* to the alog configuration database using the **odmadd** command. You can change the attributes of *LogType* defined in the alog configuration database using the **alog** command.

Flags

Item	Description
-C	Changes the attributes for a specified *LogType*. Use the **-C** flag with the **-f**, **-s**, and **-w** flags to change the **File**, **Size**, and **Verbosity** attributes for the specified *LogType*. The **-t** *LogType* flag is required. **Note:** Using the `-C` flag with `-s`*Size* only changes the size value in ODM and does not change the size of the actual log file. If the **-C** flag is used, the **alog** command does not copy standard input to standard output or to a log file. When the **-C** flag is used to modify the attributes for the console log type, the console log file is also modified and the console device driver is updated to use the new values. This is a deviation from the normal operation of **alog -C** and is done to accommodate special formatting in the console log file. **Note:** You must have root user authority to change **alog** attributes.
-f *LogFile*	Specifies the name of a log file. If the specified log file does not exist, one is created. If the **alog** command is unable to write to a log file, it writes to **/dev/null**. Use the **-f** *LogFile* flag with the **-C** and **-t** flags to change the **File** attribute for a *LogType* defined in the alog configuration database.
-H	Displays the usage of the **alog** command.
-L	Lists the log types currently defined in the alog configuration database. If you use the **-L** flag with the **-t** *LogType* flag, the attributes for a specified *LogType* are listed. The current values of the **File**, **Size**, and **Verbosity** attributes are listed as colon separated values: `<File>:<Size>:<Verbosity>` If the **-L** flag is used, the **alog** command does not copy standard input to standard output or to **File**.
-o	Lists the contents of the log file. Writes the contents of the log file to standard output in sequential order.
-q	Copies standard input to a log file but does not write to standard output.

Item	Description
-s *Size*	Specifies the size limit of the log file in bytes. The space for the log file is reserved when it is created. If you create a new log file and do not specify the **Size** attribute, the minimum size, 4096 bytes, is used. If the log file already exists, its size will be changed. The size you specify is rounded upward to the next integral multiple of 4096 bytes. The maximum size for a log file is 2 GB. If the specified size is greater than 2 GB, only 2 GB is considered. If you decrease the size of the log file, the oldest entries in the log are deleted if they do not fit within the new size limit. You must have write permission for the log file to change its size. Use the **-s** *Size* flag with the **-C** and the **-t** flags to change the **Size** attribute for *LogType* defined in the alog configuration database. Only the size value in ODM is changed. The size of the actual log file remains the same. The new **Size** attribute value is used the next time a log file is created.
-t *LogType*	Identifies a log defined in the alog configuration database. The **alog** command gets the log's file name and size from the alog configuration database. If *LogFile* does not exist, one is created. If the **alog** command cannot get the information for the specified *LogType* from the alog configuration database or if the **alog** command is unable to write to *LogFile*, it writes to **/dev/null**. If you specify *LogType* and *LogFile* using the **-f** flag, *LogFile* is used and *LogType* is ignored.
-V	Writes the current value of the **Verbosity** attribute for *LogType* that is defined in the alog configuration database to standard output. If you do not specify *LogType*, or the *LogType* you specify is not defined, nothing is written to standard output. The value output using the **alog** command with the **-t** *LogType* and the **-V** flags can be used by a command that is piping its output to the **alog** command to control the verbosity of the data it writes to the pipe.
-w *Verbosity*	Changes the **Verbosity** attribute for *LogType* defined in the alog configuration database when used with the **-C** and the **-t** flags. The **Verbosity** attribute can have a value from 0 to 9. If the value is 0, no information is copied to *LogFile* by the **alog** command. All of the information is still written to standard output. If the value is not 0, all of the information piped to the **alog** command's standard input is copied to *LogFile* and to standard output.

Examples

1. To record the current date and time in a log file named `sample.log`, enter:

```
date | alog -f /tmp/sample.log
```

2. To list the contents of `/tmp/sample.log` log file, enter:

```
alog -f /tmp/sample.log -o
```

3. To change the size of the log file named `/tmp/sample.log` to 8192 bytes, enter:

```
echo "resizing log file" | alog -f /tmp/sample.log -s 8192
```

4. To add a new log type `sample` to the alog configuration database, create the `alog.add` file in the following format:

```
SWservAt:
    attribute="alog_type"
    deflt="sample"
```

```
    value="sample"
```

```
SWservAt:
    attribute="sample_logname"
    deflt="/tmp/sample.log"
    value="/tmp/sample.log"
```

```
SWservAt:
    attribute="sample_logsize"
    deflt="4096"
    value="4096"
```

```
SWservAt:
    attribute="sample_logverb"
    deflt="1"
    value="1"
```

After creating the `alog.add` file, enter:

```
odmadd alog.add
```

This adds the `alog.add` file to the `SWservAt` database.

5. To change the name of the log file for the log type `sample` to `/var/sample.log` in the alog configuration database, enter:

```
alog -C -t sample -f /var/sample.log
```

6. To change the size of the boot log to 8192 bytes and reflect the new size in ODM, enter:

```
alog -C -t boot -s 8192
echo "Changed log size" | alog -t boot -s 8192
```

Files

Item	Description
/etc/objrepos/SWservAt	Software Service Aids Attributes Object Class

alstat Command

Purpose

Shows alignment exception statistics.

Syntax

alstat [**-e** | **-v**] [*Interval*] [*Count*]

Description

The **alstat** command displays alignment exception statistics. Alignment exceptions may occur when the processor cannot perform a memory access due to an unsupported memory alignment offset (such as a floating point double load from an address that is not a multiple of 8). However, some types of unaligned memory references may be corrected by some processors and does not generate an alignment exception.

The alignment exception count since the last time the machine was rebooted and the count in the current interval are displayed. You can optionally display emulation exception statistics or individual processor alignment statistics.

The default output displays statistics every second. The sampling *Interval* and *Count* of iterations can be also specified.

Parameters

Item	Description
Interval	Interval between samples.
Count	Number of iterations.

Flags

Item	Description
-e	Displays emulation exception statistics. This flag cannot be used with the **-v** flag.
-v	Display individual processor statistics. This flag cannot be used with the **-e** flag.

Examples

1. To display alignment exception statistics every second, type:

```
alstat
```

This produces the following output:

```
  Alignment  Alignment
  SinceBoot      Delta
    8845591          0
    8845591          0
    8845591          0
    8845591          0
    8845591          0
    8845591          0
  ...
```

2. To display emulation and alignment exception statistics every two seconds, a total of 5 times, type:

```
alstat -e 2 5
```

This produces the following output:

```
  Emulation  Emulation  Alignment  Alignment
  SinceBoot      Delta  SinceBoot      Delta
   21260604          0   70091846          0
   23423104    2162500   72193861    2102015
   25609796    2186692   74292759    2098898
   27772897    2163101   76392234    2099475
   29958509    2185612   78490284    2098050
```

3. To display alignment exception statistics, every 5 seconds, for each processor, type:

```
alstat -v 5
```

This produces the following output:

```
  Alignment  Alignment  Alignment  Alignment
  SinceBoot      Delta    Delta00    Delta01
   88406295          0          0          0
   93697825    5291530          0    5291530
   98930330    5232505    5232505          0
```

```
  102595591    3665261     232697    3432564
  102595591          0          0          0
```

alt_disk_copy Command

Purpose

Clones (makes a copy of) the currently running system to an alternate disk.

Syntax

To copy rootvg to an alternate disk:

alt_disk_copy -d *targetdisks...* [-i **image.data**] [-s *script*] [-b *bundlename*] [-I *installpflags*] [-l *imageslocation*] [-f *fixbundle*] [-F *fixes*] [-e *excludelist*] [-w *filesets*] [-n] [-P *phases*] [-c *console*] [-x *first_boot_script*] [-R *resolvconf*] [-DBOVgruTS]

Description

The **alt_disk_copy** command allows users to copy the current rootvg to an alternate disk and to update the operating system to the next maintenance or technology level, without taking the machine down for an extended period of time and mitigating outage risk. This can be done by creating a copy of the current rootvg on an alternate disk and simultaneously applying software updates. If needed, the **bootlist** command can be run after the new disk has been booted, and the bootlist can be changed to boot back to the older maintenance or technology level of the operating system.

Cloning the running rootvg, allows the user to create a backup copy of the root volume group. This copy can be used as a back up in case the rootvg failed, or it can be modified by installing additional updates. One scenario might be to clone a 5300-00 system, and then install updates to bring the cloned rootvg to 5300-01. This would update the system while it was still running. Rebooting from the new rootvg would bring the level of the running system to 5300-01. If there was a problem with this level, changing the bootlist back to the 5300-00 disk and rebooting would bring the system back to 5300-00. Other scenarios would include cloning the rootvg and applying individual fixes, rebooting the system and testing those fixes, and rebooting back to the original rootvg if there was a problem.

At the end of the install, a volume group, altinst_rootvg, is left on the target disks in the varied off state as a place holder. If varied on, it indicates that it owns no logical volumes; however, the volume group does contain logical volumes, but they have been removed from the ODM because their names now conflict with the names of the logical volumes on the running system. Do not vary on the altinst_rootvg volume group; instead, leave the definition there as a placeholder.

After rebooting from the new alternate disk, the former rootvg volume group shows up in a lspv listing as `old_rootvg`, and it includes all disks in the original rootvg. This former rootvg volume group is set to not vary-on at reboot, and it should only be removed with the **alt_rootvg_op -X old_rootvg** or **alt_disk_install -X old_rootvg** commands.

If a return to the original rootvg is necessary, the **bootlist** command is used to change the bootlist to reboot from the original rootvg.

Notes:

1. Alternate disk operations create volume groups, logical volumes, special device files, and file systems using the **alt** prefix. If **alt_disk_copy** is utilized on a system, the administrator should avoid having or creating volume groups, logical volumes, special device files, or file systems with the **alt**, prefix—alternate disk operations might inadvertently remove, alter, or damage these items.
2. NIM alternate disk migration (upgrading version or release levels) is supported with the **nimadm** command. Please see the **nimadm** documentation for more details.

3. The current LVM limit for logical volume names is 15 characters. Because the alternate disk installation commands prepend the 4-character **alt_** prefix, the limit for the original logical volume names in the rootvg to be copied or installed is 11 characters. If an original logical volume name exceeds 11 characters, it can be shortened by using a customized **image.data** (see the **-i** flag).
4. When cloning the rootvg volume group, a new boot image is created with the **bosboot** command. If the **/dev/ipldevice** is removed or altered then the **bosboot** command will fail.
5. Do not use direct LVM commands (such as **exportvg**, **importvg**, **varyoffvg**, or **chlv**) on alternate rootvg volume groups.
6. This function is also available with the Network Installation Management (NIM). See the NIM Guide for more information.
7. The **alt_disk_copy** command only backs up mounted file systems. Mount all file systems that you want to back up. The **mksysb** command backs up mounted journaled file systems (JFS) and enhanced journaled file systems (JFS2) in the rootvg. For more information about backing up file systems, see the **mount** command.
8. To avoid back up errors, the system activity must be quiesced during the backup of the system. If backup or restore errors happen when you are running the **alt_disk_copy** command, messages are printed, but the command continues, and if no other issues, the command returns 0. This behavior can be controlled by using the **ALT_BAK_ERR_FAIL** and **ALT_BAK_ERR_FAIL** environment variables. If the **ALT_BAK_ERR_FAIL** environment variable is set to 1 and if an error occurs during a backup or restore operation, the **alt_disk_copy** command runs cleanup operation and stops running. If the **ALT_BAK_ERR_REPORT** environment variable is set to 1 and if an error occurs during a backup or restore operation, the **alt_disk_copy** command continues to run but the return code is set to 1 and the **bootlist** is not set to boot from the alternate disk.
9. If you are using the **alt_disk_copy** command to upgrade a system and the current level of the rootvg is prior to 6100-08 SP2 or 7100-02 SP2, install the **bos.alt_disk_install.rte** fileset at the level you are doing the upgrade to, on the original rootvg, before the **alt_disk_copy** operation. If you do not install the **bos.alt_disk_install.rte** fileset, error messages are displayed while creating the boot image in the alternate rootvg.
10. After an **alt_disk_copy** operation following a **tcbck -n ALL** command, the TCB-enabled system might encounter the following error:

    ```
    error: 3001-020 The file /dev/altinst_rootvg was not found.
    ```

    The **altinst_rootvg** entry in the TCB database can be removed by running the `# tcbck -d /dev/altinst_rootvg` command.
11. After booting the system to an alternate disk, Network File System (NFS) clients might receive ESTALE errors when the clients access NFS directories from the copied system. These clients must unmount and remount the affected directories.

Flags

Item	Description
-b *bundlename*	Path name of optional file with a list of packages or filesets that are installed after a rootvg clone. The **-l** flag must be used with this option.
-B	Would specify not running bootlist after the **mksysb** or clone. If set, then the **-r** flag cannot be used.
-c *console*	The device name to be used as the alternate rootvg's system console. This option is only valid with the **-O** flag.
-d *targetdisks*	Specifies a space-delimited list of the name or names of the target disks where the alternate rootvg will be created. However, when specifying multiple disks, the list must be enclosed in quotes (**" "**). These disks must not currently contain any volume group definition. The **lspv** command should show these disks as belonging to volume group None.

Item	Description
-D	Turns on debug (sets **-x** output).
-e *excludelist*	Optional *exclude.list* to use when cloning rootvg. The rules for exclusion follow the pattern-matching rules of the grep command. The *excludelist* must be a full path name. **Note:** If you want to exclude certain files from the backup, create the **/etc/exclude.rootvg** file with an ASCII editor and enter the patterns of file names that you do not want included in your system backup image. The patterns in this file are input to the pattern-matching conventions of the **grep** command to determine which files will be excluded from the backup. If you want to exclude files listed in the **/etc/exclude.rootvg** file, select the **Exclude Files** field and press the Tab key once to change the default value to yes. For example, to exclude all the contents of the **scratch** directory, edit the exclude file to read as follows: `/scratch/` For example, to exclude the contents of the **/tmp** directory, and avoid excluding any other directories that have **/tmp** in the path name, edit the exclude file to read as follows: `^./tmp/` All files are backed up relative to **.** (current working directory). To exclude any file or directory for which it is important to have the search match the string at the beginning of the line, use the caret character (^) as the first character in the search string, followed by the dot character (.), followed by the filename or directory to be excluded. If the filename or directory being excluded is a substring of another filename or directory, use the caret character followed by the dot character (^.) to indicate that the search should start at the beginning of the line, and use the dollar sign character ($) to indicate that the search should end at the end of the line.
-f *fixbundle*	Optional file with a list of APARs to install after a clone of rootvg. The **-l** flag must be used with this option.
-F *fixes*	Optional list of APARs (for example, IX123456) to install after a clone of rootvg. The **-l** flag must be used with this option.
-g	Skips disk bootability checks.
-i *image.data*	Optional **image.data** file to use instead of the default **image.data** file created from **rootvg**. The **image.data** file name must be a full path name (such as **/tmp/my_image.data**).
-I *installpflags*	The flags to use when updating or installing new filesets into the cloned **altinst_rootvg**. The default flag is **-acgX**. The **-l** flag must be used with this option.
-l *imageslocation*	Location of **installp** images or updates to apply after a clone of rootvg. This can be a directory full path name or device name (such as **/dev/rmt0**).
-n	Remain NIM client. The **/.rhosts** and **/etc/niminfo** files are copied to the file system of the alternate rootvg.
-O	Performs a device reset on the target **altinst_rootvg**. This causes the alternate disk install to not retain any user-defined device configurations. This flag is useful if the target disk or disks become the rootvg of a different system (such as in the case of logical partitioning or system disk swap).

Item	Description
-P *phases*	The phase or phases to execute during this invocation of **alt_disk_copy**. Valid values are: 1, 2, 3, 12, 23, or all (default). **12** Performs phases 1 and 2. **23** Performs phases 2 and 3. **all** Performs all three phases.
-r	Specifies to reboot from the alternate disk when the `alt_disk_copy` command finishes.
-R *resolvconf*	The **resolv.conf** file to replace the existing one after the rootvg has been cloned. You must specify a full path name.
-s *script*	Optional customization script to run at the end of the **mksysb** install or the rootvg clone. This file must be executable. This script is called on the running system before the **/alt_inst** file systems are unmounted, so files can be copied from the running system to the **/alt_inst** file systems before the reboot.
-S	Indicates that you want to skip space-checking on target disks before you start performing the cloning or installation operations. **Important :** JFS2 file systems contain more metadata than JFS file systems. When you use the -S flag in conjunction with the -T flag, it skips space-checking. In this situation, it does not verify that there is enough space in the newly created JFS2 file system to store the contents of the file system plus the additional metadata.
-T	Indicates that you want to convert JFS file systems to JFS2 file systems during the process of recreating the rootvg volume group on target disks.
-u	Copies file systems that belong to a workload partition (WPAR) in the defined state in the alternate system. **Note:** To be included in the alternate disk, all file systems that belong to a WPAR in the defined state need to be in the rootvg volume group.
-V	Turn on verbose output. This shows the files that are being backed up for rootvg clones.
-w *filesets*	List of filesets to install after cloning a rootvg. The **-l** flag must be used with this option.
-x *first_boot_script*	Optional customization script to run during the initial boot of the alternate rootvg, after all file systems are mounted.

Exit Status

Item	Description
0	All **alt_disk_copy** related operations completed successfully.
>0	An error occurred.

Examples

1. To clone the running 5300-00 rootvg to **hdisk3**, then apply updates from **/updates** to bring the cloned rootvg to a 5300-01 level:

   ```
   alt_disk_copy -d hdisk3 -F 5300-01_AIX_ML -l /updates
   ```

   The bootlist would then be set to boot from **hdisk3** at the next reboot.
2. To clone the running rootvg to **hdisk3** and **hdisk4**, and execute **update_all** on all updates from **/updates**:

   ```
   alt_disk_copy -d "hdisk3 hdisk4" -b update_all -l /updates
   ```

   The bootlist would then be set to boot from **hdisk3** at the next reboot.
3. To clone the running rootvg to **hdisk1** and stop after phase 1:

   ```
   alt_disk_copy -d hdisk1 -P1
   ```

   Attention : Do not change the bootlist to use the cloned **rootvg**.
4. To execute phases 2 and 3 on an existing alternate rootvg and reboot the system on successful completion:

   ```
   alt_disk_copy -d hdisk1 -P23 -r
   ```
5. To clone the running system to **hdisk1** and **hdisk2**, and to convert the file systems from JFS file systems to JFS2 file systems, run the following command:

   ```
   alt_disk_copy -B -T -d "hdisk1 hdisk2"
   ```

Location

/usr/sbin/alt_disk_copy

Files

Item	Description
/usr/sbin/alt_disk_copy	Contains the **alt_disk_copy** command.

alt_disk_install Command

Purpose

Installs an alternate disk with a **mksysb** install image or clones the currently running system to an alternate disk. This command is obsolete in AIX 5.3.

Note: In AIX 5.3, the **alt_disk_install** command is replaced by the **alt_disk_copy**, **alt_disk_mksysb**, and **alt_rootvg_op** commands. The **alt_disk_install** module continues to be shipped as a wrapper to the new commands, but the **alt_disk_install** command does not support any new functions, flags, or features.

Syntax

" Create Alternate Disk: "
alt_disk_install { **-d** *device* | **-C**} [**-i** *image.data*] [**-s** *script*] [**-R** *resolv_conf*] [**-D**] [**-B**] [**-V**] [**-r**] [**-O**]

[**-p** *platform*] [**-L** *mksysb_level*]

[**-b** *bundle_name*] [**-I** *installp_flags*]

[**-l** *images_location*] [**-f** *fix_bundle*]

[**-F** *fixes*] [**-e** *exclude_list*] [**-w** *filesets*]

[**-n**] [**-P** *phase_option*] *target_disks...*

"Clean Up Alternate Disk Volume Group:"
alt_disk_install -X

For alt_disk_install or later:

" Determine Volume Group Boot Disk:"
alt_disk_install -q *disk*

"Put-to-sleep Volume Group:"
alt_disk_install -S

"Rename Alternate Disk Volume Group:"
alt_disk_install -v *new_volume_group_name disk*

"Wake-up Volume Group:"
alt_disk_install -W *disk*

"Clean Up Alternate Disk Volume Group:"
alt_disk_install -X [*volume_group*]

Description

Note: In AIX 5.3 the **alt_disk_install** command has been broken up into three commands: **alt_disk_copy**, **alt_disk_mksysb**, and **alt_rootvg_op**. No new functionality will be added to this command.

The **alt_disk_install** command allows users a way to update the operating system to the next release, maintenance level, or technology level, without taking the machine down for an extended period of time. This can be done in two ways, by installing a mksysb image on a separate disk, or by cloning the current system and then applying updates to get to the next maintenance or technology level.

Attention: alt_disk_install creates volume groups, logical volumes, special device files, and file systems using the "alt" prefix. If **alt_disk_install** is utilized on a system, the administrator should avoid having or creating volume groups, logical volumes, special device files, or file systems with the "alt" prefix - **alt_disk_install** operations may inadvertently remove, alter, or damage these items.

The first function, installing a mksysb, requires an AIX 4.3 or later mksysb image, an AIX 4.3 or later mksysb tape, or an AIX 4.3.3 or later mksysb CD. The **alt_disk_install** command is called with a disk or disks that are not currently in use, and the mksysb is restored to those disks such that, if the user chooses, the next reboot boots the system on an AIX 4.3 or later system.

Note:

1. You cannot use **alt_disk_install** to install an earlier version of AIX than the one currently installed on the system. For example, you cannot install an AIX 4.3 mksysb on an AIX 5.1 system.
2. If needed, the **bootlist** command can be run after the new disk has been booted, and the bootlist can be changed to boot back to the older version of the operating system.

The second function, cloning the running rootvg, allows the user to create a backup copy of the root volume group. This copy could be used as a back up in case the rootvg failed, or it could be modified by installing additional updates. One scenario might be to clone a 4.2.0 system, then install updates to bring the cloned rootvg to 4.2.1.0. This would update the system while it was still running, then rebooting from the new rootvg would bring the level of the running system to 4.2.1. If there was a problem with this level, changing the bootlist back to the 4.2.0 disk and rebooting would bring the system back to 4.2.0. Other

scenarios would include cloning the rootvg and applying individual fixes, rebooting the system and testing those fixes, and rebooting back to the original rootvg if there was a problem.

Note: NIM alternate disk migration (upgrading version or release levels) is supported with the **nimadm** command in AIX 5.1 and later. Please see the **nimadm** documentation for more details.

Currently, you can run the **alt_disk_install** command on 4.1.4.0 and higher systems for both of these functions. The **bos.alt_disk_install.rte** fileset must be installed on the system to execute the **alt_disk_install** command, and the **bos.alt_disk_install.boot_images** fileset must also be installed to perform a mksysb install to an alternate disk.

The mksysb image that is used must be created ahead of time and have all the necessary device and kernel support required for the system that it's going to be installed on. No new device or kernel support can be installed before the system is rebooted from the newly installed disk.

Note: The version release maintenance or technology level of mksysb that you are installing must match the level of the **bos.alt_disk_install.boot_images** fileset.

When cloning the rootvg volume group, a new boot image is created with the **bosboot** command. When installing a mksysb image, a boot image for the level of mksysb and platform type is copied to the boot logical volume for the new alternate rootvg. When the system is rebooted, the **bosboot** command is run in the early stage of boot, and the system is rebooted once again. This is to synchronize the boot image with the mksysb that was just restored. The system then boots in normal mode.

At the end of the install, a volume group, altinst_rootvg, is left on the target disks in the varied off state as a place holder. If varied on, it shows as owning no logical volumes, but it does indeed contain logical volumes, but they have been removed from the ODM because their names now conflict with the names of the logical volumes on the running system. It is recommended that you not vary on the altinst_rootvg volume group, but just leave the definition there as a place holder.

After the system reboots from the new alternate disk, the former rootvg volume group does not show up in a **lspv** listing, unless the **alt_disk_install** version is 4.3.2 or higher.

For **alt_disk_install** 4.3.2 or greater:

> After rebooting from the new alternate disk, the former rootvg volume group shows up in a **lspv** listing as "old_rootvg", and includes all disk(s) in the original rootvg. This former rootvg volume group is set to NOT varyon at reboot, and should ONLY be removed with the -X flag (i.e. **alt_disk_install** -X old_rootvg).

If a return to the original rootvg is necessary, the bootlist command is used to change the bootlist to reboot from the original rootvg.

For **alt_disk_install** 4.3.2 or greater:

> If it is unclear which disk is the boot disk for a specific volume group, the -q flag can be used to determine the boot disk. This can be useful when a volume group is comprised of multiple disks and a change in the bootlist is necessary.

The alternate root file system is mounted as **/alt_inst**, so other file systems would have that prefix (**/alt_inst/usr**, **/alt_inst/var**). This is how they should be accessed if using a customization script.

> **Attention:** If you have created an alternate rootvg with **alt_disk_install**, but no longer wish to use it, or want to run **alt_disk_install** commands, do not run exportvg on altinst_rootvg.
>
> Simply run the **alt_disk_install -X** command to remove the altinst_rootvg definition from the ODM database. The reason you cannot run the **exportvg** command (or the **reducevg** command) is that the logical volume names and file systems now have the real names, and **exportvg** removes the stanza's for the real file system from **/etc/filesystems** for the real rootvg.
>
> If **exportvg** is run by accident, be sure to recreate the **/etc/filesystems** file before rebooting the system. The system will not reboot without a correct **/etc/filesystems** file.

This function is also available with the Network Installation Management (NIM). See the NIM Guide for more information.

The AIX 4.3.1 and greater version of **alt_disk_install** can be executed in phases. The install is divided into three phases, and the default is to perform all three phases.

Item	Description
Phase 1	Creates the **altinst_rootvg** volume group, the **alt_** "logical volumes", the **/alt_inst** file systems, and restores the mksysb or rootvg data.
Phase 2	Runs any specified customization script, installs updates, new filesets, fixes or bundles (cloning only), copies a **resolv.conf** file if specified, and copies files over to remain a NIM client if specified.
Phase 3	Unmounts the **/alt_inst** file systems, renames the file systems and logical volumes, removes the **alt_ logical** volumes, names ODM and varies off the altinst_rootvg. It sets the bootlist and reboots if specified.

You can run each phase separately, run Phases 1 and 2 together, or run Phases 2 and 3 together. Phase 2 can be run multiple times before Phase 3 is run.

You must run Phase 3 to get a volume group that is a usable rootvg. Running Phase 1 and 2 leave the /alt_inst file systems mounted.

If you have run Phase 1 and or Phase 2, and want to start over (remove the altinst_rootvg), run the **alt_disk_install -X** command to clean up.

For **alt_disk_install** 4.3.2 or greater:

If data access is necessary between the original rootvg and the new alternate disk, a volume group "wake-up" can be accomplished, using the -W flag, on the non-booted volume group. The "wake-up" puts the volume group in a post **alt_disk_install** phase 1 state (i.e. the /alt_inst file systems will be mounted).

Note: The volume group that experiences the "wake-up" will be renamed "altinst_rootvg".

Limitation

The running system's version of operating system must be greater than or equal to the operating system version of the volume group that undergoes the "wake-up". This may mean that it's necessary to boot from the "altinst_rootvg" and "wake-up" the "old_rootvg".

For example: An alternate disk is created from an **alt_disk_install** 4.3.3 mksysb, on a 4.1.5 running system. To access data between the two volume groups, it is necessary to boot from the 4.3.3 alternate disk and "wake-up" the 4.1.5 "old_rootvg" volume group.

This limitation is caused by a jfs log entry incompatibility. It is possible to "wake-up" a volume group that contains a greater operating system version, but the volume group could not have ever been the system rootvg. If so, the volume group would have made jfs log entries that could not be interpreted by an older operating system version rootvg, when the volume group was experiencing a "wake-up". JFS log entries are usually present for file systems that were not unmounted before a reboot, for example, /,/usr.

The **alt_disk_install** command will not allow a "wake-up" to occur on a volume group with a greater operating system version, unless the FORCE environment variable is set to "yes".

Attention: If a FORCE "wake-up" is attempted on a volume group that contains a greater operating system version then the running operating system, **AND** the "waking" volume group has been a system rootvg, errors will occur.

When data access is no longer needed, the volume group can be put to sleep, using the -S flag.

Note: The volume group that has experienced a "wake-up" MUST be "put-to-sleep" before it can be booted and used as the rootvg.

Flags

Item	Description
-B	Would specify not running bootlist after the mksysb or clone. If set, the **-r** flag cannot be used. **Note:** The **-B** and **-X** flags are mutually exclusive.
-C	Clone rootvg. **Note: -d** and **-C** are mutually exclusive.
-d *device*	The value for *device* can be: `tape device - for example, /dev/rmt0` OR `path name of mksysb image in a file system.` **Note: -d** and **-C** are mutually exclusive.
-D	Turns on debug (set **-x** output).
-i *image.data*	Optional image.data file to use instead of default image.data from mksysb image or image.data created from rootvg. The image.data file name must be a full pathname, for example, `/tmp/my_image.data`. For **alt_disk_install** 4.3.2 or greater: If certain logical volumes need to be placed on a specific target disk, this should be annotated in the logical volume LV_SOURCE_DISK_LIST field of the user specified image.data file.
-p *platform*	This is a platform to use to create the name of the disk boot image, which may be supplied by a vendor that wanted to support this function. This flag is only valid for mksysb installs (**-d** flag).
-P*phase*	The *phase* to execute during this invocation of **alt_disk_install**. Valid values are: 1, 2, 3, 12, 23, or all. • 12 - performs phases 1 and 2. • 23 - performs phases 2 and 3. • all - performs all three phases
-r	Would specify to reboot from the new disk when the **alt_disk_install** command is complete.
-R *resolv_conf*	The **resolv.conf** file to replace the existing one after the mksysb has been restored or the rootvg has been cloned. You must use a full pathname for *resolv_conf*.
-s *script*	Optional customization script to run at the end of the mksysb install or the rootvg clone. This file must be executable. This script is called on the running system before the /alt_inst file systems are unmounted, so files can be copied from the running system to the **/alt_inst** file systems before the reboot. This is the only opportunity to copy or modify files in the alternate file system because the logical volume names will be changed to match rootvg's, and they will not be accessible until the system is rebooted with the new alternate rootvg, or a "wake-up" is performed on the altinst_rootvg. You must use a full pathname for *script*.

Item	Description
-V	Turn on verbose output. This shows the files that are being backed up for rootvg clones. This flag shows files that are restored for mksysb alt_disk_installs.
-L *mksysb_level*	This level will be combined with the platform type to create the boot image name to use (for example, rspc_4.3.0_boot in AIX 5.1 and earlier). This must be in the form V.R.M. The mksysb image will be checked against this level to verify that they are the same.
-n	Remain NIM client. The **/.rhosts** and **/etc/niminfo** files are copied to the alternate rootvg's file system.
-X	Removes the altinst_rootvg volume group definition from the ODM database. This returns the **lspv** listing for the volume group to "None". This will not remove actual data from the volume group. Therefore, you can still reboot from that volume group, if you reset your bootlist. For **alt_disk_install** 4.3.2 or greater, the flag allows for specified volume group name ODM database definition removal, for example, `-X old_rootvg`. **Note:** 1. The **-B** and **-X** flags are mutually exclusive. 2. If you specify the **-X** flag, all other flags are ignored.
-O	Performs a device reset on the target altinst_rootvg. This will cause **alt_disk_install** to NOT retain any user defined device configurations. This flag is useful if the target disk or disks will become the rootvg of a different system (such as in the case of logical partitioning or system disk swap).

The following flags are only valid for use when cloning the rootvg (**-C**).

Item	Description
-b *bundle_name*	Pathname of optional file with a list of packages or filesets that will be installed after a rootvg clone. The **-l** flag must be used with this option.

Item	Description
-e *exclude_list*	Optional exclude.list to use when cloning rootvg. The rules for exclusion follow the pattern matching rules of the **grep** command. The *exclude_list* must be a full pathname. **Note:** If you want to exclude certain files from the backup, create the **/etc/exclude.rootvg** file, with an ASCII editor, and enter the patterns of file names that you do not want included in your system backup image. The patterns in this file are input to the pattern matching conventions of the **grep** command to determine which files will be excluded from the backup. If you want to exclude files listed in the **/etc/exclude.rootvg** file, select the Exclude Files field and press the Tab key once to change the default value to yes. For example, to exclude all the contents of the directory called scratch, edit the exclude file to read as follows: `/scratch/` For example, to exclude the contents of the directory called **/tmp**, and avoid excluding any other directories that have **/tmp** in the pathname, edit the exclude file to read as follows: `^./tmp/` All files are backed up relative to . (current working directory). To exclude any file or directory for which the it is important to have the search match the string at the beginning of the line, use ^ (caret character) as the first character in the search string, followed by **.** (dot character), followed by the filename or directory to be excluded. If the filename or directory being excluded is a substring of another filename or directory, use **^.** (caret character followed by dot character) to indicate that the search should begin at the beginning of the line and/or use **$** (dollar sign character) to indicate that the search should end at the end of the line.
-f *fix_bundle*	Optional file with a list of APARs to install after a clone of rootvg. The **-l** flag must be used with this option.
-F *fixes*	Optional list of APARs (for example, "IX123456") to install after a clone of rootvg. The **-l** flag must be used with this option.
-I *installp_flags*	The flags to use when updating or installing new filesets into the cloned alt_inst_rootvg. Default flags: "-acgX" The **-l** flag must be used with this option.
-l *images_location*	Location of installp images or updates to apply after a clone of rootvg. This can be a directory full pathname or device name (like **/dev/rmt0**).
-w *filesets*	List of filesets to install after cloning a rootvg. The **-l** flag must be used with this option.

The following flags are available for **alt_disk_install** version 4.3.2 or greater:

Item	Description
-q *disk*	Used to return the volume group boot disk name. This is especially useful when trying to determine the boot disk from several disks in the "old_rootvg" volume group, after rebooting from the alternate disk.
-S	Will "put-to-sleep" the volume group. This is used after a volume group "wake-up". (**-W**).
-v *new_volume_group_name disk*	Used to rename the alternate disk volume group. This is especially useful when creating multiple alternate disks, on multiple volume groups, and name identification is necessary.
-W *disk*	Used to "wake-up" a volume group for data access between the rootvg and the alternate disk rootvg. **Note**: The volume group that experiences the "wake-up" will be renamed "altinst_rootvg".

Limitation

The running system's version of the operating system must be greater than or equal to the operating system version of the volume group that undergoes the "wake-up". This may mean that it's necessary to boot from the "altinst_rootvg" and "wake-up" the "old_rootvg".

Parameters

Item	Description
target_disks	Specifies the name or names of the target disks where the alternate rootvg will be created. This disk or these disks must not currently contain any volume group definition. The **lspv** command should show these disks as belonging to volume group **None**.

Examples

1. To clone the running 4.2.0 rootvg to hdisk3, then apply updates from /updates to bring the cloned rootvg to a 4.2.1 level:

   ```
   alt_disk_install -C -F 4.2.1.0_AIX_ML -l /updates hdisk3
   ```

   The bootlist would then be set to boot from hdisk3 at the next reboot.
2. To install a 4.3 mksysb image on hdisk3, then run a customized script (/home/myscript) to copy some user files over to the alternate rootvg file systems before reboot:

   ```
   alt_disk_install -d /mksysb_images/4.3_mksysb -s /home/myscript hdisk3
   ```
3. To remove the original rootvg ODM database entry, after booting from the new alternate disk:

   ```
   alt_disk_install -X old_rootvg
   ```

   The **lspv** listing for the original rootvg will be changed to "None". Therefore, a new volume group could be created on those disks.
4. To determine the boot disk for a volume group with multiple physical volume:

   ```
   alt_disk_install -q hdisk0
   ```

Illustrated Example

```
# lspv
hdisk0          00006091aef8b687     old_rootvg
hdisk1          00076443210a72ea     rootvg
hdisk2          0000875f48998649     old_rootvg
# alt_disk_install -q hdisk0
hdisk2
```

In this case, the boot disk for "old_rootvg" is actually hdisk2. Therefore, you could reset your bootlist to hdisk2 and reboot to the original rootvg volume group.

5. To modify an **alt_disk_install** volume group name:

```
alt_disk_install -v alt_disk_432 hdisk2
```

Illustrated Example

```
# lspv
hdisk0          00006091aef8b687     rootvg
hdisk1          00000103000d1a78     rootvg
hdisk2          000040445043d9f3     altinst_rootvg
hdisk3          00076443210a72ea     altinst_rootvg
hdisk4          0000875f48998649     None
hdisk5          000005317c58000e     None
# alt_disk_install -v alt_disk_432   hdisk2
#lspv
hdisk0          00006091aef8b687     rootvg
hdisk1          00000103000d1a78     rootvg
hdisk2          000040445043d9f3     alt_disk_432
hdisk3          00076443210a72ea     alt_disk_432
hdisk4          0000875f48998649     None
hdisk5          000005317c58000e     None
```

6. To "wake_up" an original rootvg, after booting from the new alternate disk:

```
alt_disk_install -W hdisk0
```

Illustrated Example

```
# lspv
hdisk0          000040445043d9f3     old_rootvg
hdisk1          00076443210a72ea     rootvg
# alt_disk_install -W hdisk0
# lspv
hdisk0          000040445043d9f3     altinst_rootvg
hdisk1          00076443210a72ea     rootvg
```

At this point, the "altinst_rootvg" volume group is varied-on and the /alt_inst file systems will be mounted.

7. To "put-to-sleep" a volume group that had experienced a "wake-up":

```
alt_disk_install -S
```

Illustrated Example

```
# lspv
hdisk0          000040445043d9f3     altinst_rootvg
hdisk1          00076443210a72ea     rootvg
# alt_disk_install -S
# lspv
hdisk0          000040445043d9f3     altinst_rootvg
hdisk1          00076443210a72ea     rootvg
```

The "altinst_rootvg" is no longer varied-on and the /alt_inst file systems are no longer mounted. If it's necessary for the "altinst_rootvg" volume group name to be changed back to "old_rootvg", this can be done with the "-v" flag.

Files

Item	Description
/usr/sbin/alt_disk_install	Contains the alt_disk_install command

alt_disk_mksysb Command

Purpose

Installs an alternate disk with a **mksysb** install base install image.

Syntax

alt_disk_mksysb -m *device* -d *target_disks...* [-i **image.data**] [-s *script*] [-R *resolv_conf*] [-p *platform*] [-L *mksysb_level*] [-n] [-P *phase_option*] [-c *console*] [-K] [-D B O V g k r y z T S **C**]

Description

The **alt_disk_mksysb** command allows the users to install a **mksysb** system backup to a separate disk without taking the machine down for an extended period, thus mitigating outage risk. Using the **alt_disk_mksysb** command is the only method available to restore a backup containing **multibos** Base Operating System (BOS) instances.

An AIX level of the **mksysb** image, the **mksysb** tape, or the **mksysb** CD is required to install an **mksysb** system. The **alt_disk_mksysb** command is called with a disk or a set of disks that is currently not in use, and the **mksysb** image is restored to disks such that, if the user chooses, the next reboot boots the system on an AIX level of the **mksysb** image.

The **bos.alt_disk_install.rte** and **bos.alt_disk_install.boot_images** filesets must be installed on the system to run the **alt_disk_mksysb** command.

The **mksysb** image that is used must have all the necessary device and kernel support required for the system it is installed on. You cannot install a new device or kernel support before the system is rebooted from the newly installed disk.

The alternate root file system is mounted as **/alt_inst** to ensure that the other file systems have a prefix, such as **/alt_inst/usr**, **/alt_inst/var**). This is the method in which the files must be accessed using a customization script.

At the end of the install, a volume group, **altinst_rootvg**, is left on the target disks in the varied-off state as a place holder. If varied on, it indicates that it owns no logical volumes; however, it does contain logical volumes, but they have been removed from the ODM because their names now conflict with the names of the logical volumes on the running system. Do not vary on the **altinst_rootvg** volume group; instead, leave the **altinst_rootvg** volume group as a placeholder.

After the system reboots from the new alternate disk, the former rootvg volume group shows up in the lspv listing as `old_rootvg`. Do not vary on the **old_rootvg** volume group; instead, leave the **old_rootvg** volume group as a placeholder.

If a return to the original **rootvg** is necessary, the **bootlist** command is used to change the bootlist to reboot from the original **rootvg**.

Notes:

1. Alternate disk operations create volume groups, logical volumes, special device files, and file systems using the **alt** prefix. If **alt_disk_copy** is used on a system, the administrator must avoid having or creating volume groups, logical volumes, special device files, or file systems with the **alt**, prefix—alternate disk operations might inadvertently remove, alter, or damage these items.

2. **alt_disk_mksysb** needs to use preexisting boot images during **mksysb** installation. **alt_disk_mksysb** first looks for the boot images in the alternate **rootvg** (that is, the contents of the **mksysb**); if boot images are not found, **alt_disk_mksysb** searches for them in the current **rootvg**.
   - The alternate disk install boot image location for **altinst_rootvg** is: **/alt_inst/usr/lpp/bos.alt_disk_install/boot_images**
   - The alternate disk install boot image location for the current **rootvg** is: **/usr/lpp/bos.alt_disk_install/boot_images**
   - The generic versions of the alternate install boot images are provided by the **bos.alt_disk_install.boot_images** fileset.

   If the pre-existing boot image in the **mksysb** command does not work because of additional interim fixes or updates that affect the kernel, you can use the **–C** flag. The **-C** flag uses only the alternate disk installation boot image from the current rootvg volume group. After you save a copy of the original `/usr/lpp/bos.alt_disk_install.boot_images/bosboot.disk.chrp` image, the image can be replaced with a new image, which is built on the source system by using the *bosboot –a –b <new_location>* command. You can build the image on another system if the software installed matches the **mksysb** source system. The **–C** flag is not supported when you run the **alt_disk_mksysb** command by using Network Installation Manager (NIM).

   Alternatively, if you know other interim fixes that affects the kernel on the source system, you can create the **mksysb** image from that system by using the **–C** flag. This process customizes the boot image when you create the **mksysb** image.
3. The version, release, maintenance or technology level of the **mksysb** command that you are installing must match the level of the **bos.alt_disk_install.boot_images** fileset. For example, if the `oslevel` on the source system (the system where the **mksysb** command was created) returns `6.1.0.0`, the **bos.alt_disk_install.boot_images** fileset must be at 6.1.0.*X*, where *X* is the highest available fix level.
4. If **alt_disk_mksysb** needs to use the generic boot images shipped with the **bos.alt_disk_install.boot_images** fileset, the system performs an additional reboot when booting from the alternate **rootvg** for the first time.
5. You cannot use the **alt_disk_mksysb** command to install an earlier version of the AIX Version 7.1 than the version of the AIX that is installed on the system. For example, you cannot install an AIX Version 6.1 **mksysb** on a system that is running AIX Version 7.1 operating system. For a **multibos mksysb**, the version of the active AIX that is used to create the **mksysb** will be the AIX version of the **mksysb**.
6. The current LVM limit for logical volume names is 15 characters. Because the alternate disk installation commands contain the 4-character **alt_** prefix, the limit for the original logical volume names in the **rootvg** to be copied or installed is 11 characters. If an original logical volume name exceeds 11 characters, it can be shortened using a customized **image.data** (see the **-i** flag).
7. Do not use direct LVM commands (such as **exportvg**, **importvg**, **varyoffvg**, and **chlv**) on alternate **rootvg** volume groups.
8. The **alt_disk_mksysb** function is also available on NIM. The **–C** flag is not supported when you run the **alt_disk_mksysb** command by using NIM.

Flags

Item	Description
-B	Specifies not running bootlist after the operation. If set, then the **-r** flag cannot be used.
-C	Specifies to use the `/usr/lpp/bos.alt_disk_install/boot_images/bosboot.disk.chrp` file from the current rootvg volume group only. This flag is not supported when you run the **alt_disk_mksysb** command by using NIM. This flag does not affect a standby **multibos** Base Operating System (BOS) instance of the AIX operating system. For more information about the **-C** flag, see the **Notes** section.

Item	Description
-c *console*	Specifies the device name to be used as the alternate `rootvg`'s system console. This option is only valid with the `-0` flag.
-D	Turns on debug (sets **-x** output).
-d *target_disks*	Specifies a space-delimited list of the name or names of the target disks where the alternate **rootvg** is created. This disk or these disks must not currently contain any volume group definition. The **lspv** command must indicate that these disks belong to volume group **None**.
-g	Specifies that bootable checks for the **target_disks** are overlooked.
-K	Specifies that the 64 - bit kernel must be used, if possible.
-k	Specifies that **mksysb** devices be kept (formally the **ALT_KEEP_MDEV** variable).
-i *image_data*	Optional **image.data** file to use instead of the default **image.data** file from **mksysb** image. The **image.data** file name must be a full path name (for example, **/tmp/my_image.data**).
-L *mksysb_level*	This level is combined with the platform type to create the boot image name (for example, **rspc_6.1.0_boot** in AIX 6.1 and earlier). This must be in the form V.R.M. The **mksysb** image is checked against this level to verify that they are the same.
-m *device*	The value for device can be: • Tape device (for example, **/dev/rmt0**) • Path name of **mksysb** image in a file system
-n	Remain NIM client. The **/.rhosts** and **/etc/niminfo** files are copied to the alternate **rootvg**'s file system.
-P *Phases*	The phase or phases to execute during this invocation of the **alt_disk_mksysb** command. Valid values are: 1, 2, 3, 12, 23, or all. **12** Performs phases 1 and 2. **23** Performs phases 2 and 3. **all** Performs all three phases.
-p *platform*	Platform used to create the name of the disk boot image, which might be supplied by a vendor that wanted to support this function.
-O	Performs a device reset on the target **altinst_rootvg**. This causes **alt_disk_install** to not retain any user-defined device configurations. This flag is useful if the target disk or disks become the **rootvg** of a different system (such as in the case of logical partitioning or system disk swap).
-R *resolv_conf*	The **resolv.conf** file that replaces the existing one after the **mksysb** has been restored. You must use a full path name for *resolv_conf*.
-r	Specifies to reboot from the new disk when the **alt_disk_mksysb** command is complete.

Item	Description
-s *script*	Optional customization script to run at the end of the **mksysb** install. This file must be executable. This script is called on the running system before the **/alt_inst** file systems are unmounted, so files can be copied from the running system to the **/alt_inst** file systems before the reboot. This is the only opportunity to copy or modify files in the alternate file system because the logical volume names will be changed to match **rootvg**'s, and they will not be accessible until the system is rebooted with the new alternate **rootvg**, or a "wake-up" is performed on the **altinst_rootvg** using the **alt_rootvg_op** command. You must use a full path name for the script.
-S	Indicates that you want to skip space-checking on target disks before you start performing the cloning or installation operations. **Important :** JFS2 file systems contain more metadata than JFS file systems. When you use the -S flag with the -T flag, it skips space-checking. In this situation, it does not verify that there is enough space in the newly created JFS2 file system to store the contents of the file system plus the additional metadata.
-T	Indicates that you want to convert JFS file systems to JFS2 file systems during the process of recreating the **rootvg** volume group on target disks.
-V	Turn on verbose output. This shows the files that are restored during the **alt_disk_mksysb** operation.
-y	Looks for and imports (if found) **mksysb** volume groups. This flag causes **alt_disk_install** to import the data VGs known to the **mksysb** and to not import the local data VGs known at install time (the default). The imports are performed with the following script: **/usr/lpp/bos.alt_disk_install/bin/alt_import_oldvgs**.
-z	Does not import any type of non-**rootvg** volume groups. This flag overrides the **-y** flag.

Exit Status

Item	Description
0	All **alt_disk_mksysb** related operations completed successfully.
>0	An error occurred.

Examples

1. To install a **mksysb** image on **hdisk3** and **hdisk4** , then run a customized script (**/tmp/script**) to copy some user files over to the alternate **rootvg** file systems before reboot:

   ```
   alt_disk_mksysb -m /mksysb_images/my_mksysb -d "hdisk3 hdisk4" -s /tmp/script
   ```

2. To install a **mksysb** image on **hdisk2** and stop after phase 1:

   ```
   alt_disk_mksysb -m /mksysb_images/my_mksysb -d hdisk2 -P1
   ```

   Attention : Do not change the bootlist to use the cloned **rootvg**.

3. To execute phases 2 and 3 on an existing alternate **rootvg** on **hdisk4** and reboot the system upon successful completion:

   ```
   alt_disk_mksysb -d hdisk4 -m /mksysb_images/my_mksysb -P23 -r
   ```

4. To install a **mksysb** image on **hdisk1**, and to convert the file system from a JFS file system to a JFS2 file system, run the following command:

```
alt_disk_mksysb -B -T -m  /mksysb_images/my_mksysb -d hdisk1
```

Location

/usr/sbin/alt_disk_mksysb

Files

Item	Description
/usr/sbin/alt_disk_mksysb	Contains the **alt_disk_mksysb** command.

alt_rootvg_op Command

Purpose

Performs operations on existing alternate **rootvg** volume groups.

Syntax

To determine Volume Group Boot Disk (-q):

alt_rootvg_op -q -d *disk* [-D]

To rename Alternate Disk Volume Group (-v):

alt_rootvg_op -v *new volume group name* -d *disk* [-D]

To wake up Volume Group (-W):

alt_rootvg_op -W -d *disk* [-D]

To put to sleep Volume Group (-S):

alt_rootvg_op -S [-tD]

To clean up Alternate Disk Volume Group (-X):

alt_rootvg_op -X [*volume group*] [-D]

To customize Alternate Disk Volume Group (-C):

alt_rootvg_op -C [-R *resolv_conf*] [-s *script*] [-b *bundle_name*] [-I *installp_flags*] [-l *images_location*] [-f *fix_bundle*] [-F *fixes*] [-w *filesets*] [-DV]

Description

The **alt_rootvg_op** command can be used to determine which disk is the boot disk for a specific volume group. Use the **-q** flag to determine the boot disk. This can be useful when a volume group is comprised of multiple disks and a change in the bootlist is necessary.

This command can also be used to rename the alternate disk volume groups. This is especially useful when creating multiple alternate disks, on multiple volume groups, and name identification is necessary.

If data access is necessary between the current **rootvg** and an alternate disk, use the **alt_rootvg_op** command to perform a volume group *wake-up* operation (using the **-W** flag) on the non-booted volume group. The wake-up puts the volume group in a post phase 1 state (that is, the **/alt_inst** file systems will be mounted). The customize operation (**-C** flag) can be executed at this time. The wake-up operation is not intended to execute commands in a `chroot` environment.

The operating system version in the LPAR must be greater than or equal to the operating system version of the volume group that undergoes the wake-up operation. Therefore, you must boot the LPAR from the **altinst_rootvg** volume group and *wake* up the **old_rootvg** volume group. If the operating system in the **altinst_rootvg** volume group is a greater Technology Level (TL) or Service Pack (SP) level than the global volume groups, although the wake-up operation is allowed, some operations on the **altinst_rootvg** volume group might not work properly.

The **alt_rootvg_op** command does not allow a wake-up to occur on a volume group with a greater operating system version, unless the **FORCE** environment variable is set to Yes.

Note:

1. The volume group that experiences the wake-up is renamed **altinst_rootvg**.
2. Do not execute phase 3 on the volume group that experiences the wake-up.
3. Do not reboot the system if there is a volume group in the `wake` state. This can cause damage or data loss to the volume group that is in the wake state. Volume groups in the `wake` state can be put to `sleep` with the **-S** flag.

When data access is no longer needed, the **alt_rootvg_op** command can be used to put to sleep the volume group in the `wake` state, using the **-S** flag. The boot image on the target alternate **rootvg** can be rebuilt if necessary with the **-t** flag. The sleep operations revert the alternate volume group to an inactive state.

When cleaning up the alternate disk volume group, the **alt_rootvg_op** command uses the -X flag to remove the **altinst_rootvg** volume group definition from the ODM database. If the target volume group is varied off at the time this operation is executed, only the ODM definitions associated with the target volume group are removed. The actual volume group data is not removed. If the volume group is bootable, you can still reboot from that volume group, by setting the bootlist to a boot disk in this volume group. The **-X** flag accepts a volume group name as an argument and acts on the **altinst_rootvg** volume group by default.

The customize operation of the **alt_rootvg_op** command (using the **-C** flag) can be used to perform the following functions on an active alternate root volume group:

- Install software and software updates. Apply this operation only to alternate volume groups created with the **rootvg** copy operation.
- Execute customization script.
- Copy **resolv.conf** files.

Flags

Item	Description
-b *bundle_name*	Path name of optional file with a list of packages or filesets that will be installed after a **rootvg** clone. The **-l** flag must be used with this option.
-C	Performs the customization operation on the active **rootvg** volume group.
-d *target_disk*	Specifies a space-delimited list of the name or names of the target disks that will be targets of the given operation.
-D	Turns on debug (sets **-x** output).
-f *fix_bundle*	Optional file with a list of APARs to install after a clone of **rootvg**. The **-l** flag must be used with this option.
-F *fixes*	Optional list of APARs (for example, IY123456) to install after a clone of **rootvg**. The **-l** flag must be used with this option.
-I *installp_flags*	The flags to use when updating or installing new filesets into the cloned **altinst_rootvg**. The default flag is **-acgX**. The **-l** flag must be used with this option.

Item	Description
-l *images_location*	Location of **installp** images or updates to apply after a clone of **rootvg**. This can be a directory full path name or device name (like **/dev/rmt0**).
-q	Determines the volume group boot disk.
-R *resolv_conf*	The **resolv.conf** file to replace the existing one in the **rootvg**. You must specify a full path name.
-s *script*	Optional customization script to be executed during the customization phase. This file must be executable. This script is called on the running system before the **/alt_inst** file systems are unmounted, so files can be copied from the running system to the **/alt_inst** file systems before the reboot.
-S	Puts to sleep the alternate root volume group that experienced the previous "wake" operation.
-t	Rebuilds the alternate boot image before putting the volume group to "sleep." This flag is only valid for alternate root volume groups created with the clone or copy install operation. The **-t** flag requires the **-S** flag.
-v *Name*	Renames an alternate disk volume group to the name specified with the *Name* parameter.
-V	Turn on verbose output.
-w *filesets*	List of filesets to install after cloning a **rootvg**. The **-l** flag must be used with this option.
-W	Performs a wake-up on the root volume group located on the **target_disk**.
-X	Removes the **altinst_rootvg** volume group definition from the ODM database.

Exit Status

Item	Description
0	All **alt_rootvg_op** related operations completed successfully.
>0	An error occurred.

Examples

1. To remove the original **rootvg** ODM database entry, after booting from the new alternate disk, enter the following command:

   ```
   alt_rootvg_op -X old_rootvg
   ```

2. To cleanup the current alternate disk install operation, enter the following command:

   ```
   alt_rootvg_op -X
   ```

3. To determine the boot disk for a volume group with multiple physical volume, enter the following command:

   ```
   alt_rootvg_op -q -d hdisk0
   ```

   Illustrated Example

   ```
   # lspv

   hdisk0          00006091aef8b687     old_rootvg
   hdisk1          00076443210a72ea     rootvg
   hdisk2          0000875f48998649     old_rootvg

   # alt_rootvg_op -q -d hdisk0
   ```

```
hdisk2
```

4. To modify an **alt_disk_install** volume group name, enter the following command:

```
alt_rootvg_op -v alt_disk_530 -d hdisk2
```

Illustrated Example

```
# lspv

hdisk0          00006091aef8b687     rootvg
hdisk1          00000103000d1a78     rootvg
hdisk2          000040445043d9f3     altinst_rootvg
hdisk3          00076443210a72ea     altinst_rootvg
hdisk4          0000875f48998649     None
hdisk5          000005317c58000e     None

# alt_rootvg_op -v alt_disk_432 -d hdisk2

#lspv

hdisk0          00006091aef8b687     rootvg
hdisk1          00000103000d1a78     rootvg
hdisk2          000040445043d9f3     alt_disk_432
hdisk3          00076443210a72ea     alt_disk_432
hdisk4          0000875f48998649     None
hdisk5          000005317c58000e     None
```

5. To "wake up" an original **rootvg** after booting from the new alternate disk, enter the following command:

```
alt_rootvg_op -W -d hdisk0
```

6. To "put to sleep" a volume group that had experienced a "wake-up" and rebuild the boot image, enter the following command:

```
alt_rootvg_op -S -t
```

7. To update the active alternate **rootvg** to the latest fileset levels available in **/updates** and install them into the alternate **root** volume group, enter the following command:

```
alt_rootvg_op -C -b update_all -l /updates
```

Location

/usr/sbin/alt_rootvg_op

Files

Item	Description
/usr/sbin/alt_rootvg_op	Contains the **alt_rootvg_op** command.

amepat Command

Purpose

Active Memory Expansion Planning and Advisory Tool **amepat** reports Active Memory Expansion (AME) information and statistics as well as provides advisory report that assists in planning the use of Active Memory Expansion for an existing workload.

Syntax

amepat [{{[**-c** *max_ame_cpuusage%*] | [**-C** *max_ame_cpuusage*]}|[**-e** *startexpfactor [:stopexpfactor [:incexpfactor]*]]}][{[**-t** *tgt_expmem_size*]|[**-a**]}]

[**-n** *num_entries*] [**-m** *min_mem_gain*] [**-u** *minucomp_poolsize*]

[**-v**] [**-M**] [**-N**] [-O proc=<*processor implementation*>][{ [**-P** *recfile*] | [**Duration**] | [**Interval <Samples>**]}]

amepat [**-N**] [**-R** *recfile*] {[**Duration**] | [**Interval <Samples>**]}

Description

Active Memory Expansion Planning and Advisory Tool **amepat** serves two key functions:

1. **Workload Planning -** The **amepat** can be run to determine a workload that would benefit from Active Memory Expansion, and also to provide a list of possible Active Memory Expansion configurations for a workload.
2. **Monitoring -** When Active Memory Expansion is enabled, the **amepat** tool can be used to monitor the workload and Active Memory Expansion performance statistics.

The **amepat** can be started in two different modes:

1. In the **Recording** mode **amepat** records systems configuration and various performance statistics into a user specified recording file.
2. In the **Reporting** mode **amepat** analyzes the system configuration and performance statistics, collected in real time or from the user specified recording file, to generate workload utilization and planning reports.

Note: This tool is available from AIX Version 6.1 with the 6100-04 Technology Level-SP2 release, or later.

Workload Planning

When considering using Active Memory Expansion for an existing workload, **amepat** can be used to provide guidance on possible Active Memory Expansion configurations for the workload. When **amepat** is run concurrently with an existing workload that is not using Active Memory Expansion, **amepat** monitors the memory usage, memory reference patterns, and data compressibility over a user-configurable time period of the workload. The tool then generate a report with a list of possible Active Memory Expansion configurations for the workload. The tool includes an estimate of the processor utilization impacts for the different Active Memory Expansion configurations.

The **amepat** command can be run on all versions of IBM® Power Systems servers supported by AIX 6.1, and later.

There are two key considerations when running **amepat** to do workload planning: the time at which to run the tool and the duration to run the tool. To get the best possible results from the tool, the tool must be run during the period of peak utilization of the workload. It ensures that the tool captures peak of utilization and memory usage information of the workload.

To use **amepat** to generate a report for workload planning, a monitoring duration must be specified when starting **amepat**.

In addition to using **amepat** on workload that are not yet using Active Memory Expansion, **amepat** can also be run in LPAR's where Active Memory Expansion is already enabled. When used in this mode, **amepat** it provides a report of other possible Active Memory Expansion configurations for the workload.

Note: amepat requires privileged access to do Workload Planning. When a user starts the tool without the required privilege then the Workload Planning Capability is disabled (**-N** flag is turned on implicitly)

Monitoring

amepat can also be used to monitor the processor and memory utilization statistics (Disabling the workload planning capability). With this Monitoring capability, **amepat** just gathers processor and memory

utilization statistics, does not gather the additional data required for generating the report for workload planning. Thus, Active Memory Expansion Modeling and Advisory reports are not generated.

When **amepat** is started without a duration or interval, **amepat** defaults to monitoring only capability, and **amepat** reports a snapshot of the LPAR's memory, processor utilization.

amepat can be started with duration and run with Monitoring only capability using the **-N** flag. The **-N** flag disables the workload planning capability of this tool, thus disabling the data gathering process & reporting for workload planning.

Note: Both Recording and Reporting modes can be started with **-N** flag. The **-N** flag is supported both in Active Memory Expansion Enabled and Disabled Machines.

amepat Report

Following are the six different sections of report displayed by the **amepat** tool:

Command Information Section

The Command Information Section provides details about the arguments passed to the **amepat** tool, time of invocation, the total time the system is monitored and the number of samples collected.

System Configuration Section

The System Configuration Section provides details about the system configuration. The following table provides the complete list of information reported.

Item	Description
Partition Name	Node name from where **amepat** is started
Processor Implementation Mode	The processor implementation mode. It can be POWER4, POWER5, POWER6®, and so on.
Number Of Logical CPUs	The total number of logical processors configured and active in the partition.
Processor Entitled Capacity	Capacity Entitlement of the partition, represented in the unit of number of physical processors. **Note:** The physical processor units can be in fraction as well, for example, 0.5 physical processor.
Processor Max. Capacity	Maximum Capacity this partition can have, represented in the unit of number of physical processors **Note:** The physical processor units can be in fraction as well, for example, 0.5 physical processor.
True Memory	The true memory represents real physical or logical memory configured for this LPAR.
SMT Threads	Number of SMT threads configured in the partition. The value can be **1**, **2**, **4** or **8**. **Note:** The maximum number of SMT threads per processor is based on the Power® Architecture.

Item	Description
Shared Processor Mode	Indicates whether Shared Processor Mode is configured for this partition. The possible values are: **Disabled** Shared Processor Mode is not configured. **Enabled-Capped** Shared Processor Mode is enabled & running in capped mode. **Enabled-Uncapped** Shared Processor Mode is enabled & running in uncapped mode.
Active Memory Sharing	Indicates whether Active Memory Sharing is **Enabled** or **Disabled**
Active Memory Expansion	Indicates whether Active Memory Expansion is **Enabled** or **Disabled**
Target Expanded Memory Size	Indicates the target expanded memory size in MB for the LPAR. The Target Expanded Memory Size is the True Memory Size multiplied by the Target Memory Expansion Factor. **Note:** This get displayed only when Active Memory Expansion is enabled
Target Memory Expansion factor	Indicates the target memory expansion factor configured for the LPAR. **Note:** This get displayed only when Active Memory Expansion is enabled

System Resource Statistics

System Resource Statistics provides details about the system resource utilization from CPU/Memory Stand point. The following table shows various statistics related to system resource utilization

Item	Description
CPU Util	The Partition's processor utilization in the units of number of physical processors. The percentage of utilization against the Maximum Capacity is also reported. **Note:** If Active Memory Expansion is enabled, the processor utilization due to memory compression / decompression is also included
Virtual Memory Size	The Active Virtual Memory Size in MB. The percentage against the True Memory Size is also reported.
True Memory In-Use	This is amount of the LPAR’s real physical (or logical) memory in MB. The percentage against the True Memory Size is also reported.
Pinned Memory	This represents the pinned memory size in MB. The percentage against the True Memory Size is also reported.
File Cache Size	This represents the non-computational file cache size in MB. The percentage against the True Memory Size is also reported.
Available Memory	This represents the size of the memory available, in MB, for application execution. The percentage against the True Memory Size is also reported.

Note: For all the utilization metrics Average, Minimum and Maximum values get displayed if **amepat** is run with duration/interval.

Active Memory Expansion Statistics

Active Memory Expansion Statistics provides details about the Active Memory Expansion statistics. This section is only displayed if Active Memory Expansion has been enabled for the LPAR. The following table describes the various statistics that are reported

Item	Description
AME processor Usage	The processor utilization for Active Memory Expansion activity in units of physical processors. It indicates the amount of processing capacity used for memory compression activity. The percentage of utilization against the Maximum Capacity is also reported.
Compressed Memory	The total amount of virtual memory that is compressed. This is measured in MB. The percentage against the Target Expanded Memory Size is also reported.
Compression Ratio	This represents how well the data is compressed in memory. A higher compression ratio indicates that the data compresses to a smaller size. For example, if 4 KB of data can be compressed down to 1 KB, then the compression ratio is 4.0.
Deficit Memory Size	The size of the expanded memory, in MB, deficit for the LPAR. This is only displayed if the LPAR has a memory deficit. The percentage against the Target Expanded Memory Size is also reported.

Note: The Active Memory Expansion Statistics section displays only when the tool is started in an Active Memory Expansion enabled machine. It also displays the average, minimum and maximum values of the statistics when the tool started with duration/ interval.

Active Memory Expansion Modeled Statistics

Active Memory Expansion Modeled Statistics provides details about the modeled statistics for Active Memory Expansion. The following table provides the information about the modeled statistics.

Item	Description
Modeled Expanded Memory Size	It represents the size of expanded memory that is used to produce the modeled statistics.
Average Compression Ratio	It represents the average compression ratio of the in-memory data of the workload. This compression ratio is used to produce the modeled statistics.
Modeled Expansion Factor	It represents the modeled target memory expansion factor.
Modeled True Memory Size	It represents the modeled true memory size (real physical or logical memory)
Modeled Memory Gain	It represents the amount of memory the partition can gain by enabling Active Memory Expansion for the reported modeled expansion factor
AME processor Usage Estimate	It represents an estimate of the processor that would be used for Active Memory Expansion activity for the specified configuration. It estimates the amount of processing capacity that would be used for memory compression activity. The processor usage is reported in units of physical processors. The percentage of utilization against the Maximum Capacity is also reported. **Note:** This is just an estimate and should only be used as guidance; the actual usage can be higher or lower depending on the workload.

Item	Description
Modeled Implementation	It represents the processor implementation for which modeling is done. This is available only if the **–O proc** option is used.

Note: This section is displayed only when **-N** flag is not used & when run by a privileged user. The generation of Modeled statistics requires Operating System to do certain simulation operation; hence the actual duration of monitoring can be higher than the user specified monitoring time.

Recommendation

Recommendation provides details about the Active Memory Expansion configuration that would provide optimal benefits to the current running workload.

Note: The recommendations are purely done based on the behavior during the monitoring period of the workload and hence the recommendations provided can be used only as guidance. The actual statistics can vary based on the actual behavior in real time of the workload.

Note: Active Memory Expansion Modeled Statistics & Recommendation are used for Workload Planning. When **-N** is specified both these reports is not displayed. Active Memory Expansion Statistics is reported only when running in Active Memory Expansion Enabled System.

amepat can be started using the System Management Interface Tool (SMIT) **smit amepat** fast path to run this command.

Note: This command is restricted inside WPAR. When **amepat** is started without specifying duration or interval then the utilization statistics(System, AME) will not display any Average, Minimum, or Maximum values. It just displays the Current value. The processor utilization just displays the average from the system boot time.

Note: When the Active Memory Expansion is enabled, multiple page size support is disabled and only the page sizes of 4 KB and 64 KB are used.

Flags

Item	Description
-a	Specifies to auto-tune the expanded memory size for Active Memory Expansion Modeled Statistics. When this option is selected, the Modeled Expanded Memory Size is estimated based on the current memory usage of the workload (excludes the available memory size). **Note:** The **-a** and **-t** options are mutually exclusive.
-c *max_ame_cpuusage%*	Specifies the maximum Active Memory Expansion processor usage in terms of percentage to be used for producing the Modeled statistics & recommendation. **Note:** The default maximum used is 15%. The **-C** and **-c** option cannot be specified together. The **-c** and **-e** options are mutually exclusive.
-C *max_ame_cpuusage*	Specifies the maximum Active Memory Expansion processor usage in terms of number of physical processors to be used for producing the Modeled statistics and recommendation. **Note:** The **-C** and **-c** option cannot be specified together. The **-C** and **-e** option are mutually exclusive.

Item	Description
-e *startexpfactor:stopexpfactor:incexpfactor*	Specifies the range of expansion factors to be reported in the Active Memory Expansion Modeled Statistics section. **Startexpfactor** Starting expansion factor. This field is mandatory if **-e** is used. **Stopexpfactor** Stop expansion factor. If not specified then the modeled statistics is generated for the start expansion factor alone. **incexpfactor** Incremental expansion factor. Allowed range is 0.01-1.0. Default is 0.5. Stop expansion factor need to be specified to specify incremental expansion factor. **Note:** The **-e** option cannot be combined with **-C** or **-c** options.
-m *min_mem_gain*	Specifies the Minimum Memory Gain. This value is specified in MB. This value is used in determining the various possible expansion factors reported in the Modeled Statistics and also influence the produced recommendations.
-M	Does not break the 64 KB page into 4 KB chunks and compresses the entire 64 KB page when the workloads are modeled. **Note:** The **-M** flag can be specified only in POWER8 processor-based servers, or later.
-n *num_entries*	Specifies the number of entries that need to be displayed in the Modeled Statistics. **Note:** When **-e** with **incexpfactor** specified then **-n** value is ignored.
-N	Disable Active Memory Expansion Modeling (Workload Planning Capability)
-O proc=*processor implementation*	Specifies the processor implementation for which modeling is done. You can specify the following processor versions: • P7 or p7 • P7+ or p7+ • P8 or p8 • P9 or p9 • ALL or all (Displays all current AME supported processors) **Note:** The **-O** option cannot be specified with the **–R** option.
-P *recfile*	Process the specified recording file and generate report.
-R *recfile*	Record the active memory expansion data in the specified recording file. The recorded data can be post processed later using the **-P** option. **Note:** Only **-N** option can be combined with -R.
-t *tgt_expmem_size*	Specifies the Modeled Target Expanded Memory Size. This makes the tool to use the user specified size for modeling instead of the calculated one. **Note:** The **-t** and **-a** options are mutually exclusive.

Item	Description
-u *minuncompressedpoolsize*	Specifies the minimum uncompressed pool size in MB. This value over-rides the tool calculated value for producing Modeled Statistics. **Note:** This flag can be used only when Active Memory Expansion is disabled.
-v	Enables Verbose Logging. When specified a verbose log file is generated, named as **amepat_yyyymmddhmm.log**, where **yyyymmddhmm** represents the time of invocation. **Note:** The verbose log also contains detailed information on various samples collected and hence the file will be larger than the output generated by the tool.
Duration	Duration represents the amount of total time the tool need to monitor the system before generating any reports. **Note:** When duration is specified interval/samples cannot be specified. The interval & samples will be determined by the tool automatically. The actual monitoring time can be higher than the duration specified based on the memory usage and access patterns of the workload.
Interval <Samples>	Interval represents the amount of sampling time, Samples represents the number of samples need to be collected. **Note:** When interval, samples are specified, duration is calculated automatically as (interval x Samples). The actual monitoring time can be higher than the duration specified based on the memory usage and access patterns of the workload.

Notes:

1. The default behavior of the **amepat** command on a modeling report would be as follows:
   - When the **amepat** command is run on POWER7 or earlier processor implementations, the default modeled processor implementation is POWER7.
   - When the **amepat** command is run on a processor implementation later than POWER7, the default modeled processor is the same as the processor implementation where it runs.
2. When AME is enabled, the **-O proc** option can be used to model processors equal or newer than the processor implementation where the **amepat** command is running.
3. The **amepat** command facilitates the user to provide minimum and/or maximum values for certain flags (like the **-e** flag) that helps alter the modeling behavior. The specified values are taken as suggested values by the **amepat** command. The **amepat** command overrides these values if they are not within the permissible ranges determined by the command during its course of execution.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

```
ATTENTION: RBAC users and Trusted AIX users:
```

This command can perform privileged operations. Only privileged users can run privileged operations.

Examples

1. To display Active Memory Expansion Monitoring only report, enter:

   ```
   amepat
   ```

2. To monitor the workload, for the duration of 16 minutes with 8 minute sampling interval and 2 samples, generate report for Workload Planning, enter:

   ```
   amepat 8 2
   ```

3. To monitor the workload for a duration of 16 minutes and generate Active Memory Expansion report for Workload Planning with modeled memory expansion factors between 1.5 and 3 at 0.5 incremental factor, enter:

   ```
   amepat –e 1.50:3.00:0.5 16
   ```

4. To monitor the workload for a duration of 16 minutes and generate Active Memory Expansion report for Workload Planning with capping the modeled AME processor usage to 30%, enter:

   ```
   amepat –c 30 16
   ```

5. To monitor the workload for a duration of 16 minutes and generate Active Memory Expansion report for Workload Planning with starting modeled memory gain of 1000 MB, enter:

   ```
   amepat –m 1000 16
   ```

6. To monitor the workload for a duration of 16 minutes and generate Active Memory Expansion report for Workload Planning by modeling a minimum uncompressed pool size 2000 MB, enter:

   ```
   amepat –u 2000 16
   ```

7. To use the recording mode of **amepat** to generate the recording file and generate reports with various filters, enter:

   Start Recording for a duration of 60 minutes.

   ```
   amepat -R myrecord_amepat 60
   ```

   Note: The recording mode will switch itself into background process.

   Generate Report for Workload Planning

   ```
   amepat -P myrecord_amepat
   ```

   Generate Report for Workload Planning with the modeled memory expansion factors ranging between 2 to 4 with 0.5 delta factor

   ```
   amepat -e 2.0:4.0:0.5 -P myrecord_amepat
   ```

   Generate Monitoring only report

   ```
   amepat -N -P myrecord_amepat
   ```

8. To disable Workload Planning Capability & monitor the system for 30 minutes, enter:

   ```
   amepat -N 30
   ```

9. To monitor the workload for a duration of 60 minutes and to model for Processor Implementation P8, enter the following command:

   ```
   amepat –O proc=P8  60
   ```

anno Command

Purpose

Annotates messages.

Syntax

anno [**+***Folder*] [*Messages*] [**-component** *Field*] [**-inplace** | **-noinplace**] [**-text "***String***"**]

Description

The **anno** command annotates messages with text and dates. If you enter the **anno** command without any flags, the system responds with the following prompt:

```
Enter component name:
```

Typing a component name and pressing the Enter key annotates the component name and system date to the top of the message being processed. You cannot annotate an existing field. You can only add lines to the top of a message file. The annotation fields can contain only alphanumeric characters and dashes.

> **Note:** To simply add distribution information to a message, use the **dist**, **forw**, or **repl** commands.

Flags

Item	Description
-component *Field*	Specifies the field name for the annotation text. The *Field* variable must consist of alphanumeric characters and dashes. If you do not specify this flag, the **anno** command prompts you for the name of the field.
+*Folder*	Identifies the message folder that contains the message to annotate. The default is the current folder.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH (Message Handler), the name of this flag must be fully spelled out.
-inplace	Forces annotation to be done in place in order to preserve links to the annotated messages.

Item	Description
Messages	Specifies what messages to annotate. This parameter can specify several messages, a range of messages, or a single message. If several messages are specified, the first message annotated becomes the current message. Use the following references to specify messages: ***Number*** Number of the message. When specifying several messages, separate each number with a comma. When specifying a range, separate the first and last number in the range with a hyphen. ***Sequence*** A group of messages specified by the user. Recognized values include: **all** All messages in the folder. **cur or . (period)** Current message. This is the default. **first** First message in a folder. **last** Last message in a folder. **next** Message following the current message. **prev** Message preceding the current message.
-noinplace	Prevents annotation in place. This flag is the default.
-text "*String***"**	Specifies the text to be annotated to the messages. The text must be enclosed with quotation marks.

Profile Entries

The following entries can be made to the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Current-Folder:`	Sets the default current folder.
`Path:`	Specifies the location of a user's MH (Message Handler) directory.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To annotate the message being processed with the date and time, enter:

   ```
   anno
   ```

   The following prompt is displayed on your screen:

```
Enter component name: _
```

After responding to this prompt, type:

```
Date
```

Press Enter. The component name you entered becomes the prefix to the date and time on the message. The caption appended to the message is similar to the following:

```
Date: Tues, 28 Mar 89 13:36:32 -0600
```

2. To annotate the message being processed with the date, time, and a message, enter:

   ```
   anno -component NOTE -text "Meeting canceled."
   ```

   A two-line caption similar to the following is appended to the message:

   ```
   NOTE: Mon, 15 Mar 89 10:19:45 -0600
   NOTE: Meeting canceled.
   ```

3. To annotate message 25 in the `meetings` folder, enter:

   ```
   anno +meetings 25 -component NOTE -text "Meeting delayed
   until Friday."
   ```

   The top of message 25 is annotated with a caption similar to the following:

   ```
   NOTE: Wed, 19 Jun 87 15:20:12 -0600
   NOTE: Meeting delayed until Friday.
   ```

   Note: Do not press the Enter key until the entire message has been entered, even though the message may be wider than the screen.

Files

Item	Description
$HOME/.mh_profile	Contains the MH user profile.
/usr/bin/anno	Contains **anno** command.

ap Command

Purpose

Parses and reformats addresses.

Syntax

ap [**-form** *File* | **-format** *String*] [**-normalize** | **-nonormalize**] [**-width** *Number*] *Address*

Description

The **ap** command parses and reformats addresses. The **ap** command is not started by the user. The **ap** command is called by other programs. The command is typically called by its full path name, **/usr/lib/mh/ap**.

The **ap** command parses each string specified by the address parameter and attempts to reformat it. The default output format for the **ap** command is the ARPA RFC 822 standard. When the default format is used, the **ap** command displays an error message for each string it is unable to parse.

Alternate file and string formats are specified by using the **-form** and **-format** flags.

Flags

Item	Description
-form *File*	Reformats the address string specified by the *Address* parameter into the alternate format described in the *File* variable.
-format *String*	Reformats the address string specified by the *Address* parameter into the alternate format specified by the *String* variable. The default format string follows: `%<{error}%{error}:%{Address}%:%(putstr(proper{ Address}))%>`
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-nonormalize	Does not attempt to convert local nicknames of hosts to their official host names.
-normalize	Attempts to convert local nicknames of hosts to their official host names. This flag is the default.
-width *Number*	Sets the maximum number of columns the **ap** command uses to display dates and error messages. The default is the width of the display.

Files

Item	Description
/etc/mh/mtstailor	Contains the MH tailor file.
$HOME/.mh_profile	Contains the MH user profile.

apply Command

Purpose

Applies a command to a set of parameters.

Syntax

apply [**-a***Character*] [**-***Number*] *CommandString Parameter* ...

Description

The **apply** command runs a command string specified by the *CommandString* parameter on each specified value of the *Parameter* parameter in turn. Normally, *Parameter* values are chosen individually; the optional **-***Number* flag specifies the number of *Parameter* values to be passed to the specified command string. If the value of the *Number* variable is 0, the command string is run without parameters once for each *Parameter* value.

If you include character sequences of the form %*n* (where *n* is a digit from 1 to 9) in *CommandString*, they are replaced by the *n*th unused *Parameter* value following the *CommandString* parameter when the command string is executed. If any such sequences occur, the **apply** command ignores the **-***Number* flag, and the number of parameters passed to *CommandString* is the maximum value of *n* in the *CommandString* parameter.

You can specify a character other than % (percent sign) to designate parameter substitution character strings with the **-a** flag; for example, `-a@` would indicate that the sequences @1 and @2 would be replaced by the first and second unused parameters following the *CommandString* parameter.

Notes:

1. Because pattern-matching characters in *CommandString* may have undesirable effects, it is recommended that complicated commands be enclosed in ' ' (single quotation marks).
2. You cannot pass a literal % (percent sign) followed immediately by any number without using the **-a** flag.

Flags

Item	Description
-a*Character*	Specifies a character (other than %) to designate parameter substitution strings.
-*Number*	Specifies the number of parameters to be passed to *CommandString* each time it is run.

Examples

1. To obtain results similar to those of the **ls** command, enter:

```
apply echo *
```

2. To compare the file named **a1** to the file named **b1**, and the file named **a2** to the file named **b2**, enter:

```
apply -2 cmp a1 b1 a2 b2
```

3. To run the **who** command five times, enter:

```
apply -0 who 1 2 3 4 5
```

4. To link all files in the current directory to the directory **/usr/joe**, enter:

```
apply 'ln %1 /usr/joe' *
```

apropos Command

Purpose

Locates commands by keyword lookup.

Syntax

apropos [**-M** *PathName*] *Keyword ...*

Description

The **apropos** command shows the manual sections that contain any of the keywords specified by the *Keyword* parameter in their title. The **apropos** command considers each word separately and does not take into account if a letter is in uppercase or lowercase. Words that are part of other words are also displayed. For example, when looking for the word `compile`, the **apropos** command also finds all instances of the word `compiler`. The database containing the keywords is **/usr/share/man/whatis**, which must first be generated with the **catman -w** command.

If the output of the **apropos** command begins with a name and section number, you can enter **man** *Section Title*. For example, if the output of the **apropos** command is `printf(3)`, you can enter `man 3 printf` to obtain the manual page on the **printf** subroutine.

The **apropos** command is equivalent to using the **man** command with the **-k** option.

> **Note:** When the **/usr/share/man/whatis** database is built from the HTML library using the **catman -w** command, section 3 is equivalent to section 2 or 3. See the **man** command for further explanation of sections.

Flag

Item	Description
-M *PathName*	Specifies an alternative search path. The search path is specified by the *PathName* parameter, and is a colon-separated list of directories.

Examples

1. To find the manual sections that contain the word `password` in their titles, enter:

```
apropos password
```

2. To find the manual sections that contain the word `editor` in their titles, enter:

```
apropos editor
```

File

Item	Description
/usr/share/man/whatis	Contains the **whatis** database.

ar Command

Purpose

Maintains the indexed libraries used by the linkage editor.

Syntax

ar [**-c**] [**-l**] [**-g** | **-o**] [**-s**] [**-v**] [**-C**] [**-T**] [**-z**] { **-h** | **-p** | **-t** | **-x** } [**-X** {**32**|**64**|**32_64**|**d64**| **any**}] *ArchiveFile* [*File* ...]

ar [**-c**] [**-l**] [**-g** | **-o**] [**-s**] [**-v**] [**-C**] [**-T**] [**-z**] { **-m** | **-r** [**-u**] } [{ **-a** | **-b** | **-i** } *PositionName*] [**-X** {**32**|**64**|**32_64**|**d64**|**any**}] *ArchiveFile File* ...

ar [**-c**] [**-l**] [**-g** | **-o**] [**-s**] [**-v**] [**-C**] [**-T**] [**-z**] { **-d** | **-q** } [**-X** {**32**|**64**|**32_64**|**d64**|**any**}] *ArchiveFile File* ...

ar [**-c**] [**-l**] [**-v**] [**-C**] [**-T**] [**-z**] { **-g** | **-o** | **-s** | **-w** } [**-X** {**32**|**64**|**32_64**|**d64**|**any**}] *ArchiveFile*

Description

The **ar** command maintains the indexed libraries used by the linkage editor. The **ar** command combines one or more named files into a single archive file written in **ar** archive format. When the **ar** command creates a library, it creates headers in a transportable format; when it creates or updates a library, it rebuilds the symbol table. See the **ar** file format entry for information on the format and structure of indexed archives and symbol tables.

There are two file formats that the **ar** command recognizes. The Big Archive Format, **ar_big**, is the default file format and supports both 32-bit and 64-bit object files. The Small Archive Format can be used to create archives that are recognized on versions older than AIX 4.3, see the **-g** flag. If a 64-bit object is added to a small format archive, **ar** first converts it to the big format, unless **-g** is specified. By default, **ar**

only handles 32-bit object files; any 64-bit object files in an archive are silently ignored. To change this behavior, use the **-X** flag or set the **OBJECT_MODE** environment variable.

Flags

In an **ar** command, you can specify any number of optional flags from the set **cClosTv**. You must specify one flag from the set of flags **dhmopqrstwx**. If you select the **-m** or **-r** flag, you may also specify a positioning flag (**-a**, **-b**, or **-i**); for the **-a**, **-b**, or **-i** flags, you must also specify the name of a file within *ArchiveFile* (*PositionName*), immediately following the flag list and separated from it by a blank.

Item	Description
-a *PositionName*	Positions the named files after the existing file identified by the *PositionName* parameter.
-b *PositionName*	Positions the named files before the existing file identified by the *PositionName* parameter.
-c	Suppresses the normal message that is produced when *library* is created.
-C	Prevents extracted files from replacing like-named files in the file system.
-d	Deletes the named files from the library.
-g	Orders the members of the archive to ensure maximum loader efficiency with a minimum amount of unused space. In almost all cases, the **-g** flag physically positions the archive members in the order in which they are logically linked. The resulting archive is always written in the small format, so this flag can be used to convert a big-format archive to a small-format archive. Archives that contain 64-bit **XCOFF** objects cannot be created in or converted to the small format.
-h	Sets the modification times in the member headers of the named files to the current date and time. If you do not specify any file names, the **ar** command sets the time stamps of all member headers. This flag cannot be used with the **-z** flag.
-i *PositionName*	Positions the named files before the existing file identified by the *PositionName* parameter (same as the **-b**).
-l	Places temporary files in the current (local) directory instead of the **TMPDIR** directory (by default **/tmp**).
-m	Moves the named files to some other position in the library. By default, it moves the named files to the end of the library. Use a positioning flag (**abi**) to specify some other position.
-o	Orders the members of the archive to ensure maximum loader efficiency with a minimum amount of unused space. In almost all cases, the **-o** flag physically positions the archive members in the order in which they are logically linked. The resulting archive is always written in the big archive format, so this flag can be used to convert a small-format archive to a big-format archive.
-p	Writes to standard output the contents of the named in the *Files* parameter, or all files specified in the *ArchiveFile* parameter if you do not specify any files.
-q	Adds the named files to the end of the library. In addition, if you name the same file twice, it may be put in the library twice.

Item	Description
-r	Replaces a named file if it already appears in the library. Because the named files occupy the same position in the library as the files they replace, a positioning flag does not have any additional effect. When used with the **-u** flag (update), the **-r** flag replaces only files modified since they were last added to the library file. If a named file does not already appear in the library, the **ar** command adds it. In this case, positioning flags do affect placement. If you do not specify a position, new files are placed at the end of the library. If you name the same file twice, it may be put in the library twice.
-s	Forces the regeneration of the library symbol table whether or not the **ar** command modifies the library contents. Use this flag to restore the library symbol table after using the **strip** command on the library.
-t	Writes to the standard output a table of contents for the library. If you specify file names, only those files appear. If you do not specify any files, the **-t** flag lists all files in the library.
-T	Allows file name truncation if the archive member name is longer than the file system supports. This option has no effect because the file system supports names equal in length to the maximum archive member name of 255 characters.
-u	Copies only files that have been changed since they were last copied (see the **-r** flag discussed previously).
-v	Writes to standard output a verbose file-by-file description of the making of the new library. When used with the **-t** flag, it gives a long listing similar to that of the **ls -l** command. When used with the **-x** flag, it precedes each file with a name. When used with the **-h** flag, it lists the member name and the updated modification times.
-w	Displays the archive symbol table. Each symbol is listed with the name of the file in which the symbol is defined.
-x	Extracts the named files by copying them into the current directory. These copies have the same name as the original files, which remain in the library. If you do not specify any files, the **-x** flag copies all files out of the library. This process does not alter the library.
-X *mode*	Specifies the type of object file **ar** should examine. The *mode* must be one of the following: **32** Processes only 32-bit object files **64** Processes only 64-bit object files **32_64** Processes both 32-bit and 64-bit object files **d64** Examines discontinued 64-bit XCOFF files (magic number == U803XTOCMAGIC). **any** Processes all of the supported object files. The default is to process 32-bit object files (ignore 64-bit objects). The *mode* can also be set with the **OBJECT_MODE** environment variable. For example, **OBJECT_MODE=64** causes **ar** to process any 64-bit objects and ignore 32-bit objects. The **-X** flag overrides the **OBJECT_MODE** variable.

Item	Description
-z	Creates a temporary copy of the archive and performs all requested modifications to the copy. When all operations have completed successfully, the working copy of the archive is copied over the original copy. This flag cannot be used with the **-h** flag.
ArchiveFile	Specifies an archive file name; required.
MemberName ...	Names of individual archive members.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To create a library, enter:

   ```
   ar -v -q lib.a strlen.o strcpy.o
   ```

   If the `lib.a` library does not exist, this command creates it and enters into it copies of the files `strlen.o` and `strcpy.o`. If the `lib.a` library does exist, then this command adds the new members to the end without checking for duplicate members. The **v** flag sets verbose mode, in which the **ar** command displays progress reports as it proceeds.
2. To list the table of contents of a library, enter:

   ```
   ar -v -t lib.a
   ```

   This command lists the table of contents of the `lib.a` library, displaying a long listing similar to the output of the **ls -l** command. To list only the member file names, omit the **-v** flag.
3. To replace or add new members to a library, enter:

   ```
   ar -v -r lib.a strlen.o strcat.o
   ```

   This command replaces the members `strlen.o` and `strcat.o`. If `lib.a` was created as shown in example 1, then the `strlen.o` member is replaced. A member named `strcat.o` does not already exist, so it is added to the end of the library.
4. To specify where to insert a new member, enter:

   ```
   ar -v -r -b strlen.o lib.a strcmp.o
   ```

   This command adds the `strcmp.o` file, placing the new member before the `strlen.o` member.
5. To update a member if it has been changed, enter:

   ```
   ar -v -r -u lib.a strcpy.o
   ```

   This command replaces the existing `strcpy.o` member, but only if the file `strcpy.o` has been modified since it was last added to the library.
6. To change the order of the library members, enter:

   ```
   ar -v -m -a strcmp.o lib.a strcat.o strcpy.o
   ```

This command moves the members `strcat.o` and `strcpy.o` to positions immediately after the `strcmp.o` member. The relative order of the `strcat.o` and `strcpy.o` members is preserved. In other words, if the `strcpy.o` member preceded the `strcat.o` member before the move, it still does.

7. To extract library members, enter:

```
ar -v -x lib.a strcat.o strcpy.o
```

This command copies the members `strcat.o` and `strcpy.o` into individual files named `strcat.o` and `strcpy.o`, respectively.

8. To extract and rename a member, enter:

```
ar -p lib.a strcpy.o >stringcopy.o
```

This command copies the member `strcpy.o` to a file named `stringcopy.o`.

9. To delete a member, enter:

```
ar -v -d lib.a strlen.o
```

This command deletes the member `strlen.o` from the `lib.a` library.

10. To create an archive library from multiple shared modules created with the **ld**command, enter:

```
ar -r -v libshr.a shrsub.o shrsub2.o shrsub3.o ...
```

This command creates an archive library named `libshr.a` from the shared modules named `shrsub.o`, `shrsub2.o`, `shrsub3.o`, and so on. To compile and link the `main` program using the `libshr.a` archive library, use the following command:

```
cc -o main main.c -L/u/sharedlib -lshr
```

The `main` program is now executable. Any symbols referenced by the `main` program that are contained by the `libshr.a` archive library have been marked for deferred resolution. The **-l** flag specifies that the `libshr.a` library be searched for the symbols.

11. To list the contents of **lib.a**, ignoring any 32-bit object file, enter:

```
ar -X64 -t -v lib.a
```

12. To extract all 32-bit object files from **lib.a**, enter:

```
ar -X32 -x lib.a
```

13. To list all files in **lib.a**, whether 32-bit, 64-bit, or non-objects, enter:

```
ar -X32_64 -t -v lib.a
```

File

Item	Description
/tmp/ar*	Contains temporary files.

arithmetic Command

Purpose

Tests arithmetic skills.

Syntax

arithmetic [**+**] [**-**] [**x**] [**/**] [*Range*]

Description

The **arithmetic** command displays simple arithmetic problems and waits for you to enter an answer. If your answer is correct, the program displays `Right!` and presents a new problem. If your answer is wrong, it displays `What?` and waits for another answer. After a set of 20 problems, the **arithmetic** command displays the number of correct and incorrect responses and the time required to answer.

The **arithmetic** command does not give the correct answers to the problems it displays. It provides practice rather than instruction in performing arithmetic calculations.

To quit the game, press the Interrupt (Ctrl-C) key sequence; the **arithmetic** command displays the final game statistics and exits.

Flags

The optional flags modify the action of the **arithmetic** command. These flags are:

Item	Description
+	Specifies addition problems.
-	Specifies subtraction problems.
x	Specifies multiplication problems.
/	Specifies division problems.
Range	A decimal number that specifies the permissible range of numbers. This range goes up to and includes 99. For addition and multiplication problems, the range applies to all numbers (except answers). For subtraction and division problems, the range applies only to the answers. At the start of the game, all numbers within this range are equally likely to appear. If you make a mistake, the numbers in the problem you missed become more likely to reappear.

If you do not select any flags, the **arithmetic** command selects addition and subtraction problems and a default range of 10. If you give more than one problem specifier (**+**, **-** ,**x**, **/**), the program mixes the specified types of problems in random order.

Examples

1. To drill on addition and subtraction of integers from 0 to 10:

```
arithmetic
```

2. To drill on addition, multiplication, and division of integers from 0 to 50:

```
arithmetic  +x/  50
```

File

Item	Description
/usr/games	Location of the system's games.

arp Command

Purpose

Displays and modifies address resolution, including ATM (Asynchronous Transfer Mode) interfaces.

Syntax

To Display ARP Entries

arp { [**-t** *ifType*] *HostName* | **-a** [**n**] [**/dev/kmem**] }

To Display ARP ATM Entries

arp { **-t atm** *HostName* | **-a** [**n**] [**/dev/kmem**] [**pvc** | **svc**] }

To Delete an ARP Entry

arp [**-t** *ifType*] **-d** *HostName*

To Delete a PVC ARP ATM Entry

arp -t atm -d pvc *vpi:vci* **if** *ifName*

To Create an ARP Entry

arp [**-t** *ifType*] **-s** *Type HostName AdapterAddress* [*Route*] [**temp**] [**pub**]

To Create an SVC ARP ATM Entry

arp -t atm -s *Type HostName AdapterAddress* [**temp**]

To Create a PVC ARP ATM Entry

arp -t atm -s *Type* **pvc** *vpi:vci* { *HostName* | **if** *ifName* } [**no-llc**] [**no-arp**] [**temp**]

To Import ARP Entries from Another File

arp [**-t** *ifType*] **-f** *FileName* [*Type*]

Description

The **arp** command displays and modifies the Internet-to-adapter address translation tables used by the **Address** in *Networks and communication management*. The **arp** command displays the current ARP entry for the host specified by the *HostName* variable. The host can be specified by name or number, using Internet dotted decimal notation.

Flags

Item	Description
-a	Used as { [**-t** *ifType*] *HostName* | **-a** [**n**] [**/dev/kmem**] } Displays all of the current ARP entries. Specify the **-a /dev/kmem** flag to display ARP information for kernel memory. The '**n**' modifier causes hostname lookups to be suppressed. Used as { **-t atm** *HostName* | **-a** [**n**] [**/dev/kmem**] [**pvc** | **svc**] } The **pvc** specification will display only ATM PVC (Permanent Virtual Circuits) types of virtual circuits, **svc** specification will display only ATM SVC (Switched Virtual Circuits) types of virtual circuits. If the **pvc** | **svc** parameter is omitted, all ATM virtual circuits will be displayed.

Item	Description
-d	Used as [**-t** *ifType*] **-d** *HostName* Deletes an entry for the host specified by the *HostName* variable if the user has root user authority. Used as **-t atm -d pvc** *vpi:vci* **if** *ifName* Deletes a PVC ARP entry by specifying *vpi:vci* rather than hostname. The *vpi:vc*i variables specify the virtual circuit that is to be deleted. The *ifname* variable specifies the name of the ATM interface on which the virtual circuit is to be deleted.
-f *FileName* [Type]	Causes the file specified by the *FileName* variable to be read and multiple entries to be set in the ARP tables. Entries in the file should be in the form: `[Type] HostName AdapterAddress [Route] [temp] [pub]` where **Type** Specifies the type of hardware address. If the address type is specified when invoking **arp** from the command line, it should not be specified in the file entries. Otherwise, it should be specified in each file entry. Valid hardware address types are: • `ether` for an Ethernet interface • `802.3` for an 802.3 interface • `fddi` for a Fiber Distributed Data interface • `802.5` for a Token-Ring interface • `hf` for a Host-Fabric interface **HostName** Specifies the remote host. **AdapterAddress** Specifies the hardware address of the adapter for this host as 6 hexadecimal bytes separated by colons. Use the **netstat -v** command to display the local hardware address. **Route** Specifies the route for a Token-Ring interface or Fiber Distributed Data Interface (FDDI) as defined in the Token-Ring or FDDI header. **temp** Specifies that this ARP table entry is temporary. The table entry is permanent if this argument is omitted. **pub** Specifies that this table entry is to be published, and that this system will act as an ARP server responding to requests for `HostName`, even though the host address is not its own. **Note:** The **-f** flag is not supported for ATM.

Item	Description
-s	Used as [**-t** *ifType*] **-s** *Type HostName AdapterAddress* [*Route*] [**temp**] [**pub**] Creates an ARP entry of the type specified by the *Type* variable for the host specified by the *HostName* variable with the adapter address specified by the *AdapterAddress* variable. Only users with root authority can use the **-s** flag. The adapter address is given as 6 hexadecimal bytes separated by colons. The line must be in the following format: `Type HostName AdapterAddress [Route] [temp] [pub]` where the *Type*, *HostName*, *AdapterAddress*, *Route*, **temp**, and **pub** parameters have the same purpose and definitions as the parameters for the **-f** flag. Used as **-t atm -s** *Type HostName AdapterAddress* [**temp**] Creates a SVC type of ARP entry for the remote host, specified by the *HostName* variable, with the ATM address specified by the *ATMAddress* variable. The ATM address is given as 20 hexadecimal bytes separated by colons. Creation of this entry causes this IP station to not use ARP server mechanism to resolve IP addresses. Used as **-t atm -s** *Type* **pvc** *vpi:vci* { *HostName* | **if** *ifName* } [**no-llc**] [**no-arp**] [**temp**] Creates a PVC type of ARP entry for the remote host, specified by the *HostName* variable, with the PVC specified by the *vpi:vci*. Either destination *Hostname* or the local *ifname* needs to be specified. The **no-llc** flag is used to indicate that LLC/SNAP encapsulation will not be used on this virtual circuit, in this case, the destination *Hostname* needs to be specified. The **no-arp** flag is used to indicate that ARP protocol will not be used on this virtual circuit, in this case, the destination *Hostname* needs to be specified. The *temp* parameter specifies that this ARP table entry is temporary, the table entry is permanent if this argument is omitted.
-t *ifType*	The **-t** *iftype* flag is used to indicate the type of Network interface. This flag is only required for the following interfaces: • at for ATM • ib for InfiniBand

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To add a single entry to the **arp** mapping tables until the next time the system is restarted, type:

```
arp -s 802.3 host2 0:dd:0:a:85:0 temp
```

2. To delete a map table entry for the specified host with the **arp** command, type:

```
arp   -d   host1 flag
```

3. To display arp entries for atm host `host1` , type:

```
arp -t atm -a host1
```

4. To add a PVC arp entry for atm host `host2`, type:

```
arp   -t atm   -s atm pvc 0:20 host2
```

5. To add a PVC arp entry for an interface `at0`, type:

```
arp   -t atm   -s atm pvc 0:20 if at0
```

artexdiff Command

Purpose

The **artexdiff** command compares the parameters and values between two profiles or between a profile and a system.

Syntax

artexdiff [**-a**] [**-q**|**-v**] [**-r**|**-n**] [**-u**|**-c**] [**-f** {**csv**|**xml**}] [**-g** *category*] [**-g** *level*] **profileA**

artexdiff [**-a**] [**-q**|**-v**] [**-r**|**-n**] [**-u**|**-c**] [[**-d**|**-s**] **-f txt**] [**-g** *category*] [**-g** *level*] **profileA**

artexdiff [**-a**] [**-q**|**-v**] [**-r**|**-n**] [**-p** [**-V** version] [**-m** *comment*]] [**-g** *category*] [**-g** *level*] **profileA**

artexdiff [**-a**] [**-q**|**-v**] [**-u**|**-c**] [**-f** {**csv**|**xml**}] [**-g** *category*] [**-g** *level*] ***profileA profile***

artexdiff [**-a**] [**-q**|**-v**][**-u**|**-c**] [[**-d**|**-s**] **-f txt**] [**-g** *category*] [**-g** *level*] ***profileA profile***

Description

The **artexdiff** command compares the parameters and values between profiles or between a profile and a system.

When the comparison is between a profile and a system, the current values of the parameters of the running system are compared. If the current value cannot be retrieved, then it compares with **nextboot** values. If **–n** option is specified, then the comparison uses the **nextboot** values for the systems with the parameters specified in the profile. If the **-r** option is specified, the current values are retrieved.

This command displays the output in three different formats to `stdout`. This output can be saved into a file using the redirector (>). If none of the output formats are specified, it displays in XML format. If Comma Separated Values (CSV) format (`-f csv`) is specified, then it displays in csv format, which can be used to open in a spreadsheet. If a text format (`-f txt`) is specified, the output will be in a table like readable format. When text format is specified, the output format can be either **diff** command output format (**-d** option) or **sdiff** command output format (**-s** option). So, the **-s** and **-d** flags can only be used in conjunction with the **-f** txt flag. When the **–p** option is specified, this command generates XML output in profile format that includes the parameters and values from the profile that are different from the system. Use the XML output in profile format to set the system by calling the **artexset** command. This ensures that the system is compliant with the input profile. When the **–p** option is specified, the output is always XML in profile format .

You can add comment and version number to the output profile if the **-p** option is specified. If you specify the **–m** option with a comment, the comment is included in the output profile. If you specify the **–V** option with a user revision number, the version number of the output profile is updated and the revision number is changed to the user-specified revision number. Otherwise, the revision number of the output profile version is set to 0.

Selection criteria, as specified by the **-u** or **-c** flags, indicate how to list the comparison results. When no selection criteria is specified, all comparison results display. If the **–c** option is specified, only parameters that are different in the comparison are displayed. If the **–u** option is specified, only the parameters that have the same values are displayed.

The specified profile can exist on the local file system using a relative or absolute path or on an LDAP server.

The **artexget -d** command creates a profile that has compressed attributes that belong to the same device but not a particular instance. The **artexdiff** command can compare this new profile to the system by searching the devices on the system for comparison. If the profile does not have the *setDiscover* attribute, the profile can still perform device discovery as compared to the system. The **artexdiff** command has the following limitations:

- If the compressed profile is compared to another profile except the system, the devices are not discovered even if the *setDiscover* parameter is set true. In this case, the contents of each profile are compared as is and the discovery operation is not performed.
- Also, if some of the instances and classes are populated in a profile, the instances and classes cause a parser error because the new profile is not a valid profile.

Flags

Item	Description
-a	Indicates that **artexdiff** output will be recorded in the AIX audit log.
-c	Indicates to output only the values found by the comparison that are found to be different. If neither **-u** nor **-c** is specified, all parameter values are noted in the output.
-d	Indicates to output the comparison results into a format like the **diff** command.
-f	Specifies the output formats. Possible formats include the following: • The *txt* option indicates to use plain text format. The flags **–d** and **–s** can be used only when this **-f** flag is set. • The *csv* option indicates to use comma-separated values format. • The *xml* option indicates to use xml format. This is the default format.
-g *categories*	Displays debug messages for the specified coma-separated list of categories. This option is useful while you write new catalog files. The available categories follow: • **ALL**: Includes all of the following categories. • **COMMANDS**: Prints information about the AIX command that is being run. • **DISCOVERY**: Prints information about the discovery commands that are being run. • **THREADS**: Prints information about threads that are being run within the framework. • **PARSING**: Prints information about the parsing of profile and catalog files. • **FLOW**: Prints information about the progress of the operation. **Note:** The default category is ALL.
-g *level*	Specifies the verbosity of the debug traces, as an integer in the range of 0 (no debug traces) - 3 (most verbose level). The default level is 0.

Item	Description
-m *comment*	Allows users to add comments to the profile. If the **-m** flag is used, the specified comment is added to the result profile. **Note:** This optional flag can only be used with the **-p** flag.
-n	Indicates to use the system's `nextboot` values for comparison. This option is only valid when the comparison includes a system.
-p	Generates XML output in profile format that includes the parameters and values from the profile that are different from the system. This option is valid only when the comparison is between a profile and a system.
-q	Allows users to ignore the nonfatal warning messages. The ignored messages are not displayed on the screen. This is an optional flag. **Note:** This flag cannot be used with the **-v** flag.
-r	Indicates to use the system's current values for comparison. This option is only valid when the comparison includes a system.
-s	Indicates to output the comparison results into a format like the **sdiff** command.
-u	Indicates to output only the values found by the comparison that are found to be identical. If neither **-u** nor **-c** is specified, all parameter values are noted in the output.
-v	Displays the warning and error messages generated by the AIX commands that are run during the processing of the **artexdiff** command. The messages are displayed on the `stderr`. This is an optional flag. **Note:** This flag cannot be used with the **-q** flag.
-V *version*	Sets the user revision number of the resulting profile. By default, the revision number of the resulting profile is set to 0. This is an optional flag. **Note:** This flag can only be used with the **-p** flag.

Parameters

Item	Description
profileA	Specifies the filename for the profile that lists the tunables by which all other information is gathered for comparison. A profile name of - (dash) can be specified for standard input.
profile	Specifies the filename for the profile to compare to the profile noted by the *profileA* parameter. If no profile is specified for the *profile* parameter, the comparison is performed against *profileA* and the system. A profile name of - (dash) can be specified for standard input.

Exit Status

Item	Description
`0`	The command completed successfully and no differences were found.
1	Differences were found.
>1	An error occurred.

Security

Access Control: This command should grant execute (x) access only to the root user and members of the security group. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the root user with the **setuid** (SUID) bit set.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand. To get the full functionality of the command, besides the **accessauths**, the role should also have the following authorizations:

- **aix.security.user.audit**
- **aix.security.role.assign**
- **aix.security.group.change**
- **aix.security.user.change**

Files Accessed:

Mode	File
rw	**/etc/passwd**
rw	**/etc/security/user**
rw	**/etc/security/user.roles**
rw	**/etc/security/limits**
rw	**/etc/security/environ**
rw	**/etc/group**
rw	**/etc/security/group**
r	**/usr/lib/security/artexdiff.default**
x	**/usr/lib/security/artexdiff.sys**

Auditing Events:

Event	Information
USER_Create	user

Examples

The following example illustrates how to compare the parameters and values between two profiles.

```
artexdiff profile1.xml profile2.xml
```

The following example illustrates how to compare the parameters and values between the `ldap_profile.xml` profile stored on LDAP server and the system.

```
artexdiff ldap://ldap_profile.xml
```

The following example illustrates how to create a new profile with the parameters and values from an input profile that are different from the system.

```
artexdiff -p profile.xml > diff_profile.xml
```

artexget Command

Purpose

The **artexget** command lists the configuration and tuning parameter information from a specified profile or from the system.

Syntax

artexget [**-v**] [**-d**] [**-p** | **-r** | **-n**] [**-l** {**dynamic** | **disruptive** | **reboot**}] [**-f** {**txt** |**csv** | **xml**}] [**-m** *comment*] [**-V** *version*] [**-g** *categories*] [**-g** *level*] *profile*

artexget [**-q**] [**-d**] [**-p** | **-r** | **-n**] [**-l** {**dynamic** | **disruptive** | **reboot**}] [**-f** {**txt** |**csv** | **xml**}] [**-m** *comment*] [**-V** *version*] [**-g** *categories*] [**-g** *level*] *profile*

Description

The **artexget** command lists the configuration and tuning parameter information from a profile or from the system. If none of the options **-p**, **-r**, or **-n** are specified, the command outputs the parameter and value pairs from the argument *profile*. If **-r** option is specified, the command outputs the current values of the parameters from the system. If the **-n** option is specified, the command outputs the values of the parameters after the next system restart. If **-p** option is specified, it outputs either current values of the parameters or values of the parameters after the next system restart, based on the `applyType` attribute value in the profile.

This command can also list the subset of the parameters based on selection criteria. If no selection criteria is specified, the command outputs a list of all parameters listed in the profile. If dynamic selection criteria (`-l dynamic`) is specified, then the command outputs a list of the parameters that do not require a reboot or disruptive action for the changes to take effect. The disruptive actions can be stopping and restarting a service or unmounting and mounting a file system. If disruptive selection criteria (`-l disruptive`) is specified, then the command outputs a list of parameters that need a disruptive action for the changes to take effect. If the selection criteria reboot (`-l reboot`) is specified, the command outputs a list of parameters that require a reboot for the changes to take effect.

This command displays the output in three different formats to `stdout`. This output can be saved into a file using the redirector (>). If none of the output formats are specified, the output displays in XML format. If Comma Separated Values (CSV) format (`-f csv`) is specified, then it displays in csv format, which can be used to open in a spreadsheet. If a text format (`-f txt`) is specified, the output is in a table like readable format.

A user comment and version can be added to the profile. If the **-m** option with a comment is specified, the comment is included in the output profile. If the **-V** option is specified with a user revision number, the version number of the output profile is updated and the revision number is changed to the user-specified revision number. Otherwise, the revision number of the output profile version number is incremented by 1.

The specified profile can exist on the local file system using a relative or absolute path or on a Lightweight Directory Access Protocol (LDAP) server.

Flags

Item	Description
-d	Creates a profile that sets all the instances of a parameter to the same value when used with the **-d** flag of the **artexset** command. The output profile contains only those parameters for which all instances would share the same value if the **-d** flag were not used; other parameters are removed from the profile.

Item	Description
-f	Specifies the output format. The **-f** flag has the following variables: • The *txt* variable specifies plain text format. • The *csv* variable specifies comma separated values format. • The *xml* format specifies xml format. This is the default format.
-g *categories*	Displays debug messages for the specified coma-separated list of categories. This option is useful while you write new catalog files. The available categories follow: • **ALL**: Includes all of the following categories. • **COMMANDS**: Prints information about the AIX command that is being run. • **DISCOVERY**: Prints information about the discovery commands that are being run. • **THREADS**: Prints information about threads that are being run within the framework. • **PARSING**: Prints information about the parsing of profile and catalog files. • **FLOW**: Prints information about the progress of the operation. **Note:** The default category is ALL.
-g *level*	Specifies the verbosity of the debug traces, as an integer in the range of 0 (no debug traces) - 3 (most verbose level). The default level is 0.
-l {*dynamic* | *disruptive* | *reboot*}	Indicates what tunable values to list in the output. The **-l** flag has the following options: • The *dynamic* variable indicates to list the tunable parameters for which the changes take effect immediately, without any condition. • The *disruptive* variable indicates to list the tunable parameters that require a disruptive operation such as an interruption of service or the recycling of a resource for the changes to take effect. • The *reboot* variable indicates to list the tunable parameters that require a system reboot for changes to take effect.
-m *comment*	Allows users to add comments to the profile. If the **-m** flag is used, the specified comment overwrites the previous comment. This is an optional flag.
-n	Lists the values of the parameters after the next system restart. If the **-p**, **-r**, or **-n** option is not specified, then list the tunable values described by the profile.
-p	Lists either the current values of the parameters or values of the parameters after the next system restart, based on the `applyType` attribute value in the .
-q	Allows users to ignore the nonfatal warning messages. The ignored messages are not displayed on the screen. This is an optional flag. **Note:** This flag cannot be used with the**-v** flag.
-r	Lists the current values on the running system.

Item	Description
-v	Displays the warning and error messages generated by the AIX commands that are run during the processing of the **artexget** command. The messages are displayed on the `stderr`. This is an optional flag. **Note:** This flag cannot be used with the **-q** flag.
-V *version*	Sets the user revision number of the resulting profile. By default, the user revision number of the entry profile is incremented. If the flag **-V** is used, the specified user revision number overwrites the existing revision number in the profile version number.

Parameters

Item	Description
profile	This is a mandatory file. The file specified includes a list of the tunable parameters. A profile name of - (dash) can be specified for standard input.

Exit Status

Item	Description
`0`	The command completed successfully
`>0`	An error occurred.

Security

Access Control: This command should grant execute (x) access only to the root user and members of the security group. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the root user with the **setuid** (SUID) bit set.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand. To get the full functionality of the command, besides the **accessauths**, the role should also have the following authorizations:

- **aix.security.user.audit**
- **aix.security.role.assign**
- **aix.security.group.change**
- **aix.security.user.change**

Files Accessed:

Mode	File
rw	**/etc/passwd**
rw	**/etc/security/user**
rw	**/etc/security/user.roles**
rw	**/etc/security/limits**
rw	**/etc/security/environ**
rw	**/etc/group**

Mode	File
rw	/etc/security/group
r	/usr/lib/security/artexget.default
x	/usr/lib/security/artexget.sys

Auditing Events:

Event	Information
USER_Create	user

Examples

The following example illustrates how to output the parameter and value pairs from the `profile1.xml` profile that is stored on a LDAP server.

```
artexget ldap://profile1.xml
```

The following example illustrates how to output the values of parameters after the next system restart from the system using the `local_profile.xml` profile.

```
artexget -n local_profile.xml
```

The following example illustrates how to output the current values of the parameters in text format from the system using the `local_profile.xml` profile.

```
artexget -r -f txt local_profile.xml
```

artexlist Command

Purpose

Outputs a list of profiles from the local system or LDAP server or outputs a list of catalogs that are installed on the local system.

Syntax

```
artexlist [-c | [-l] path][-q] [-g categories ] [-g level ]
```

Description

The command **artexlist** finds and lists the AIX Runtime Expert profiles on the local system or on LDAP server.

If the **-c** option is specified, the output returns a list of catalogs that are installed on the local system rather than a list of profiles.

By default, this command outputs a list of the profiles from `/etc/security/artex/samples` directory. To override the default path, set the environment variable ARTEX_PROFILE_PATH to one or more semicolon delimited paths. Otherwise, use the *path* argument. In addition to the local system profiles, use the **–l** option to list the profiles from the LDAP server.

Flags

Item	Description
-c	Indicates to list the catalogs installed on the local system in `/etc/security/artex/catalogs` directory.
-l	Indicates to list the profiles from the LDAP server.
-g *categories*	Displays debug messages for the specified coma-separated list of categories. This option is useful while you write new catalog files. The available categories follow: • **ALL**: Includes all of the following categories. • **COMMANDS**: Prints information about the AIX command that is being run. • **DISCOVERY**: Prints information about the discovery commands that are being run. • **THREADS**: Prints information about threads that are being run within the framework. • **PARSING**: Prints information about the parsing of profile and catalog files. • **FLOW**: Prints information about the progress of the operation. **Note:** The default category is ALL.
-g *level*	Specifies the verbosity of the debug traces, as an integer in the range of 0 (no debug traces) - 3 (most verbose level). The default level is 0.
path	Specifies the path on the local system that contains the list of profiles that are to be returned in the output.
-q	Allows users to ignore the nonfatal warning messages. The ignored messages are not displayed on the screen. This is an optional flag. **Note:** This flag cannot be used with the **-q** flag.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

Access Control: This command should grant execute (x) access only to the root user and members of the security group. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the root user with the **setuid** (SUID) bit set.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand. To get the full functionality of the command, besides the **accessauths**, the role should also have the following authorizations:

- **aix.security.user.audit**
- **aix.security.role.assign**
- **aix.security.group.change**
- **aix.security.user.change**

Files Accessed:

Mode	File
rw	**/etc/passwd**
rw	**/etc/security/user**
rw	**/etc/security/user.roles**
rw	**/etc/security/limits**
rw	**/etc/security/environ**
rw	**/etc/group**
rw	**/etc/security/group**
r	**/usr/lib/security/artexlist.default**
x	**/usr/lib/security/artexlist.sys**

Auditing Events:

Event	Information
USER_Create	user

Examples

The following example illustrates how to list the sample profiles from the default path `/etc/security/artex/samples`.

```
artexlist
```

The following example illustrates how to list the profiles using environment variable ARTEX_PROFILE_PATH.

```
export ARTEX_PROFILE_PATH="/tmp:/$HOME/profiles"
artexlist
```

The following example illustrates how to list the profiles from `/data/profiles` directory.

```
artexlist /data/profiles
```

The following example illustrates how to list the profiles from an LDAP server and from a local system.

```
artexlist -l
```

The following example illustrates how to list the catalogs installed on the system.

```
artexlist -c
```

artexmerge Command

Purpose

The **artexmerge** command merges two or more profiles.

Syntax

```
artexmerge [-q] [-v | -t] [-f] [-m {comment}] [-V {version}][-g categories] [-g level]
profile . . .
```

Description

The command **artexmerge** merges two or more profiles and displays the output to `stdout`. You can also save the output to a file using the redirector (>).

When merging the profiles, the command returns an error if a parameter exists in more than one profile, with different values. To override this error condition, use the **-f** option. The **-f** option indicates to use the parameter and value from the last profile listed in the command syntax.

The **artexmerge** command validates the parameters of the profiles specified to be merged. If the **–v** option is specified, the parameters for each profile specified are verified prior to the merge. If the **–t** option is specified, the parameters are verified in the merged profile, after the profiles are merged. These two options are mutually exclusive.

You can add comment and version number to the profile. If you specify the **–m** option with a comment, the comment is included in the output profile. If you specify the **–V** option with a user revision number, the version number of the output profile is updated and the revision number set to the user-specified revision number.

The specified profiles can exist on the local file system using a relative or absolute path or on an LDAP server.

Flags

Item	Description
-g *categories*	Displays debug messages for the specified coma-separated list of categories. This option is useful while you write new catalog files. The available categories follow: • **ALL**: Includes all of the following categories. • **COMMANDS**: Prints information about the AIX command that is being run. • **DISCOVERY**: Prints information about the discovery commands that are being run. • **THREADS**: Prints information about threads that are being run within the framework. • **PARSING**: Prints information about the parsing of profile and catalog files. • **FLOW**: Prints information about the progress of the operation. **Note:** The default category is ALL.
–g *level*	Specifies the verbosity of the debug traces, as an integer in the range of 0 (no debug traces) - 3 (most verbose level). The default level is 0.
-q	Allows users to ignore the nonfatal warning messages. The ignored messages are not displayed on the screen. This is an optional flag. **Note:** This flag cannot be used with the **-v** flag.
-v	Displays the warning and error messages generated by the AIX commands that are run during the processing of the **artexmerge** command. The messages are displayed on the `stderr`. This is an optional flag. **Note:** This flag cannot be used with the **-q** flag.
-t	Indicates to verify the parameters in the merged profile, rather than prior to the merge.
-f	Indicates to force the merge. If two or more profiles contain the same parameter with different values, indicates to use the value of the parameter included in the last profile.

Item	Description
-m {*comment* }	Allows users to add comments to the profile. If the **-m** flag is used, the specified comment is added to the resulting profile.
-V {*version*}	Sets the user revision number of the resulting profile. By default, the revision number of the resulting profile is set to 0. This is an optional flag.

Parameters

Item	Description
profile . . .	Lists the filenames of the profiles to merge, separated by a space. For example, `profileA profileB profileC`.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

Access Control: This command should grant execute (x) access only to the root user and members of the security group. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the root user with the **setuid** (SUID) bit set.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand. To get the full functionality of the command, besides the **accessauths**, the role should also have the following authorizations:

- **aix.security.user.audit**
- **aix.security.role.assign**
- **aix.security.group.change**
- **aix.security.user.change**

Files Accessed:

Mode	File
rw	**/etc/passwd**
rw	**/etc/security/user**
rw	**/etc/security/user.roles**
rw	**/etc/security/limits**
rw	**/etc/security/environ**
rw	**/etc/group**
rw	**/etc/security/group**
r	**/usr/lib/security/artexmerge.default**
x	**/usr/lib/security/artexmerge.sys**

Auditing Events:

Event	Information
USER_Create	user

Examples

The following example illustrates how to combine profiles located on a LDAP server and on a local file system.

```
artexmerge /tmp/no_profile1.xml ldap://ldap_raso_profile.xml /data/nfs_profile.xml
```

The following example illustrates how to combine two profiles with duplicate parameters and save as `merged_profile.xml`.

```
artexmerge –f profile1.xml profile2.xml > merged_profile.xml
```

artexremset Command

Purpose

artexremset command executes **artexset** command on one or more remote systems.

Syntax

```
artexremset [ [ [ [ [-q] [-c] [-r] [-R] ] | -t | -p] [-l {dynamic | noreboot | reboot |
all} ] ]
| -b | -x | -u] [-L] [-D] profile {clientname | nim_mac_group}

artexremset [-q] [-c] [-r] [-R] [-l {dynamic | noreboot | reboot | all} ] [-L] [-D]
profile {clientname | nim_mac_group}

artexremset [-l {dynamic | noreboot | reboot | all} ] -t [-L] [-D] profile {clientname |
nim_mac_group}

artexremset [-l {dynamic | noreboot | reboot | all} ] -p [-L] [-D] profile {clientname |
nim_mac_group}

artexremset -b [-L] [-D] profile {clientname | nim_mac_group}

artexremset -x [-D] {clientname | nim_mac_group}

artexremset -u [-D] {clientname | nim_mac_group}
```

Description

artexremset provides the ability to execute **artexset** commands on each client with a designated profile provided by the server or a profile stored on an LDAP server. Therefore, all the command options designated for the local **artexset** command must also be provided by the server so these options can be directly conveyed to each client's local **artexset** command.

The **artexremset** command runs only on NIM master. When the profile is on NIM master, the **artexremset** command copies the profile to a remote client machine prior to requesting the client to execute **artexset** command. When the **–L** option is specified, the profile name given is assumed to be the pathname to a profile that exists in LDAP. Thus, no profile is copied to the client from NIM master. Instead, the LDAP pathname is packaged in the custom script file and the local **artexset** command should realize that the `ldap://` prefix represents an LDAP file.

By default, the exit status of the **artexremset** command will be a cumulative "OR" of all the remote **artexset** commands. With the **–D** option, the results of each individual NIM command result is captured and associated with each individual node and listed in a `stdout` listing.

Flags

Item	Description
-q	Indicates to ignore non-fatal warning messages.
-c	Indicates to verify that the **artexset** command set the values and that they were successfully applied to the system.
-r	If the **-c** option indicates that not all parameters were applied successfully, the **-r** option indicates to rollback the parameter values for the specified *profile* to their original state. To do this, the command applies the values stored in the `latest_rollback.xml` file.
-l { *dynamic* | *noreboot* | *reboot* | *all*}	Indicates the level to which to apply the command. The **-l** flag has the following options: • The *dynamic* variable indicates to apply non-disruptive parameters only. • The *noreboot* variable indicates to apply all parameters that do not need a reboot, and recycle the resources as needed. • The *reboot* variable indicates to apply only the parameters that have reboot constraint • The *all* variable indicates to apply all parameters, including the ones that need reboot.
-R	Specifies to not create a rollback profile.
-b	Indicates to enable the master profile, which is also referred to as the boot profile.
-x	Indicates to disable the master profile, which is also referred to as the boot profile. This flag is the opposite of the **-b** option. If the -x option is specified, no profile parameter is required.
-t	Indicates to test if the values listed in the *profile* are valid tunables, as recognized by the runtime system.
-p	Generates XML output that includes the parameters and values from the profile that are different from the system. This option is valid only when the comparison is between a profile and a system.
-u	Indicates to rollback the parameter values of the last applied profile, as they were prior to issuing the last **artexset** command. To do this, the command applies the values stored in the `/etc/security/artex/latest_rollback.xml` file. If the **-u** option is specified, no profile parameter is required.
-L	Instructs each individual AIX Runtime Expert client to download the profile from an LDAP repository designated by the profile string.
-D	Indicates to output the results of the remote **artexset** command associated with each individual node.

Parameters

Item	Description
profile	This is a mandatory file, except when the **-x** or **-u** option is specified. The file specified includes a list of the tunable parameters.
clientname | nim_mac_group	The name of the client node or pre-defined NIM machine groups.

Exit Status

Individual error messages from the resulting NIM commands are masked unless the **-D** option is used. A cumulative return value that consists of an “OR” of all the individual nodes or groups is retuned by the **artexremset** command.

Item	Description
0	The command completed successfully
>0	An error occurred.

Examples

The following example illustrates how to execute the **artexset** command on a client machine, using a profile located on NIM master.

```
artexremset nim_profile.xml client1
```

The following example illustrates how to execute the **artexset** command on multiple client machines, using a profile located on LDAP server.

```
artexremset –L ldap://profile1.xml client1 mac_group1 client2
```

The following example illustrates how to output the results of the remote **artexset** command associated with each individual client machine.

```
artexremset –D profile1.xml client1 client2
```

artexset Command

Purpose

The **artexset** command applies an AIX Runtime Expert profile to a system. The profile contains values for parameters that are to be set on the system.

Syntax

artexset [**-c**][**-d**][**-r**][**-R**][**-F**][**-l** {**dynamic**|**noreboot**|**reboot**|**all**}][|**-v**][**-g** *categories*][**-g** *level*] *profile*

artexset **-u** [**-q**|**-v**][**-g** *categories*][**-g** *level*]

artexset **-t** [**-q**|**-v**][**-g** *categories*] [**-g** *level*] *profile*

artexset **-p** [**-F**][**-l** {**dynamic**|**noreboot**|**reboot**|**all**}][**-q**|**-v**][**-g** *categories*][**-g** *level*] *profile*

artexset **-b** [**-q**|**-v**][**-g** *categories*][**-g** *level*] *profile*

artexset **-x** [**-q**|**-v**][**-g** *categories*][**-g** *level*] *profile*

Description

The **artexset** command applies an AIX Runtime Expert profile to a system. The profile contains values for parameters that are to be set on the system. This command also allows you to verify the accuracy of setting the parameters for a profile, preview the parameters that the command changes, enable and disable the ability to set the profile parameters during boot time, and rollback to a previous profile.

When the **–t** option is specified, the command tests the correctness of the profile. The command checks whether the profile has the correct XML format. Also, it checks whether the parameters defined in the profile are valid and supported by AIX Runtime Expert.

When the **–p** option is specified, the parameters for the profile are not set but rather the parameters that would change are identified. Only the parameter values that would change are listed in the output. For example, if the parameter value on the system is same as the parameter value in the profile, the parameter would not be listed in the output since the parameter value is not affected by the command.

By default, this command creates a rollback profile. The rollback profile allows you to undo a profile change if needed. If the **–R** option is specified, the command does not create a rollback profile.

If you want to rollback to a previous state, use the **-u** option. One level of rollback is supported. For example, after a rollback is complete, you cannot perform another subsequent rollback until **artexset** is run again to set the parameters.

When **–b** option specified, the parameters are set during each system boot. This option can be disabled by using the **-x** option.

With the **-l** option, you can set a subset of the parameters that are noted in the profile. If the **-l** option is not specified, all parameters listed in the profile are applied only if none of the parameters require a reboot. If dynamic selection criteria (`-l dynamic`) is specified, all parameters that do not require a reboot, disruptive action, like stopping and restarting a service, or unmounting and mounting a file system are set. If `noreboot` selection criteria (`-l noreboot`) is specified, all parameters that do not need a reboot are set. If the selection criteria reboot (`-l reboot`) is specified, all parameters that require a reboot are set. If the selection criteria all (`-l all`) is specified, then all parameters are set.

The specified profile can be on the local file system using a relative or absolute path or on an LDAP server.

Flags

Item	Description
-g *categories*	Displays debug messages for the specified coma-separated list of categories. This option is useful while you write new catalog files. The available categories follow: • **ALL**: Includes all of the following categories. • **COMMANDS**: Prints information about the AIX command that is being run. • **DISCOVERY**: Prints information about the discovery commands that are being run. • **THREADS**: Prints information about threads that are being run within the framework. • **PARSING**: Prints information about the parsing of profile and catalog files. • **FLOW**: Prints information about the progress of the operation. **Note:** The default category is ALL.
-g *level*	Specifies the verbosity of the debug traces, as an integer in the range of 0 (no debug traces) - 3 (most verbose level). The default level is 0.
-q	Indicates to ignore non-fatal warning messages.
-c	Indicates to verify that the command set the values and that they were successfully applied to the system. If they were not successfully applied, then the **artexset** operation is aborted.
-r	Indicates to rollback if a failure occurs.

Item	Description
-l {**dynamic**|**noreboot**|**reboot**|**all**}	Specifies the level to which to apply the parameters. The **-l** flag has the following options: • The *dynamic* variable indicates to apply non-disruptive parameters only. • The *noreboot* variable indicates to apply all parameters that do not need a reboot. • The *reboot* variable indicates to apply only the parameters that have a reboot constraint. • The *all* variable indicates to apply all parameters, including the ones that need a reboot.
-R	Specifies to not create a rollback profile.
-b	Indicates to enable the master profile, which is also referred to as the boot profile.
-x	Indicates to disable the master profile, which is also referred to as the boot profile. This flag is the opposite of the **-b** option. If the -x option is specified, no profile parameter is required.
-t	Indicates to test if the values listed in the *profile* are valid tunables, as recognized by the runtime system.
-p	Specifies to preview setting the parameters but does not set the parameters for the *profile*. This flag identifies which parameters would change as a result of issuing this command. The output lists what parameters would change, what services would restart, and whether the system would need to restart, if the profile is applied. Only the parameter values that would change are listed in the output. For example, if the parameter value on the system is same as the parameter value in the profile, the parameter would not be listed in the output since the parameter value is not affected by the command.
-u	Indicates to rollback the parameter values of the last applied profile, as they were prior to issuing the last **artexset** command. To do this, the command applies the values stored in the `/etc/security/artex/latest_rollback.xml` profile. If the **-u** option is specified, no profile parameter is required.
-d	Allows the discover method to run prior to the set operation. This flag sets all instances of parameters that have the `setDiscover` attribute to the same value. This flag is optional.
-v	Displays the warning and error messages generated by the AIX commands that are run during the processing of the **artexset** command. **Note:** This optional flag cannot be used with the **-q** flag.
-F	Sets values for all parameters, even if the parameter is already set to the required value. **Note:** This flag is optional.

Parameters

Item	Description
profile	This is a mandatory file, except when the **-x** or **-u** option is specified. The specified file includes a list of the tunable parameters. A profile name of dash (-) can be specified for standard input.

Exit Status

Item	Description
`0`	The command completed successfully
`>0`	An error occurred.

Security

Access Control: This command should grant execute (x) access only to the root user and members of the security group. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the root user with the **setuid** (SUID) bit set.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand. To get the full functionality of the command, besides the **accessauths**, the role should also have the following authorizations:

- **aix.security.user.audit**
- **aix.security.role.assign**
- **aix.security.group.change**
- **aix.security.user.change**

Files Accessed:

Mode	File
rw	**/etc/passwd**
rw	**/etc/security/user**
rw	**/etc/security/user.roles**
rw	**/etc/security/limits**
rw	**/etc/security/environ**
rw	**/etc/group**
rw	**/etc/security/group**
r	**/usr/lib/security/artexset.default**
x	**/usr/lib/security/artexset.sys**

Auditing Events:

Event	Information
USER_Create	user

Examples

The following example illustrates how to set all parameters defined in the profile `local_profile.xml`.

```
artexset -l all local_profile.xml
```

The following example illustrates how to check the correctness of the `ldap_profile.xml` profile stored on an LDAP server.

```
artexset -t ldap://ldap_profile.xml
```

The following example illustrates how to enable applying the profile `/tmp/boot_profile.xml` at every system restart.

```
artexset -b /tmp/boot_profile.xml
```

The following example illustrates how to disable applying a profile at every system restart.

```
artexset -x
```

The following example illustrates how to roll back the parameters to the values prior to previous issue of the **artexset** command.

```
artexset -u
```

as Command

Purpose

Reads and assembles a source file.

Syntax

as [**-a** *Option[:Option]*] [**-o** *ObjectFile*] [**-n** *Name*] [**-u**] [**-l** [*ListFile*]] [**-W** | **-w**] [**-x** [*XCrossFile*]] [**-s** [*ListFile*]] [**-m** *ModeName*][**-M**][**-E***off*|*on*] [**-p** *off*|*on*] [**-i**] [**-v**] [*File*]

Description

The **as** command reads and assembles the named *File* (by convention, this file ends with a **.s** suffix). If you do not specify a *File*, the **as** command reads and assembles standard input. It stores its output, by default, in a file named **a.out**. The output is stored in the **XCOFF** file format.

All flags for the **as** command are optional.

Flags

-a *Option[:Option]*
Specifies the mode in which the **as** command operates. By default, the **as** command operates in 32-bit mode, but the mode can be explicitly set by using the flag **-a32** for 32-bit mode operation or **-a64** for 64-bit mode operation.

You can specify multiple options with the **-a** parameter. If you specify conflicting flags, the succeeding flags override the preceding flags. You can specify multiple options that are separated by colons, with the **-a** parameter, as shown in the following example:

```
-a 64:align-prefixed-csect=no
```

align-prefixed-csect=<yes/no>
Specifies whether the alignment of a control section (csect) that contains prefixed instructions will be increased to at least a 64-byte boundary, if necessary. 64-byte is the minimum alignment which

ensures that the prefixed instructions are aligned properly when assembled programs are linked. If you specify **yes** for the **align-prefixed-csect** option, the alignment of csects that contain the prefixed instructions is increased, if necessary. If you specify **no** for the **align-prefixed-csect** option and the **-w** flag is used on the command line, a warning message is displayed if a prefixed instruction is part of a csect with an alignment which is not strict enough. For more information, see the .align pseudo-op.

align-prefixed-op=<yes/no>

Specifies whether a prefixed instruction is aligned by preceding the instruction with a no-op instruction when the prefixed instruction would cross a 64-byte boundary. If you specify **yes** for the **align-prefixed-op** option, prefixed instructions are aligned, if required. If you specify **no** for the **align-prefixed-op** option and the **-w** flag is used on the command line, a warning message is displayed if a prefixed instruction crosses a 64-byte boundary. For more information, see the .align pseudo-op.

-l[*ListFile*]

Produces an assembler listing. If you do not specify a file name, a default name is produced by replacing the suffix extension of the source file name with a **.lst** extension. By convention, the source file suffix is a **.s**. For example:

```
sourcefile.xyz
```

produces a default name of:

```
sourcefile.lst
```

If the source code is from standard input and the **-l** flag is used without specifying an assembler-listing file name, the listing file name is **a.lst**.

-m *ModeName*

Indicates the assembly mode. This flag has lower priority than the **.machine** pseudo-op.

If this flag is not used and no **.machine** pseudo-op is present in the source program, the default assembly mode is used. The default assembly mode has the POWER® family/PowerPC® intersection as the target environment, but treats all POWER family/PowerPC incompatibility errors (including instructions outside the POWER family/PowerPC intersection and invalid form errors) as instructional warnings.

If an assembly mode that is not valid is specified and no **.machine** pseudo-op is present in the source program, an error is reported and the default assembly mode is used for instruction validation in pass 1 of the assembler.

If the **-m** flag is used, the *ModeName* variable can specify one of the following values:

""

Explicitly specifies the default assembly mode that has the POWER family/PowerPC intersection as the target environment, but treats instructions outside the POWER family/PowerPC intersection and invalid form errors as instructional warnings. A space is required between **-m** and the null string argument (two double quotation marks).

com

Specifies the POWER family/PowerPC intersection mode. A source program can only contain instructions that are common to both POWER family and PowerPC; any other instruction causes an error. Any instruction with an invalid form causes errors, terminates the assembly process, and results in no object code being generated.

Note: Certain POWER family instructions are supported by the PowerPC 601 RISC Microprocessor, but do not conform to the PowerPC architecture. These instructions cause errors when using the **com** assembly mode.

any
Specifies the indiscriminate mode. The assembler generates object code for any recognized instruction, regardless of architecture. This mode is used primarily for operating system development and for testing and debugging purposes.

Note: All POWER family and PowerPC incompatibility errors are ignored when using the **any** assembly mode, and no warnings are generated.

ppc
Specifies the PowerPC64-bit mode. A source program can only contain PowerPC instructions. Any other instruction causes an error.

Note:

1. The PowerPC optional instructions are not implemented in every PowerPC processor and do not belong to the **ppc** mode. These instructions generate an error if they appear in a source program that is assembled using the **ppc** assembly mode.
2. Certain instructions conform to the PowerPC architecture, but are not supported by the PowerPC 601 RISC Microprocessor.

ppc64
Specifies the PowerPC64-bit mode. A source program can contain 64-bit PowerPC instructions.

pwr
Specifies the POWER mode. A source program can only contain instructions that are valid for the POWER implementation of the POWER architecture.

pwr2 or pwrx
Specifies the POWER2 mode. A source program can only contain instructions that are valid for the POWER2 implementation of the POWER architecture. **pwr2** is the preferred value. The alternate assembly mode value **pwrx** means the same thing as **pwr2**.

Note: The POWER implementation instruction set is a subset of the POWER2 implementation instruction set.

pwr4 or 620
Specifies the PowerPC64 mode. A source program can only contain instructions that are valid for POWER4 compatible processors.

601
Specifies the PowerPC 601 RISC Microprocessor mode. A source program can only contain instructions that are valid for the PowerPC 601 RISC Microprocessor.

The PowerPC 601 RISC Microprocessor design was completed before the POWER processor-based platform. Some PowerPC instructions are not be supported by the PowerPC 601 RISC Microprocessor.

Attention: The PowerPC 601 RISC Microprocessor implements the POWER Architecture plus some POWER family instructions that are not included in the PowerPC architecture. This allows existing POWER applications to run with acceptable performance on PowerPC processor-based systems.

The PowerPC 601 RISC Microprocessor implements the POWER processor-based platform plus some POWER family instructions are not included in the POWER processor-based platform. This allows existing POWER applications to run with acceptable performance on POWER processor-based systems.

603
Specifies the PowerPC 603 RISC Microprocessor mode. A source program can only contain instructions that are valid for the PowerPC 603 RISC Microprocessor.

604
Specifies the PowerPC 604 RISC Microprocessor mode. A source program can only contain instructions that are valid for the PowerPC 604 RISC Microprocessor.

ppc970 or 970
Specifies the PowerPC 970 mode. A source program can only contain instructions that are valid for PowerPC 970 compatible processors.

A35
Specifies the A35 mode. A source program can only contain instructions that are valid for the A35.

pwr5
Specifies the POWER5 mode. A source program can only contain instructions that are valid for POWER5 compatible processors.

pwr5x
Specifies the POWER5+ mode. A source program can only contain instructions that are valid for POWER5+ compatible processors.

pwr6
Specifies the POWER6 mode. A source program can only contain instructions that are valid for POWER6 compatible processors.

pwr6e
Specifies the POWER6+ mode. A source program can only contain instructions that are valid for POWER6+ compatible processors.

pwr7
Specifies the POWER7 mode. A source program can only contain instructions that are valid for POWER7 compatible processors.

pwr8
Specifies the POWER8 mode. A source program can only contain instructions that are valid for POWER8 compatible processors.

pwr9
Specifies the POWER9™ mode. A source program can only contain instructions that are valid for POWER9 compatible processors.

pwr10
Specifies the POWER10 mode. A source program can only contain instructions that are valid for POWER10 compatible processors.

-M
Lists the assembly modes that are valid for instructions listed in the input file or list instructions that are valid for the specified assembly mode.

When used with the **-m** flag, the assembler lists all the instructions that are valid in the assembly mode specified with the **-m** flag. Any other flags specified on the command line must be valid, but they are ignored. The input file is also ignored.

When used without the **-m** flag, the assembler reads lines from the specified input file, or from standard input if no input file was specified. Any other flags specified on the command line must be valid, but they are ignored. If a line of input begins with a valid instruction mnemonic, the assembler prints all the assembly modes for which the instruction is valid. If a line begins with a label, the label is removed before the line is checked for a valid instruction. Lines that do not begin with a valid instruction are ignored. Most valid assembler source files can be used as the input file when the **-M** flag is used, as long as instruction mnemonics are separated from operands by white space.

Note: The assembler does not generate an object file when the **-M** flag is used.

-n *Name*
Specifies the name that appears in the header of the assembler listing. By default, the header contains the name of the assembler source file.

-o *ObjectFile*
Writes the output of the assembly process to the specified file instead of to the **a.out** file.

-s[*ListFile*]
Indicates whether or not a mnemonics cross-reference for POWER family and PowerPC is included in the assembler listing. If this flag is omitted, no mnemonics cross-reference is produced. If this flag

is used, the assembler listing will have POWER family mnemonics if the source contains PowerPC mnemonics, and will have PowerPC mnemonics if the source contains POWER family mnemonics.

The mnemonics cross-reference is restricted to instructions that have different mnemonics in the POWER family and PowerPC, but that have the same op code, function, and input operand format.

Because the **-s** flag is used to change the assembler-listing format, it implies the **-l** flag. If both option flags are used and different assembler-listing file names (specified by the *ListFile* variable) are given, the listing file name specified by the *ListFile* variable used with the **-l** flag is used. If an assembler-listing file name is not specified with either the **-l** or **-s** flag, a default assembler listing file name is produced by replacing the suffix extension of the source file name with a **.lst** extension.

-u
Accepts an undefined symbol as an extern so that an error message is not displayed. Otherwise, undefined symbols are flagged with error messages.

-W
Turns off all warning message reporting, including the instructional warning messages (the POWER family and PowerPC incompatibility warnings).

-w
Turns on warning message reporting, including reporting of instructional warning messages (the POWER family and PowerPC incompatibility warnings).

Note: When neither **-W** nor **-w** is specified, the instructional warnings are reported, but other warnings are suppressed.

-x[*XCrossFile*]
Produces cross-reference output. If you do not specify a file name, a default name is produced by replacing the suffix extension of the source file name with a **.xref** extension. Conventionally, the suffix is a **.s**. For example:

```
sourcefile.xyz
```

produces a default name of:

```
sourcefile.xref
```

Note: The assembler does not generate an object file when the **-x** flag is used.

-E
Specifies whether to report errors due to the new v2.00 syntax (`-Eon`), or to ignore them (`-Eoff`). By default, v2.00 errors are ignored.

-p
Specifies whether to use new v2.00 branch prediction (`-pon`), or pre-v2.00 branch prediction (`-poff`). By default, pre-v2.00 branch prediction is used.

-i
Specifies that branch prediction suffixes are to be encoded. By default, this option is not set. This option will be ignored if the **-p** option is specified.

-v
Displays the version number of this command.

File
Specifies the source file. If no file is specified, the source code is taken from standard input.

Environment Variable

OBJECT_MODE
The assembler respects the setting of the OBJECT_MODE environment variable. If neither **-a32** or **-a64** is used, the environment is examined for this variable. If the value of the variable is anything other than the values listed in the following table, an error message is generated and the assembler

exits with a nonzero return code. The implied behavior corresponding to the valid settings are as follows:

OBJECT_MODE = 32
Produce 32-bit object code. The default machine setting is **com**.

OBJECT_MODE = 64
Produce 64-bit object code (XCOFF64 files). The default machine setting is **ppc64**.

OBJECT_MODE = 32_64
Invalid.

OBJECT_MODE = *anything else*
Invalid.

Examples

1. To produce a listing file named `file.lst` and an object file named `file.o`, enter:

```
as -l -o file.o file.s
```

2. To produce an object file named **file.o** that will run on the 601 processor and generate a cross-reference for POWER family and PowerPC mnemonics in an assembler listing file named **file.lst**, enter:

```
as -s -m 601 -o file.o file.s
```

3. To produce an object file named **file.o** using the default assembly mode and an assembler listing file named **xxx.lst** with no mnemonics cross-reference, enter:

```
as -lxxx.lst -o file.o file.s
```

Files

/usr/ccs/bin/as
Contains the **as** command

a.out
The default output file.

aso Command

Purpose

Starts the active system optimizer (ASO) outside of the SRC.

Syntax

aso

Description

The ASO is an AIX service, which monitors and dynamically optimizes the system. It is provided as an SRC subsystem, and can be started and stopped by the usual SRC commands, such as the **startsrc** and **stopsrc** commands.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Environment Variables

Item	Description
ASO_ENABLED	**Purpose** When set for a process, this environment variable can be used either to ensure that the process is not optimized by ASO or to increase the probability of the process being optimized. **Values** • **ALWAYS**: The ASO prioritizes this process for optimization. • **NEVER**: The ASO never optimizes this process. • Any other value: The ASO optimizes the process if it fulfills the optimization criteria for the ASO. **Change** `ASO_ENABLED=[ALWAYS|NEVER] export ASO_ENABLED` This change affects processes, which are running from the current shell after you set the variable. The change is effective until logging out of this shell. A permanent change can be made by adding the ASO_ENABLED=[ALWAYS|NEVER] option to the **/etc/environment** file.

Item	Description
ASO_OPTIONS	**Purpose** When set for a process, this environment variable can be used to control which optimizations that ASO might apply to that process. Multiple options that are separated by comma character be specified. When multiple options conflict, only the last setting takes effect. **Values** • **ALL**=[**ON**|**OFF**]: Enables or disables all optimizations for this process. • **CACHE_AFFINITY**=[**ON**|**OFF**]: Enables or disables cache affinity optimization for this process. • **MEMORY_AFFINITY**=[**ON**|**OFF**]: Enables or disables memory affinity optimization for this process. • **LARGE_PAGE**=[**ON**|**OFF**]: Enables or disables large page optimization. • **MEMORY_PREFETCH**=[**ON**|**OFF**]: Enables or disables data stream prefetch optimization. • If set to any other value or if unset: ASO performs the default set of optimizations on the process **Change** `ASO_OPTIONS=<option string> export ASO_OPTIONS` This change affects processes that are running from the current shell after setting the variable. The change is effective until logging out of this shell. Permanent change can be made by setting the variable in the **/etc/environment** file. • To turn off the cache affinity optimization, set the ASO_OPTIONS environment variable as follows: `ASO_OPTIONS=CACHE_AFFINITY=OFF` • To enable the memory affinity optimization and to turn off other optimizations off, set the ASO_OPTIONS environment variable as follows: `ASO_OPTIONS=ALL=OFF,MEMORY_AFFINITY=ON`

asoo Command

Purpose

Manages the tunable parameters of the active system optimizer (ASO).

Syntax

asoo [**-p**|**-r**] [**-y**] {**-o** *Tunable* [=*Newvalue*]}

asoo [**-p**|**-r**] [**-y**] {**-d** *Tunable* }

asoo [**-p**|**-r**] [**-y**] **-D**

asoo [**-p**|**-r**] [**-F**] **-a**

asoo [**-h**] [*Tunable*]

asoo [**-F**] **-L** [*Tunable*]

asoo [**-F**] **-x** [*Tunable*]

Note: Multiple options, such as**-o**, **-d**, **-x**, and **-L**, are allowed.

Description

The **asoo** command is used to configure the ASO tunable parameters. This command sets or displays the current or next boot values for all ASO tunable parameters. It also makes permanent changes or defers changes until the next reboot operation.

Whether the command sets or displays a parameter is determined by the accompanying flag. The **-o** flag performs both actions. It can either display the value of a parameter or set a new value for a parameter.

Note: If used incorrectly, the **asoo** command can cause serious performance degradation or operating system failure.

Before changing any tunable parameter, first carefully read about all the tunable parameter characteristics in the Tunable Parameters section , and follow any Refer To pointer to fully understand its purpose. You must then ensure that the Diagnosis and Tuning sections for this parameter actually apply to your situation and that changing the value of this parameter could help improve the performance of your system. If the Diagnosis and Tuning sections both contain only **N/A**, do not change this parameter unless specifically directed by the AIX.

Flags

Item	Description
-a	Displays the current, reboot (when used in conjunction with the **-r** option), or permanent (when used in conjunction with the **-p** option) value for all tunable parameters, one per line in pairs: *Tunable=Value*. For the permanent options, a value only displays for a parameter if its reboot and current values are equal. Otherwise, it displays NONE as the value.
-d *Tunable*	Resets the *Tunable* parameter to the default values. If a *Tunable* parameter needs to be changed (that is, it is currently not set to its default value) and is of type **Bosboot** or **Reboot**, or if it is of type **Incremental** and has been changed from its default value, and the **-r** option is not used in combination, it is not changed but a warning is displayed instead.
-D	Resets all *Tunable* parameter to their default value. If *Tunable* parameter, which need to be changed are of type **Bosboot** or **Reboot**, or are of type **Incremental** and have been changed from their default value, and the **-r** option is not used in combination, they are not changed but a warning is displayed instead.
-F	Forces display of the restricted tunable parameters when the **-a**, **-L**, and **-x** options are specified alone on the command line to list all tunable parameters. When the **-F** flag is not specified, restricted tunable parameters are not displayed, unless these restricted tunable parameters are specifically named with a display option.
-h *Tunable*	Displays help about the tunable parameter if the parameter is specified. Otherwise, displays the **asoo** command usage statement.

Item	Description
-L *Tunable*	Lists the characteristics of one or all tunable parameters, one per line, using the following format: `NAME CUR DEF BOOT MIN MAX UNIT TYPE DEPENDENCIES` `---` `aso_active 1 1 1 0 1 D boolean` `---` `...` `where:` `CUR = current value` `DEF = default value` `BOOT = reboot value` `MIN = minimal value` `MAX = maximum value` `UNIT = tunable unit of measure` `TYPE = parameter type: D (for Dynamic), S (for Static), R (for Reboot, B (for Bosboot), M (for Mount), I (for Incremental), C (for Connect), and d (for Deprecated)` `DEPENDENCIES = list of dependent tunable parameters, one per line`
-o Tunable=[*NewValue*]	Displays the value or sets *tunable* parameter to *NewValue*. If a *tunable* parameter needs to be changed (the specified value is different from the current value), and is of type **Bosboot** or **Reboot**, or if it is of type **Incremental** and its current value is larger than the specified value, and the **-r** option is not used in combination, and is not changed but a warning is displayed instead. When the **-r** option is used in combination without a new value, the *nextboot* value for *tunable* parameter is displayed.
-p	When the **-p** option is used in combination without a new value, a value is displayed only if the current and next boot values for *tunable* parameter are the same. Otherwise, it displays NONE as the value. When used in combination with the **-o**, **-d**, or **-D** options, this flag applies changes to both the current and reboot values. That is, this flag turns on the updating function of the **/etc/tunables/nextboot** file in addition to turning on the updating function of the current value. These combinations cannot be used on the **Reboot** and **Bosboot** type parameters because their current value cannot be changed. When used with the **-a** or **-o** options without specifying a new value, values are displayed only if the current and next boot values for a parameter are the same. Otherwise, it displays NONE as the value.
-r	When the **-r** option is used in combination with the **-o**, **-d**, or **-D** options, this flag applies changes to the reboot values, for example, turns on the updating function of the **/etc/tunables/nextboot** file. If any parameter of type **Bosboot** is changed, you are prompted to run the **bosboot** command. When the **-r** option is used with the **-a** or **-o** options without specifying a new value, next boot values for tunable parameters are displayed instead of current values.
-x [*Tunable*]	Lists characteristics of one or all tunable parameter, one per line, by using the following (spreadsheet) format: `tunable,current,default,reboot,min,max,unit,type,{dtunable }` `where:` `current = current value` `default = default value` `reboot = reboot value` `min = minimal value` `max = maximum value` `unit = tunable unit of measure` `type = parameter type: D (for Dynamic), S (for Static), R (for Reboot), B (for Bosboot), M (for Mount), I (for Incremental), C (for Connect), and d (for Deprecated)` `dtunable = list of dependent tunable parameters`

Item	Description
-y	Suppresses the confirmation prompt before running the **bosboot** command.

If you make any change (with the **-o**, **-d**, or **-D** option) to a restricted tunable parameter, it results in a warning message that a tunable parameter of the restricted use type has been changed. If you also specify the **-r** or **-p** options on the command line, you are prompted for confirmation of the change. In addition, at system reboot, the presence of restricted tunable parameter in the **/etc/tunables/nextboot** file, which were changed to a value that is different from the default value (by using a command line for specifying the **-r** or **-p** options), results in an error log entry that identifies the list of these changed tunable parameters.

Tunable Parameters Type

All the tunable parameters that are manipulated by the tuning commands (**no**, **nfso**, **vmo**, **ioo**, **schedo**, **raso**, and **asoo**) are classified into the following categories:

Item	Description
Dynamic	The parameter can be changed at any time.
Static	The parameter can never be changed.
Reboot	The parameter can only be changed a during reboot operation.
Bosboot	The parameter can only be changed by running the **bosboot** command, and rebooting the system.
Mount	Changes to the parameter are only effective for future file systems or directory mounts.
Incremental	The parameter can only be incremented at boot time.
Connect	Changes to the parameter are only effective for future socket connections.
Deprecated	Changes to this parameter are no longer supported by the current release of AIX.

For parameters of the **Bosboot** type, whenever a change is performed, the tuning commands automatically prompt you to determine whether you want to run the **bosboot** command. For parameters of the **Connect** type, the tuning commands automatically restarts the **inetd** daemon.

Note: The current set of parameters that are managed by the **asoo** command only includes the Dynamic and Reboot types of tunable parameters.

Tunable Parameters

For default values and range of values for tunable parameters, see the help information for the **asoo** command (**-h**<*tunable_parameter_name*>).

Item	Description
aso_active	**Purpose** Disables the ASO. **Tuning** A value of 0 indicates that the ASO is disabled. A value of 1 indicates that the ASO is enabled.

Item	Description
debug_level	**Purpose** Changes the debug level of the ASO. **Tuning** A value of -1 (default) indicates that no debug information is collected. A value that is greater than -1 indicates that all levels of debug information at or below the level specified by this tunable parameter is collected. The location of the data collected is specified by the **aso.debug** entry in the **/etc/syslog.conf** file.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, ee the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list the current and reboot values, the range, the unit, the type, and dependencies of all the tunable parameters that are managed by the **asoo** command, enter:

   ```
   asoo -L
   ```

2. To list (spreadsheet format) the current and reboot values, the range, the unit, the type, and dependencies of all the tunable parameters that are managed by the **asoo** command, enter:

   ```
   asoo -x
   ```

3. To reset the **aso_active** tunable parameter to the default, enter:

   ```
   asoo -d aso_active
   ```

4. To display help information for the **aso_active** tunable parameter, enter:

   ```
   asoo -h aso_active
   ```

5. To permanently reset all the **asoo** tunable parameters to the default, enter:

   ```
   asoo -p -D
   ```

6. To list the reboot value for all the **asoo** parameters, enter:

   ```
   asoo -r -a
   ```

asa, fpr Command

Purpose

Prints FORTRAN files to in line-printer conventions.

Syntax

{ **asa** | **fpr** } [*File ...*]

Description

The **asa** and **fpr** commands print FORTRAN files to conform to this operating systems line-printer conventions. Both commands work like a filter to transform files formatted according to FORTRAN carriage control conventions into files formatted according to line-printer conventions.

The *File* variable specifies the name of the input file that the **asa** and **fpr** commands read instead of the standard input. The **asa** and **fpr** commands read the file, replace the carriage control characters with recognizable operating system characters, and print the file to standard output.

Both commands read the first character of each line from the input file, interpret the character, and space the line according to the definition of the first character. If the first character is either a **Blank**, a **0**, a dash (**-**) , a **1**, or a plus sign (**+**), either command does the following:

Item	Description
Blank	Advances the carriage one line and prints the input line.
0	Advances the carriage two lines and prints the input line.
-	Advances the carriage three lines and prints the input line.
1	Advances the carriage to the top of the next page.
+	Does not advance the carriage and starts printing the input line in the first space of the output file.

The commands interpret a blank line as if its first character is a blank and delete a blank that appears as a carriage control character. It treats lines that begin with characters other than the defined control characters as if they begin with a blank character. The first character of a line is not printed. If any such lines appear, an appropriate diagnostic appears in the standard error.

Note: Results are undefined for input lines longer than 170 characters.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. Use the **fpr** command in the following manner to change the carriage control characters in an `a.out` file produced by a FORTRAN compiler into carriage control characters and print the resulting file:

   ```
   a.out | fpr | qprt
   ```

2. Use the **asa** command in the following manner to run the `f77.output` file through the **asa** command to change carriage control characters from FORTRAN to the operating system and print the resulting file.

   ```
   asa f77.output | qprt
   ```

Files

Item	Description
/usr/ucb/fpr	Contains the **fpr** command.

Item	Description
/usr/bin/asa	Contains the **asa** command.

asa, fpr Command

Purpose

Prints FORTRAN files to in line-printer conventions.

Syntax

{ **asa** | **fpr** } [*File ...*]

Description

The **asa** and **fpr** commands print FORTRAN files to conform to this operating systems line-printer conventions. Both commands work like a filter to transform files formatted according to FORTRAN carriage control conventions into files formatted according to line-printer conventions.

The *File* variable specifies the name of the input file that the **asa** and **fpr** commands read instead of the standard input. The **asa** and **fpr** commands read the file, replace the carriage control characters with recognizable operating system characters, and print the file to standard output.

Both commands read the first character of each line from the input file, interpret the character, and space the line according to the definition of the first character. If the first character is either a **Blank**, a **0**, a dash (**-**) , a **1**, or a plus sign (**+**), either command does the following:

Item	Description
Blank	Advances the carriage one line and prints the input line.
0	Advances the carriage two lines and prints the input line.
-	Advances the carriage three lines and prints the input line.
1	Advances the carriage to the top of the next page.
+	Does not advance the carriage and starts printing the input line in the first space of the output file.

The commands interpret a blank line as if its first character is a blank and delete a blank that appears as a carriage control character. It treats lines that begin with characters other than the defined control characters as if they begin with a blank character. The first character of a line is not printed. If any such lines appear, an appropriate diagnostic appears in the standard error.

Note: Results are undefined for input lines longer than 170 characters.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. Use the **fpr** command in the following manner to change the carriage control characters in an `a.out` file produced by a FORTRAN compiler into carriage control characters and print the resulting file:

```
a.out | fpr | qprt
```

2. Use the **asa** command in the following manner to run the `f77.output` file through the **asa** command to change carriage control characters from FORTRAN to the operating system and print the resulting file.

```
asa f77.output | qprt
```

Files

Item	Description
/usr/ucb/fpr	Contains the **fpr** command.
/usr/bin/asa	Contains the **asa** command.

at Command

Purpose

Runs commands at a later time.

Syntax

To Schedule Jobs to Run at a Later Time

at [**-c** | **-k** | **-s** | **-q** *Queue*] [**-m**] [**-f** *File*] { **-t** *Date* |*Time* [*Day*] [*Increment*] }

To Report Scheduled Jobs

at -l [**-v**] [**-o**] [*Job ...* | **-q** *Queue*]

at -n [*User*]

To Remove Scheduled Jobs

at -r [**-F**] [**-i**] *Job ...*

at -r [**-F**] [**-i**] **-u** *User*

Description

The **at** command reads from standard input the names of commands to be run at a later time and allows you to specify when the commands should be run.

The **at** command mails you all output from standard output and standard error for the scheduled commands, unless you redirect that output. It also writes the job number and the scheduled time to standard error.

When the **at** command is executed, it retains the current process environment. It does not retain open file descriptors, traps, and priority.

The **/var/adm/cron/at.allow** and **/var/adm/cron/at.deny** files control what users can use the **at** command. A person with root user authority can create, edit, or delete these files. Entries in these files are user login names with one name to a line. The following is an example of an **at.allow** file:

```
root
nick
```

```
dee
sarah
```

If the **at.allow** file exists, only users whose login names appear in it can use the **at** command. A system administrator can explicitly stop a user from using the **at** command by listing the user's login name in the **at.deny** file. If only the **at.deny** file exists, any user whose name does not appear in the file can use the **at** command.

A user cannot use the **at** command if one of the following is true:

- The **at.allow** file and the **at.deny** file do not exist (allows root user only).
- The **at.allow** file exists but the user's login name is not listed in it.
- The **at.deny** file exists and the user's login name is listed in it.

If the **at.allow** file does not exist and the **at.deny** file does not exist, only users with root authority can submit a job with the **at** command.

To schedule a job to run later, you must specify a time to start the job. You might specify the time by using either the **-t** *Date* flag or the *Time*, *Day*, and *Increment* parameters. You can schedule any number of jobs at maximum granularity of 60 per second.

The *Date* variable to the **-t** flag is specified using the following format:

[[*CC*]*YY*]*MMDDhhmm*[**.***SS*]

The digits in the *Date* variable are defined as follows:

Item	Description
CC	Specifies the first two digits of the year (the century).
YY	Specifies the second two digits of the year.
MM	Specifies the month of the year (01 through 12).
DD	Specifies the day of the month (01 through 31).
hh	Specifies the hour of the day (00 through 23).
mm	Specifies the minute of the hour (00 through 59).
SS	Specifies the second of the minute (00 through 59).

Both the *CC* and *YY* digits are optional. If neither is given, the current year is assumed. If the *YY* digits are specified but the *CC* digits are not, the *CC* digits are defined as follows:

- If the value of the *YY* digits is between 70 and 99, the value of the *CC* digits is assumed to be 19.
- If the value of the *YY* digits is between 00 and 37, the value of the *CC* digits is assumed to be 20.
- The default value of *SS* is 00.

  For years between 2038 and 2105, specify year in the *yyyy* format.

The resulting time is affected by the value of the **TZ** environment variable.

The *Time* parameter may be specified as a number followed by an optional suffix. The **at** command interprets one- and two-digit numbers as hours. It interprets four digits as hours and minutes. The **T_FMT** item in the **LC_TIME** locale category specifies the order of hours and minutes. The default order is the hour followed by the minute. You can also separate hours and minutes with a : (colon). The default order is *Hour*:*Minute*.

In addition, you may specify one of the following suffixes:

- **am**
- **pm**
- **zulu**

If you do not specify **am** or **pm**, the **at** command uses a 24-hour clock. These suffixes can follow the time as a separate argument or separated with spaces. The **am** and **pm** suffixes are defined values from the **AM_STR** and **PM_STR** items in the **LC_TIME** locale category. The suffix **zulu** indicates that the time is **GMT** (Greenwich Mean Time).

The **at** command also recognizes the following keywords as special values for the *Time* parameter:

- **noon**
- **midnight**
- **now**
- **A** for AM
- **P** for PM
- **N** for noon
- **M** for midnight

You may specify the optional *Day* parameter as either a month name and a day number (and possibly a year number preceded by a comma), or a day of the week. The **D_FMT** item in the **LC_TIME** locale category specifies the order of the month and day (by default, month followed by day). The **DAY_1** through **DAY_7** items in the **LC_TIME** locale category specify long day names. The **ABDAY_1** through **ABDAY_7** items in the **LC_TIME** locale category specify short day names. The **MON_1** through **MON_12** items in the **LC_TIME** locale category specify long month names. The **ABMON_1** through **ABMON_12** items in the **LC_TIME** locale category specify short month names. By default, the long name is fully spelled out; the short name is abbreviated to two or more characters for weekdays, and three characters for months.

The **at** command recognizes **today** and **tomorrow** as special default values for the *Day* parameter. The **today** value is the default *Day* if the specified time is later than the current hour; the **tomorrow** value is the default if the time is earlier than the current hour. If the specified month is less than the current month (and a year is not given), next year is the default year.

Flags

Item	Description
-c	Requests that the **csh** command be used for executing this job.
-f *File*	Uses the specified file as input rather than using standard input.
-F	Suppresses delete verification. Use this flag with the **-r** flag.
-i	Specifies interactive delete. Use this flag with the **-r** flag.
-k	Requests that the **ksh** command be used for executing this job.
-l	Reports your scheduled jobs. If you have root user authority, you can get jobs issued by other users.
-m	Mails a message to the user about the successful execution of the command.
-n [*User*]	Reports the number of files in your queue or user's queue.
-o	Lists jobs in schedule order. This flag is useful only with the **-l** flag.

Item	Description
-q *Queue*	Specifies the queue in which to schedule a job for submission. When used with the **-l** flag, the report is limited to the queue specified by the *Queue* variable. By default, **at** jobs are scheduled in the **a** queue. The **b, c** and **d** queues are reserved for **batch**jobs, **cron** jobs, and **sync** jobs respectively. **-q a** Queues **at** jobs. **-q b** Queues **batch** jobs. The **batch** command calls the **at** command with this flag. **Note:** When using the **b** queue, commands are read from standard input. Also, the **now** keyword is used for the *Time* parameter, regardless of what you specify on the command line. **-q e** Queues **ksh** jobs. Equivalent to the **-k** flag. **-q f** Queues **csh** jobs. Equivalent to the **-c** flag. **-q g-z** Queues user defined queue jobs.
-r *Job*...	Removes *Job*s previously scheduled by the **at** or **batch** commands, where *Job* is the number assigned by the **at** or **batch** commands. If you do not have root user authority (see the **su** command), you can remove only your own jobs. The **atrm** command is available to the root user to remove jobs issued by other users or all jobs issued by a specific user.
-s	Requests that the **bsh** command (Bourne shell) be used for executing this job.
-t *Date*	Submits the job to be run at the time specified by the *Date* variable.
-u *User*	Deletes all jobs for the specified user. If used with the **-r** flag, do not specify a *Job* variable (the correct syntax is **at -r -u** *User*).
`-v`	Used with `-l` flag to show content of listed jobs.

Parameters

Item	Description
Day	Specifies the optional *Day* parameter as either a month name and a day number (and possibly a year number preceded by a comma), or a day of the week.
Increment	The optional *Increment* parameter can be one of the following:

- A **+** (plus sign) followed by a number and one of the following words:
  - **minute**[**s**]
  - **hour**[**s**]
  - **day**[**s**]
  - **week**[**s**]
  - **month**[**s**]
  - **year**[**s**]
- The special word **next** followed by a one of the following words:
  - **minute**[**s**]
  - **hour**[**s**]

- **day**[**s**]
- **week**[**s**]
- **month**[**s**]
- **year**[**s**]

Security

Auditing Events

If the auditing subsystem is properly configured and is enabled, the **at** command generates the following audit record or event every time the command is run:

Event	Information
AT_JobAdd	Lists **at** jobs that were run, the time the task was completed, and the user who issued the command.

For more details about how to properly select and group audit events, and how to configure audit event data collection, see **Setting Up Auditing** in *Security*.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Exit Status

This command returns the following exit values:

Item	Description
0	The **at** command successfully submitted, removed, or listed a job or jobs.
>0	An error occurred.

Examples

1. To schedule the command from the terminal, enter a command similar to one of the following:

   If uuclean is in your current directory, enter:

   ```
   at   5   pm   Friday
   uuclean
   <Ctrl-D>

   at   now   next   week
   uuclean
   <Ctrl-D>
   ```

   If uuclean is in **$HOME/bin/uuclean**, enter:

   ```
   at   now   +   2   days
   $HOME/bin/uuclean
   <Ctrl-D>
   ```

   Note: When entering a command name as the last item on the command line, a full path name must be given if the command is not in the current directory, and the **at** command will not accept any arguments.
2. To run the **uuclean** command at 3:00 in the afternoon on the 24th of January, enter any one of the following commands:

```
echo   uuclean   |   at   3:00   pm   January   24
echo   uuclean   |   at   3 pm   Jan   24
echo   uuclean   |   at   1500   jan   24
```

3. To have a job reschedule itself, invoke the **at** command from within the shell procedure by including code similar to the following within the shell file:

```
echo   "ksh   shellfile"   |   at   now   tomorrow
```

4. To list the jobs you have sent to be run later, enter:

   ```
   at   -l
   ```

5. To cancel a job, enter:

   ```
   at   -r   ctw.635677200.a
   ```

   This cancels job `ctw.635677200.a`. Use the **at -l** command to list the job numbers assigned to your jobs.

Files

Item	Description
/var/adm/cron/FIFO	A named pipe that sends messages to the **cron** daemon when new jobs are submitted with the **crontab** or **at** commands.
/usr/bin/at	Contains the **at** command.
/var/adm/cron	Contains the main **cron** directory.
/var/adm/cron/at.allow	Specifies the list of allowed users.
/var/adm/cron/at.deny	Specifies the list of denied users.

Item	Description
/var/spool/cron/atjobs	Contains the spool area directory for **at**.

ate Command

Purpose

Syntax

ate

Description

The **ate** command starts the Asynchronous Terminal Emulation (ATE) program. The ATE program establishes a connection between a workstation and a remote computer. A workstation acts as a terminal connected to the remote computer. Using ATE the user can connect to, and exchange data with, remote databases and other systems.

> **Note:** Users must be a member of the UNIX-to-UNIX Copy Program (uucp) group in order to use ATE. A user with root authority uses System Management Interface Tool (SMIT) to install individual users in groups.

ATE establishes the connection and allows users to record and control the session. After logging in to the remote system, users execute programs, issue commands, and use files on the remote system as a local user. ATE also enables a workstation to emulate a VT100 terminal.

The ATE program uses menus and subcommands. From the menus, users issue subcommands to connect to a remote system, receive and transfer files, and execute commands. The **Unconnected Main Menu** displays any time users issue the **ate** command. The **Connected Main Menu** displays when users press the MAINMENU_KEY (usually the Ctrl-V key sequence) while connected to another system. The **connect** subcommand makes the connection.

The ATE program supports three **control key** sequences: the CAPTURE_KEY (usually Ctrl-B), PREVIOUS_KEY (usually CTRL-R), and MAINMENU_KEY (usually CTRL-V). These control keys do not function until the ATE program is started. The control keys and other ATE defaults can be changed by editing the **ate.def** file format.

Examples

To start the ATE program, enter:

```
ate
```

The ATE **Unconnected Main Menu** displays.

Subcommands

Item	Description
alter	Temporarily changes data transmission characteristics in the ATE program.
break	Interrupts current activity on a remote system.
connect	Connects to a remote computer.
directory	Displays the ATE dialing directory.
help	Provides help information for the ATE subcommands.
modify	Temporarily modifies local settings used for terminal emulation.
perform	Allows the user to issue workstation operating system commands while using ATE.
quit	Exits the Asynchronous Terminal Emulation (ATE) program.
receive	Receives a file from a remote system.
send	Sends a file to a remote system.
terminate	Terminates an ATE connection to a remote system.

alter Subcommand

a [**l** *CharacterLength*] [**s** *StopBit*] [**p** *Parity*] [**r** *BaudRate*] [**d** *Device*] [**i** *DialPrefix*] [**f** *DialSuffix*] [**w** *Seconds*] [**a** *RedialAttempts*] [**t** *TransferProtocol*] [**c** *PacingType*]

> **Note:** The default values of the **alter** subcommand flags can be permanently changed by editing the **ate.def** file format.

The **alter** subcommand is accessed from the Asynchronous Terminal Emulation (ATE) **Connected** or **Unconnected** Main Menu. Issuing the **ate** command from the command line displays the Unconnected Main Menu. The **alter** subcommand temporarily changes these data transmission characteristics:

- Data character length
- Baud rate
- Stop and parity bits
- Port name
- Modem dialing prefixes and suffixes
- Waiting time and retry limits
- File transfer protocol

- Pacing character or delay time

The settings return to the defaults as defined in the **ate.def** file format when the user exits ATE.

When issued without flags from either of the ATE main menus, the **alter** subcommand displays the Alter Menu. To bypass the Alter Menu, enter the **alter** subcommand, followed by the appropriate flags, at the command prompt on either ATE main menu.

The **alter** subcommand can change more than one feature at a time. To change the value of more than one variable, type the first flag followed by the new value, followed by a space, then the second flag and second value, and so on.

To permanently change the settings affected by the **alter** subcommand, customize the **ate.def** file format.

The Alter Menu

The Alter Menu displays the current settings of the changeable characteristics with the **alter** subcommand. Enter the letter **a** after the command prompt on either the ATE **Connected** or **Unconnected** Main Menu to view the Alter Menu.

The Alter Menu contains the following columns:

Column Names	Contents
COMMAND	Flag that changes the value of a variable
DESCRIPTION	Description of the variable that the flag affects
CURRENT	Current value of the variable
POSSIBLE CHOICES	Possible values of the variable

To change the value of a variable, enter the flag (from the COMMAND column) and new value (from the POSSIBLE CHOICES column) at the command prompt on the Alter Menu.

To return to one of the ATE main menus from the Alter Menu, press the Enter key.

Flags

Item	Description
a *RedialAttempts*	Specifies the maximum number of times the ATE program redials for a connection. If the *RedialAttempts* variable is 0, no redial attempt occurs. Options: 0 (none) or a positive integer Default: 0

Item	Description
c *PacingType*	Specifies the type of **pacing** protocol used. Default: 0 (no pacing) **Note:** The *PacingType* variable has no effect when the **xmodem** protocol is used. The *PacingType* can be either of the following: ***Character*** Signal to transmit a line. The signal can be any ASCII character. When the **send** subcommand encounters a line-feed character while transmitting data, it waits to receive the pacing character before sending the next line. When the **receive** subcommand is ready to receive data, it sends the pacing character and then waits 30 seconds to receive data. The **receive** subcommand sends a pacing character again whenever it finds a carriage-return character in the data. The **receive** subcommand ends when it receives no data for 30 seconds. ***Interval*** Number of seconds the system waits between each line it transmits. The value of the *Interval* variable must be an integer. The default value is 0 indicating a pacing delay of 0 seconds.
d *Device*	Specifies the name of the asynchronous port used to connect to a remote system. Options: Locally created port names. The first 8 characters of the port name display in the Alter Menu. Default: tty0
f *DialSuffix*	Specifies the dial suffix that must follow the telephone number when autodialed with a modem. Consult the modem documentation for the proper dial command. Options: 0 (none) or a valid modem suffix. The first 8 characters display in the Alter Menu. Default: no default
i *DialPrefix*	Specifies the dial prefix that must precede the telephone number when autodialed with a modem. Consult the modem documentation for the proper dial commands. Options: ATDT, ATDP, or other values depending on the type of modem used. The first 8 characters display in the Alter Menu. Default: ATDT
l *CharacterLength*	Specifies the number of bits in a data character. This length must match the length expected by the remote system. Options: 7 or 8 Default: 8

Item	Description
p *Parity*	Checks whether a character was successfully transmitted to or from a remote system. Must match the parity of the remote system. For example, if the user selects even parity, when the number of 1 bits in the character is odd, the parity bit is turned on to make an even number of 1 bits. Options: 0 (none), 1 (odd), or 2 (even) Default: 0
r *BaudRate*	Specifies the baud rate, or bits transmitted per second (bps). The speed must match the speed of the modem and that of the remote system. Options: 50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2400, 4800, 9600, or 19200 Default: 1200
s *StopBit*	Specifies the number of stop bits appended to a character to signal the end of that character during data transmission. This number must match the number of stop bits used by the remote system. Options: 1 or 2 Default: 1
t *TransferProtocol*	Defines the type of asynchronous protocol that transfers files during a connection. **p** File transfer protocol controls the data transmission rate by waiting for either a specified character or a certain number of seconds between line transmissions. This helps prevent loss of data when the transmission blocks are either too large or sent too quickly for the system to process. **x** An 8-bit file transfer protocol to detect data transmission errors and retransmit the data. Options: **p** (**pacing)**, or **x** (**xmodem**) Default: **p**
w *Seconds*	**wait** Specifies the number of seconds between redial attempts. The wait period does not begin until the connection attempt times out or until it is interrupted. If the **attempts** flag is set to 0, no redial attempt occurs. Options: 0 (none) or a positive integer Default: 0

Examples

1. To display the Alter Menu, enter the **alter** subcommand at the command prompt on either ATE main menu:

   ```
   a
   ```

   The Alter Menu is displayed.

2. To alter transmission settings from the Alter Menu, enter the appropriate flags at the command prompt on the Alter Menu:

- To change the value for the **rate** flag, enter:

```
r 9600
```

For the current session of ATE, the baud rate is changed to 9600 bps.

- To change the value of the **wait** flag, enter:

```
w 7
```

For the current session of ATE, the wait time for redial changes to 7 seconds.

- To bypass the Alter Menu when using the **alter** command, type the command abbreviation **a**, followed by the appropriate flags, at the prompt on one of the ATE main menus. For example, to change the **rate**, **wait**, and **attempt** values, enter the following at the prompt on either ATE main menu:

```
a r 9600 w 5 a 1
```

For the current session of ATE, the baud rate changes to 9600 bps, the wait time for redial changes to 5 seconds, and the maximum number of redial attempts changes to 1 attempt.

break Subcommand

b

The **break** subcommand sends a break signal to the remote system connected to the terminal by the Asynchronous Terminal Emulation (ATE) program. The **break** subcommand interrupts current activity on the remote system. Issue the **break** subcommand from the ATE **Connected Main Menu**.

> **Attention:** The **break** subcommand may disconnect the current session. The system may lose data.

Example

To interrupt the current session, at the remote system login screen, press the **MAINMENU_KEY** (usually the Ctrl-V key sequence). When the ATE Connected Main Menu displays, enter:

```
b
```

A break signal is sent to the remote system, and the ATE **Unconnected Main Menu** displays. Now exit the ATE program or issue other ATE subcommands.

connect Subcommand

c [*TelephoneNumber* | *PortName*]

The ATE **connect** subcommand enables users to connect to a remote computer using Asynchronous Terminal Emulation (ATE). Issue the **connect** subcommand from the **ATE Unconnected Main Menu**. The connection can be made between two machines connected by cable or by telephone line. Users establish connection in one of three ways:

Item	Description
direct	Uses an established cabled link to another system.
manually dialed	Uses a telephone number dialed by the user.
automatically dialed	Uses a modem to dial a specified telephone number (a modem-dialed connection).

If the system login is not disabled, attempts to connect to another computer return an error. To disable the workstation port that handles system login by remote users, a user with root authority must use the

pdisable command. Once the workstation port is secure from remote logins, the user must then ensure the remote system is ready to receive calls.

No connection is established if the line is busy, if the party does not answer, or if the user specified an unrecognized number. If any of these conditions exist, a message is displayed.

If a busy signal is received while trying to connect to a remote workstation, press the **PREVIOUS_KEY** (usually the Ctrl-R key sequence), and enter the *TelephoneNumber* parameter again.

Once the connection is established, ATE displays a message indicating the name of the port used for the connection.

Parameters

Item	Description
PortName	Specifies the name of the port used for a direct connection.
TelephoneNumber	Specifies the telephone number used to establish a modem connection.

Examples

1. To establish a direct connection, at the command line of the **ATE Unconnected Main Menu**, enter:

   ```
   c tty0
   ```

   This command establishes a direct connection using port `tty0`. After connection is established, a message displays, followed by a login screen. Enter the requested login information and press the **MAINMENU_KEY**(usually the Ctrl-V key sequence) to display the ATE **Connected Main Menu**.
2. To establish a manually dialed connection, at the command line of the ATE Unconnected Main Menu, enter:

   ```
   c
   ```

   The ATE program prompts the user for information necessary to establish a manually dialed connection, such as a telephone number or modem to use. After connection is established, ATE displays a message giving the port name used for the connection, followed by a login screen. Enter the requested login information and press the MAINMENU_KEY (usually the Ctrl-V key sequence) to display the ATE Connected Main Menu.
3. To establish an automatically dialed connection, at the command line of the ATE Unconnected Main Menu, enter:

   ```
   c  2229999
   ```

   This example dials the telephone number 222-9999. After connection is established, a message displays indicating the port used for the connection, followed by a login screen. Enter the requested login information and press the MAINMENU_KEY (usually the Ctrl-V key sequence) to display the ATE Connected Main Menu.

directory Subcommand

d

The ATE **directory** subcommand displays a **dialing directory**. Users establish a connection to a remote computer by selecting one of the directory entries from the displayed directory. The **directory** subcommand is issued from the ATE **Unconnected Main Menu**. The **directory** subcommand uses the information contained in the dialing directory to establish an automatically dialed (modem-dialed) connection.

When ATE starts, it checks the current directory for an **ate.def** file format. If an **ate.def** file format does not exist in the current directory, it creates one. The initial location of the dialing directory is **/usr/lib/dir**, but this value can be changed by **Editing the ATE default file** the **ate.def** file format. If users specify a different dialing directory in the **ate.def** file format, that directory is used.

The dialing directory contains entries for remote systems called with the ATE program in the format:

Name Phone Rate Length StopBit Parity Echo Linefeed

These fields give the name of the entry (usually the person or company whose computer the phone number reaches), the telephone number, and other information the ATE program uses to establish the connection.

When an entry displays on the screen using the **directory** subcommand, the entry is preceded by an entry number. Select the entry to establish a connection to by entering its entry number in response to a prompt.

Example

To display a dialing directory, at the command line of the Unconnected Main Menu, enter:

```
d
```

The dialing directory specified in the **ate.def** file format displays and prompts the user for an entry number. Enter the number of the dialing directory entry to establish a connection with. ATE establishes the connection and displays a message indicating the port name used.

help Subcommand

h [**a**] [**b**] [**c**] [**d**] [**m**] [**p**] [**q**] [**r**] [**s**] [**t**]

The ATE **help** subcommand provides help information for the ATE subcommands. Issue the **help** subcommand from either the **Unconnected** or **Connected** Main Menu of ATE. Help information is available for all the ATE subcommands, and can be requested for several subcommands at the same time.

When issuing the **help** subcommand, ATE displays a description of each subcommand requested and instructions for using the subcommand. Help information for each subcommand displays individually, in the order requested. After reading each help message, press Enter to view the next page of help text. At the end of the help text, press Enter to return to the main menu.

Issue the **help** subcommand with the first letter of an ATE subcommand for help information. These are the names for the ATE subcommands:

Name	ATE Subcommand
a	**alter** subcommand
b	**break** subcommand
c	**connect** subcommand
d	**directory** subcommand
m	**modify** subcommand
p	**perform** subcommand
q	**quit** subcommand
r	**receive** subcommand
s	**send** subcommand
t	**terminate** subcommand

Examples

1. To receive help information for a single subcommand, enter the following at one of the ATE main menus:

   ```
   h c
   ```

   Help information displays for the **connect** (c) subcommand. After viewing the help information, press the Enter key, and ATE displays the menu from which the **help** subcommand was issued.

2. To receive help information for multiple subcommands, enter the following at one of the ATE main menus:

```
h r s
```

   The help information for the **receive** subcommand (`r`) displays first. After viewing the help information, press the Enter key. Help information for the **send** subcommand (`s`) displays. After viewing the help information, press the Enter key, and ATE displays the menu from which the **help** subcommand was issued.

modify Subcommand

m [**n** *CaptureFileName*] [**e**] [**l**] [**v**] [**w**] [**x**]

> **Note:** The default *CaptureFileName* and the initial settings of the other **modify** subcommand flags can be permanently changed in the **ate.def** file format.

The **modify** subcommand is accessed from the Asynchronous Terminal Emulation (ATE) **Connected** or **Unconnected** Main Menu. The **modify** subcommand temporarily changes how ATE functions on the local system in the following ways:

- Changes the name of the capture file that receives incoming data.
- Switches (toggles) the following features on or off:
  - Add a line-feed character at the end of each line of incoming data.
  - Use echo mode.
  - Emulate a DEC VT100 terminal at the console.
  - Write incoming data to a capture file as well as to the display.
  - Use an **Xon/Xoff** (transmitter on/off) signal.

The settings return to the default values as defined in the **ate.def** file format when the user exits ATE.

When issued without flags from either of the ATE main menus, the **modify** subcommand displays the Modify Menu. The Modify Menu can be bypassed by entering **m** (the **modify** subcommand abbreviation), followed by the appropriate flags, at the command prompt on either ATE main menu.

The **modify** subcommand can change more than one feature at a time. To change the **name** variable, enter the **n** flag followed by the new file name. All other variables are switches that can be turned on or off by typing the flag. Typing the flag switches, or toggles, the value.

To permanently change the settings affected by the **modify** subcommand, customize the **ate.def** file format in the directory running ATE.

Modify Menu

The Modify Menu displays the current settings of the features changeable with the **modify** subcommand. To display the Modify Menu, enter the letter **m** after the command prompt on either the ATE **Connected Main Menu** or the ATE **Unconnected Main Menu**.

The Modify Menu contains the following columns:

Column Names	Contents
COMMAND	Flag to enter to change a value
DESCRIPTION	Description of the variable the flag affects
CURRENT	Current value of the variable
POSSIBLE CHOICES	Possible values of the variable

To change the value of a flag other than the **name** flag, enter the flag (from the COMMAND column) at the command prompt on the Modify Menu. The flag value toggles to the alternate setting. To change the name of the capture file, enter the letter **n** (the **name** flag), followed by the new file name, at the prompt on the Modify Menu.

To return to the ATE Connected or Unconnected Main Menu from the Modify Menu, press the Enter key.

Flags

Item	Description
e	**echo** Displays the input typed by the user. With a remote computer that supports echoing, each character sent returns and displays on the screen. When the **echo** flag is on, each character is displayed twice: first when it is entered and again when it returns over a connection. When the **echo** flag is off, each character displays only when it returns over the connection. Options: On or off Default: Off
l	**linefeed** Adds a line-feed character after every carriage-return character in the incoming data stream. Options: On or off Default: Off
n *CaptureFileName*	**name** Specifies the file name for incoming data when the **write** flag is on, or when the **CAPTURE_KEY** (usually the Ctrl-B key sequence) is pressed during a connection. Options: Any valid file name. The first 18 characters display in the Modify Menu. Default: **capture**
v	**VT100** The local console emulates a DEC VT100 terminal so DEC VT100 codes can be used with the remote system. With the **VT100** flag off, the local console functions like a workstation. Options: On or off Default: Off **Note:** No keys on the console keyboard are remapped. In addition, some DEC VT100 codes, such as 132 columns, double-height and double-width lines, origin mode, and graphics characters generated from a 10-key keypad, are not supported.
w	**write** Routes incoming data to the capture file (specified by the **name** flag) as well as to the display. The **write** command functions like the **CAPTURE_KEY** key sequence during a connection. Carriage return and line-feed combinations are converted to line-feed characters before being written to the capture file. In an existing file, data is appended to the end of the file. Options: On or off Default: Off

Item	Description
x	**Xon/Xoff** Controls data transmission at a port using the **Xon/Xoff** protocol, as follows: • When an **Xoff** signal is received, transmission stops. • When an **Xon** signal is received, transmission resumes. • An **Xoff** signal is sent when the receive buffer is nearly full. • An **Xon** signal is sent when the buffer is no longer full. Options: On or off Default: On **Note:** If you use a variable value with any flag other than the **name** flag, the following error message displays: `828-003 not 'command-name' command is not valid.` `Enter the first letter of a command` `from the list on the menu.` This error message indicates either an incorrect letter was entered or a value that is not valid was included.

Examples

1. To display the Modify Menu, enter the **modify** subcommand at the command prompt on either ATE main menu:

   ```
   m
   ```

   The Modify Menu displays.
2. To modify settings from the Modify Menu, enter the appropriate flag at the command prompt at the bottom of the Modify Menu:
   - To toggle the values of the **linefeed** flag, at the prompt on the Modify Menu enter:

     ```
     l
     ```

     The value of the **linefeed** flag is switched to the alternate setting.
   - To change the **name** variable to `schedule`, at the prompt on the Modify Menu enter:

     ```
     n schedule
     ```

     Any data saved is now put into the `schedule` file.
3. To bypass the Modify menu when using the **modify** subcommand, type the **m** subcommand (the **modify** subcommand abbreviation), followed by the appropriate flags, at the command prompt on either ATE main menu:
   - To toggle the values of the **linefeed** and **echo** flags, at the prompt on either ATE main menu enter:

     ```
     m l e
     ```

     The values of the **linefeed** and **echo** flags are switched to the alternate settings. Display the Modify Menu to view the current settings of the flags.
   - To change the **name** variable to `schedule` and toggle the values of the **write** and **Xon/Xoff** flags, at the prompt on either ATE main menu enter:

     ```
     m n schedule w X
     ```

Any data saved is now put into the `schedule` file, and the values of the **write** and **Xon/Xoff** flags are switched to the alternate settings. Display the Modify Menu to view the settings of the flags.

perform Subcommand

p [*Command*]

The ATE **perform** subcommand allows the user to issue workstation operating system commands while using Asynchronous Terminal Emulation (ATE). Issue the **perform** subcommand from the ATE **Unconnected** or **Connected** Main Menu. *Command* specifies a valid workstation operating system command.

Examples

1. To issue a workstation operating system command, at the command line of the ATE Unconnected or Connected Main Menu, enter:

   ```
   p
   ```

   ATE prompts the user to enter a command. ATE executes the specified command. After the command finishes, ATE displays the menu from which the **perform** subcommand was issued.
2. To specify the command to be executed, at the command line of the ATE Unconnected or Connected Main Menu, enter:

   ```
   p cat mystuff
   ```

   ATE executes the **cat** command, which displays the `mystuff` file. After the **cat** command finishes, ATE displays the menu from which the **perform** subcommand was issued.

quit Subcommand

q

The ATE **quit** subcommand exits the Asynchronous Terminal Emulation (ATE) program. Issue the **quit** subcommand from the ATE **Unconnected** or **Connected** Main Menu. Issuing the **quit** subcommand ends the ATE program and displays the command prompt.

Example

To exit the ATE program, from the command line of either ATE main menu, enter:

```
q
```

The ATE program ends and the command prompt displays.

receive Subcommand

r *FileName*

The ATE **receive** subcommand enables your system to receive a file from a remote system. The ATE **receive** subcommand is issued from the ATE **Connected Main Menu**.

The ATE **receive** subcommand uses the **xmodem** file transfer protocol, which enables your system to receive data from a remote system, a block at a time, with error checking. The remote system must be set to send the file before your system can receive. Use the **xmodem** command with the **-s** flag on the remote system to enable the remote system to send the file. Then issue the **receive** subcommand. *FileName* names the file where the received data is stored.

Example

To receive a file sent from the remote system, at the command line of the ATE Connected Main Menu, enter:

```
r myfile
```

The data is received from the remote system and is stored in the `myfile` file.

send Subcommand

s [*FileName*]

The ATE **send** subcommand sends a file to a remote system. Issue the ATE **send** subcommand from the ATE **Connected Main Menu** once a connection is established. The ATE **connect** subcommand establishes the connection and prepares the remote system to receive files.

The **send** subcommand uses the **xmodem** file transfer protocol, sending data to a remote system, a block at a time, with error checking. Issue the **xmodem** command with the **-r** flag on the remote system to enable the remote system to receive the file. Then issue the **send** subcommand. *FileName* names the file to send to the remote system.

Examples

1. To send a file to a remote system, at the command line of the ATE Connected Main Menu, enter:

   ```
   s
   ```

   ATE prompts the user for the name of the file to send to the remote system.
2. To specify a file to send to the remote system, at the command line of the ATE Connected Main Menu, enter:

   ```
   s mystuff
   ```

   The `mystuff` file is sent to the remote system.

terminate Subcommand

t

The ATE **terminate** subcommand ends an Asynchronous Terminal Emulation (ATE) connection to a remote system and returns to the ATE **Unconnected Main Menu**. Issue the **terminate** subcommand from the ATE **Connected Main Menu**.

Example

To terminate the current session, from the remote system login screen, press the **MAINMENU_KEY** (usually the Ctrl-V key sequence). When the ATE Connected Main Menu displays, enter:

```
t
```

A terminate signal is sent to the remote system, the session ends, and ATE displays the Unconnected Main Menu. Now issue other ATE subcommands or exit ATE.

File

Item	Description
/usr/lib/dir	Contains the default dialing directory.

atq Command

Purpose

Displays the queue of jobs waiting to be run.

Syntax

atq [**c** | **-n**] [*User* ...]

Description

The **atq** command displays the current user's queue of jobs that are waiting to be run at a later date, sorted in the order the jobs will be run. These jobs were created with the **at** command. If the user is root and *User* name is specified, the **atq** command displays only jobs belonging to that user.

Flags

Item	Description
-c	Sorts the queue by the time that the **at** command was issued.
-n	Displays only the number of jobs currently in the queue.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

In order to look at the queue created by the **at** command, enter:

```
atq
```

If there are jobs in the queue, a message similar to the following appears:

```
root.635623200.a         Wed     Feb 21  12:00:00 1990
root.635670000.a         Thu     Feb 22  01:00:00 1990
```

Files

Item	Description
/usr/bin/atq	Contains the **atq** program.
/var/spool/cron/atjobs	Specifies the spool area.

atrm Command

Purpose

Remove jobs spooled by the **at** command.

Syntax

atrm [**-f**] [**-i**] [**-a** | **-**] [*Job* ... | *User* ...]

Description

The **atrm** command removes jobs that were created with the **at** command, but have not executed. If one or more job numbers is specified, the **atrm** command attempts to remove only those jobs.

If one or more user names is specified, all jobs belonging to those users are removed. This form of invoking the **atrm** command is useful only if you have root user authority.

Flags

Item	Description
-	Removes all jobs belonging to the user invoking the **atrm** command.
-a	Removes all jobs belonging to the user invoking the **atrm** command. This flag is provided for System V compatibility.
-f	Suppresses all information about the jobs being removed.
-i	Prompts before a job is removed. Enter y to remove the job.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To remove job number `root.62169200.a` from the **at** command queue, enter:

```
atrm root.621619200.a
```

Files

Item	Description
/usr/bin/atrm	Contains the **atrm** program file.
/var/spool/cron/atjobs	Specifies the spool area.

attachrset Command

Purpose

Attaches an rset to a process.

Syntax

```
attachrset [ -P ]  [ -F ]  [ -S ]  rsetname pid
```

or

```
attachrset [ -P ]  [ -F ] [ -c CPUlist ] [ -m MEMlist ] pid
```

Description

The **attachrset** command attaches an rset to a process. The command limits the specified process to run only on the processors and/or memory regions contained in the rset. An rset name in the system registry can be attached to the process. Or, an rset containing the specified processors and memory regions can be attached to the process.

Flags

Item	Description
-P	Attaches an rset as a partition rset.
-F	Forces the rset attachment to occur. This option will remove a bindprocessor bind and all threads' rset in the process before attaching the new rset. If the **-P** option is also specified, it will also detach the effective all threads' rset from the process before attaching the new rset.
-c *CPUlist*	List of CPUs to be in the rset. This can be one or more CPUs or CPU ranges.
-m *MEMlist*	List of memory regions to be in the rset. This can be one or more memory regions or ranges.
-S	A hint that indicates that the process must be scheduled to run in single-threaded mode. Only one of the hardware threads of each physical processor that is included in the specified rset will be used to schedule the job. If all the hardware threads of a physical processor are not included in the specified rset, that processor will be ignored. The specified rset must be an exclusive rset or the command fails. Specifying this flag allows jobs to run with single-thread behavior.

Parameters

Item	Description
rsetname	The name of the rset to be attached to the process. The name consists of a *namespace* and an *rsname* separated by a "/" (slash). Both the *namespace* and *rsname* may contain up to 255 characters. See the **rs_registername()** service for additional information about character set limits of rset names.
pid	Process ID to connect rset.

Security

The user must have `root` authority or have **CAP_NUMA_ATTACH** capability and read access to the specified rset registry name (if **-r** option used) and target process must have the same effective userid as the command issuer. The user must have `root` authority to set the partition rset on a process (the **-P** option).

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To attach an **rset** containing CPUs 0-7 to process 18838, type:

```
attachrset -c 0-7 18838
```

2. To attach **rset** named **test/cpus0to7** to process 20124, type:

```
attachrset test/cpus0to7 20124
```

Files

Item	Description
/usr/bin/attachrset	Contains the **attachrset** command.

audit Command

Purpose

Controls system auditing.

Syntax

audit { **on** [**panic** | **fullpath**] | **off** | **query** | **start** | **shutdown** }{**-@** *wparname* ...}

Description

The **audit** command controls system auditing through several keywords. You must include one keyword each time you enter the command. The **start** keyword and the **shutdown** keyword start and stop the auditing system and reset the system configuration. The **off** keyword and the **on** keyword suspend and restart the audit system without affecting the system configuration. The **query** keyword lets you query the current status.

The auditing system follows the instructions established in the following configuration files:

- **/etc/security/audit/config**
- **/etc/security/audit/events**
- **/etc/security/audit/objects**
- **/etc/security/audit/bincmds**
- **/etc/security/audit/streamcmds**

The **-@** option is not supported when you run it in a WPAR.

Keywords

Item	Description
start	Starts the audit subsystem. This keyword reads the instructions in the configuration files and performs the following tasks: **role auditing** Audits all roles currently active in to the system, if they are configured in the roles stanza of the **/etc/security/audit/config** file. **object auditing** Writes the audit event definitions in the **/etc/security/audit/objects** file into the kernel to define the object auditing events. **Note:** When the parent directory of one of the file-system objects does not exist, the flag fails and issues an **ENOENT** error. **event auditing** Writes the audit class definitions in the **/etc/security/audit/config** file into the kernel to define the audit classes. **bin auditing** Starts the **auditbin** daemon according to the configuration information in the bin stanza in the **/etc/security/audit/config** file, if the start stanza contains `binmode=on`. **stream auditing** Invokes the audit stream commands as defined in the stream stanza in the **/etc/security/audit/config** file, if the start stanza contains `streammode=on`. **Note:** Avoid invocation of stream auditing during boot time or from remote shell (rsh) unless the standard output (`stdout`) and standard error (`stderr`) processes are closed on invocation, that is, when the following command is run: `/usr/sbin/audit start 1>&- 2>&-`. **fullpath auditing** Captures the full path name of a file for the `FILE_Open`, `FILE_Read`, and `FILE_Write` auditing events, if the start stanza in the `/etc/security/audit/config` file contains `fullpath=on`. **user auditing** Audits all users currently logged into the system, if they are set up in the users stanza of the **/etc/security/audit/config** file. **audit logging** Enables the audit logging component as defined in the start stanza in the **/etc/security/audit/config** file. **audit ranges** Writes the Trusted AIX audit ranges into the kernel if they are set up in the WPAR Audit Ranges (WAR) stanza of the **/etc/security/audit/config** file. **global-initiated WPAR auditing** Audits the WPARs, if they are stored in the WPARS stanza of the **/etc/security/audit/config** file. The auditing can be used only from global WPAR by specifying the **-@** ***wparname*** parameter in the command.

Item	Description
shutdown	Terminates the collection of audit records and resets the configuration information by removing the definition of classes from the kernel tables. All the audit records are flushed from the kernel buffers into the bin files or audit streams, according to the specifications for the backend commands, which are contained in the **/etc/security/audit/bincmds** file for binmode auditing, and in the **/etc/security/audit/streamcmds** file for streammode auditing. The collection of audit data stops until you give the next **audit start** command. When you use the **-@** ***wparname*** parameter with this keyword, auditing is disabled for the specified WPAR.
off	Suspends the auditing system, but leaves the configuration valid. Data collection pauses until you give the **audit on** command. The **-@** option is not supported with this keyword.
on [**panic** | **fullpath**]	Restarts the auditing system after a suspension, if the system is properly configured (for example, if the **audit start** command was used initially and the configuration is still valid). If auditing has already started when the command is given, only bin data collection can be changed. The **-@** option is not supported with this keyword. If you specify the **panic** option, the system halts abruptly if bin data collection is enabled but cannot be written to a bin file. The **panic** option is not supported when you run it in a WPAR. If you specify the **fullpath** option, the FILE_Open, FILE_Read and FILE_Write auditing events capture the full path name of a file.
query	Queries the auditing status of the audit subsystem. If you specify the **-@** option, this keyword queries the auditing status of a global initiated WPAR. This keyword displays the current status of the audit subsystem in the following format: `auditing on {panic | fullpath} | auditing off` `bin manager off |  is process number pid` `audit events:` `    audit class: audit event, audit event...` `audit objects:` `    object name: object mode: audit event`

Security

Access Control

This command should grant execute (x) access to the root user and members of the audit group. The command should be **setuid** to the root user and have the **trusted computing base** attribute.

Files Accessed

Mode	File
r	**/etc/security/audit/config**
r	**/etc/security/audit/objects**
x	**/usr/sbin/auditbin**
x	**/usr/sbin/auditstream**

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To start the audit process, configure the audit system as described in "Setting up Auditing" in *Security*, and add the following line to the system initialization file (the **/etc/rc** in the global environment or the **/etc/rc.bootc** in WPAR):

   ```
   /usr/sbin/audit start 1>&- 2>&-
   ```

   The audit process starts, as configured, each time the system is initialized.
2. To start the audit process for the WPAR named `wpar1` from the global WPAR, enter the following command:

   ```
   /usr/sbin/audit start -@ wpar1
   ```
3. To terminate the operation of the auditing process, enter the following command:

   ```
   /usr/sbin/audit shutdown
   ```

   Data collection stops until the **audit start** command is specified again. The configuration of classes in the operating system kernel is lost.

   Note: The **audit shutdown** command should be in the **/etc/shutdown** file as well.
4. To terminate the auditing process of the WPAR named `wpar1` from global WPAR, enter the following command:

   ```
   /usr/sbin/audit shutdown -@ wpar1
   ```

   Data collection stops until the **audit start -@ wpar1** command is specified again. The configuration of classes in the operating system kernel is lost.

   Remember: The **audit shutdown** command, without any options, shuts down the auditing process of all WPARs started from the global WPAR.
5. To suspend the audit subsystem, enter the following command:

   ```
   /usr/sbin/audit off
   ```
6. To restart an audit process that was suspended by the **audit off** command, enter the following command:

   ```
   /usr/sbin/audit on
   ```

   The suspended state ends and audit records are generated again, as long as the system is configured correctly.
7. To display the current status of the auditing system, enter the following command:

   ```
   /usr/sbin/audit query
   ```

   The following is an example of an **audit query** status message:

   ```
   auditing on

   bin manager is process number 123

   audit events:
       authentication- USER_Login, USER_Logout
       administration- USER_Create, GROUP_Create

   audit objects:
       /etc/security/passwd :
         r = AUTH_Read
       /etc/security/passwd :
         w = AUTH_Write
   ```

The query informs you that audit records are written when the specified users log in or log out, when the specified administrators create a user or a group, and when the system receives an authorized read or write instruction for the **/etc/security/passwd** file.

Files

Item	Description
/etc/security/audit/bincmds	Contains shell commands for processing audit bin data.
/etc/security/audit/config	Contains audit configuration information.
/etc/security/audit/events	Lists the audit events and their tail format specifications.
/etc/security/audit/objects	Lists the audit events for each file (object).
/etc/security/audit/streamcmds	Contains auditstream commands.
/etc/rc	Contains the system initialization commands.
/usr/sbin/audit	Contains the path of the **audit** command.

auditbin Daemon

Purpose

Manages bins of audit information.

Syntax

auditbin

Description

The **auditbin** daemon in the audit subsystem manages **bin1** and **bin2**, temporary bin files that alternately collect audit event data. The command also delivers bins of data records to backend commands for processing.

As audit events occur, the operating system kernel writes a record to a bin file. When a bin file is full, the **auditbin** daemon reads the **/etc/security/audit/bincmds** file and delivers the bin records to the backend commands defined in the file. Each line of the **/etc/security/audit/bincmds** file contains one or more commands with input and output that can be piped together or redirected. The **auditbin** daemon searches each command for the **$bin** string and the **$trail** string and substitutes the path names of the current bin file and the system trail file for these strings.

The **auditbin** daemon ensures that each command encounters each bin at least once, but does not synchronize access to the bins. When all the commands have run, the bin file is ready to collect more audit records.

If a command is unsuccessful, the **auditbin** daemon stops delivering data records and sends a message to the **/dev/tty** device every 60 seconds until the root user or a member of the audit group stops the command.

Security

Access Control

This command should grant execute (x) access to the root user and members of the audit group. The command should be **setuid** to the root user and have the **trusted computing base** attribute.

Files Accessed

Mode	File
r	**/etc/security/audit/config**
r	**/etc/security/audit/bincmds**
rw	Defined audit bins and trail file
x	All audit bin processing commands

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To configure the **auditbin** daemon, edit the start and bin stanzas of the **/etc/security/audit/config** file to include the following attribute definitions:

```
start:
         binmode = on

bin:
         trail = /audit/trail
         bin1 = /audit/bin1
         bin2 = /audit/bin2
         binsize = 25000
         cmds = /etc/security/audit/bincmds
```

2. To define the commands that process the audit trail, edit the **/etc/security/audit/bincmds** file to include one or more command lines, such as the following:

```
/usr/sbin/auditcat -p -o $trail $bin

/usr/sbin/auditselect -e "event == USER_Login" \
$bin | /usr/sbin/auditpr >> /etc/log
```

   The first command line appends compressed audit bins to the audit trail file. The second line selects USER_Login records from each bin file, passes them to the `auditpr` command for formatting, and appends the records to the **/etc/log** file.

3. To enable virtual logs in the **auditbin** daemon for capturing audit records in a centralized place, such as a Virtual I/O Server (VIOS) system, add the following attribute to the bin stanza of the **/etc/security/audit/config** file:

```
bin:
        virtual_log = /dev/vlog0
```

   Note: The **/dev/vlog0** device path is an example. The real device name might be different on each client logical partition (LPAR), based on how the virtual logs are configured from an attached VIOS system.

Files

Item	Description
/usr/sbin/auditbin	Specifies the path to the **auditbin** daemon.
/audit/bin*x*	Specifies the path to the default bin collection files, with *x* indicating the bin number.
/etc/security/audit/config	Contains audit system configuration information.
/etc/security/audit/events	Contains the audit events of the system.
/etc/security/audit/objects	Contains audit events for audited objects (files).

Item	Description
/etc/security/audit/bincmds	Contains the **auditbin** backend commands.
/etc/security/audit/streamcmds	Contains the **auditstream** commands.

auditcat Command

Purpose

Writes bins of audit records.

Syntax

auditcat [**-p** | **-u**] [**-s** <*size*>] [**-d** <*pathname*>] [**-o***OutFile*] [**-r**] [*InFile*]

Description

The **auditcat** command is part of the audit subsystem, and is one of several backend commands that process the audit data records.

The **auditcat** command reads bin files of audit records from standard input or from the file specified by the *InFile* parameter. The command then processes the records and writes its output to standard output or to the file specified by the *OutFile* parameter. The output can be compressed or not, depending on the flag selected.

One major use of the command is appending compressed bin files to the end of the system audit trail file.

If the **/etc/security/audit/bincmds** file includes **$bin** as the input file, input comes from the current bin file, **bin1** or **bin2**. If the **/etc/security/audit/bincmds** file includes **$trail** as the output file, the records are written to the end of the system audit trail file.

If a bin file is not properly formed with a valid header and tail, an error is returned. See the **auditpr** command for information about audit headers and tails and the **auditbin** command for information on error recovery.

If **-s** option is mentioned with valid value then It will take the backup of the trail file and reduces it size to the zero. If the pathname is provide it will copy the backup file in that path. The backup file name will be in the following format trail. YYYYMMDDThhmmss.<random number> If the size of the **/audit** filesystem is less then freespace (**/etc/security/audit/config** set in) and **-d** specify with valid path parameter , then it will take the backup of the trail file to that path. To see the output of the different trail file, use **auditmerge** command.

Flags

Item	Description
-o *OutFile*	Specifies the audit trail file to which the **auditcat** command writes records. If you specify **$trail** as the file for the *OutFile* parameter, the **auditbin** daemon substitutes the name of the system audit trail file.
-p	Specifies that the bin files be compressed (packed) upon output. The default value specifies that the bins not be compressed.
-r	Requests recovery procedures. File names for both the *InFile* and *OutFile* parameters must be specified for recovery to occur, so the command syntax must be **auditcat -o** *OutFile* **-r** *InFile*. The command checks to see if the bin file specified for the *InFile* parameter is appended and if not, appends the bin file to the file specified by the *OutFile* parameter. If the bin file is incomplete, the **auditcat** command adds a valid tail and then appends the bin file to the file specified by the *OutFile* parameter.
-u	Specifies that compressed trail files be uncompressed upon output.

Item	Description
-s *size*	Specifies the limit on size of the trail file, after which backup of trail had to be taken . Size should be specified in units of 512-byte blocks. If size parameter is –ve or zero or any invalid value, **auditcat** will ignore flag and value. The maximum possible value is 4194303 (about 2GB of free disk space).
-d *pathname*	Pathname should be valid full directory path , where backup of the trail file needs to be taken. Incase pathname value is invalid, **auditcat** will ignore the flag and the value.

Security

Access Control

This command should grant execute (x) access to the root user and members of the audit group. The command should be setuid to the root user and have the **trusted computing base** attribute.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To configure the system to append audit bin data to the system audit trail file, add the following line to the **/etc/security/audit/bincmds** file:

```
/usr/sbin/auditcat  -o $trail $bin
```

When the **auditbin** daemon calls the **auditcat** command, the daemon replaces the **$bin** string with the path name of the current bin file, and replaces the **$trail** string with the name of the default audit trail file.

Files

Item	Description
/usr/sbin/auditcat	Specifies the path to the **auditcat** command.
/etc/security/audit/config	Contains audit system configuration information.
/etc/security/audit/events	Contains the audit events of the system.
/etc/security/audit/objects	Contains audit events for audited objects (files).
/etc/security/audit/bincmds	Contains auditbin backend commands.

auditconv Command

Purpose

Converts previous AIX Version 4 format audit bins to the AIX Version 4 format.

Syntax

auditconv *OldFile NewFile*

Description

The **auditconv** command converts audit records which were generated by previous versions of the operating system into the format used by AIX Version 4 and higher of the operating system.

Audit records are read from the file *OldFile*, and written to the file *NewFile*. Each audit record is updated with thread information, with a default thread identifier of zero.

Notes:

1. The *OldFile* and *NewFile* parameters must be different, and must not be currently in use by the audit system.
2. AIX Version 4 and higher of the operating system cannot work with pre-AIX Version 4 audit bins. Therefore, old bins must be converted using the **auditconv** command.

Security

Access Control

This command should grant execute (x) access to the root user and members of the audit group. The command should be setuid to the root user and have the **trusted computing base** attribute.

Files Accessed

Mode	File
r	**/etc/security/audit/events**
r	**/etc/passwd**
r	**/etc/group**

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

To convert the old audit file **pre_v4_auditbin**, storing the results in **converted_auditbin**, enter the following command:

```
/usr/sbin/auditconv pre_v4_auditbin converted_auditbin
```

Files

Item	Description
/usr/sbin/auditconv	Specifies the path of the **auditconv** command.
/etc/security/audit/config	Contains audit system configuration information.
/etc/security/audit/events	Contains the audit events of the system.
/etc/security/audit/objects	Contains information about audited objects (files).
/etc/security/audit/bincmds	Contains auditbin backend commands.
/etc/security/audit/streamcmds	Contains auditstream commands.

auditldap Command

Purpose

Uploads the **/etc/security/audit/config** audit configuration file to a centralized location on a Lightweight Directory Access Protocol (LDAP) server.

Syntax

auditldap [**-a**|**-u**] **-D** *bindDN* **-w** *bindPwD* [**-b** *baseDN*] [**-f** *filename*] [**-c**] [**-v**]

auditldap [**-?**]

Description

A system administrator can store the **/etc/security/audit/config** audit configuration file in a centralized location on an LDAP server by using the **auditldap** command. By sharing this configuration file, system that is operating in a similar environment can download configuration during audit start. Therefore, systems with similar security requirements can be configured the same audit configuration stored on LDAP.

Note: With the existing LDAP setup, the **auditldap** command uses the binding distinguished name (bindDN) and the binding password (bindPwD) of the LDAP client that is in the running state to store the **/etc/security/audit/config** audit configuration file on the LDAP server.

Flags

Item	Description
-a	Adds an audit configuration file to an LDAP server.
-b *baseDN*	Specifies the centralized location where the audit configuration files are stored. If you specify the *baseDN* parameter when the **/etc/security/audit/config** audit file is being uploaded, the **/etc/security/audit/config** audit file is stored in the location specified by the *baseDN* parameter. Otherwise the files are stored at the location specified by the default *baseDN* value, for example `cn=config, ou=audit,cn=aixdata`.
-c	Continues operation during error.
-D *bindDN*	Specifies the binding distinguished name that is used to connect to an LDAP server.
-f *filename*	Specifies the full path of the audit configuration file which is uploaded to an LDAP server. If you do not specify the option, the **/etc/security/audit/config** file is uploaded to the LDAP server by default.
-u	Updates an audit configuration file to the LDAP server.
-v	Displays the Verbose mode.
-w *bindPwD*	Specifies the binding password that is to write the audit configuration file into an LDAP server.
-?	Displays the usage statement of the command.

Exit Status

Item	Description
0	Success
1	Failure

Security

Only root users can run the **auditldap** command.

Examples

1. To upload the **/etc/security/audit/config** file under the `ou=audit,cn=aixdata DN`, enter the following command:

```
auditldap –u –D binddn -w secret –b ou=audit,cn=aixdata
```

2. To add the **/etc/security/audit/config** file under the `ou=audit,cn=aixdata DN`, enter the following command:

```
auditldap -a -D binddn -w secret -b ou=audit,cn=aixdata
```

Files

Item	Description
/etc/security/audit/config	Stores the audit configuration file.

auditmerge Command

Purpose

Combines multiple audit trails into a single trail.

Syntax

/usr/sbin/auditmerge [**-q**] *file* [*file* ...]

Description

The **auditmerge** command combines multiple audit trail files from potentially multiple machines into a single audit trail file. For each file with records remaining, the record that has the oldest time stamp is added to the output. If a record is found that has a negative time change, an optional warning message may be emitted. Processing continues and any such records are output with their time values unmodified.

The **auditmerge** command is also capable of adding the CPU ID values from the bin header to each output record. The CPU ID value is encoded in the bin header and bin trailer.

The **-q** flag is used to control outputting warning messages. When a record with a negative time change is first seen, a single warning message is output. That message contains the name of the file containing the record and the time difference. These messages are suppressed when the **-q** flag is entered on the command line.

Flags

Item	Description
-q	Used to control outputting warning messages.

Security

Access Control: This command should grant execute (x) access to the root user and members of the audit group. The command should be setuid to the root user and have the **trusted computing base** attribute.

Examples

1. To merge two existing audit trail files from different hosts, enter:

```
/usr/bin/auditmerge /audit/trail.calvin /audit/trail.hobbes > /audit/trail.merge
```

2. To merge two existing data files, which were preselected for different user names, enter:

```
/usr/bin/auditmerge /audit/trail.jim /audit/trail.julie > /audit/trail.both
```

3. To merge two data files without producing warnings about incorrect times, enter:

```
/usr/bin/auditmerge -q /audit/jumbled.1 /audit/jumbled.2 > /audit/jumbled.output
```

Files

Item	Description
/etc/security/audit/hosts	Contains the CPU ID to host name mappings.

auditpr Command

Purpose

Formats bin or stream audit records to a display device or printer.

Syntax

auditpr [**-i** *inputfile*] [**-t 0** | **1** | **2**] [**-m** *Message*] [**-r**] [**-v** | **-w**] [**-X**][**-h** *field*[**,***field*]*]

Description

The **auditpr** command is part of the audit subsystem. This command reads audit records, in bin or stream format, from standard input and sends formatted records to standard output.

The output format is determined by the flags that are selected. If you specify the **-m** flag, a message is displayed before each heading. Use the **-t** and **-h** flags to change the default header titles and fields and the **-v** flag to append an audit trail. The **auditpr** command searches the local **/etc/passwd** file to convert user and group IDs to names.

An example of output using default header information follows:

```
event   login   status   time                              command
        wpar name
login   dick     OK       Fri  Feb;8   14:03:57    1990   login
        Global
 . . . . . trail portion . . . . .
```

For examples of audit trails, see the **/etc/security/audit/events** file where audit trail formats are defined.

Invalid records are skipped when possible, and an error message is issued. If the command cannot recover from an error, processing stops.

The *AIX_AUDITBUFSZ* environment variable allows buffered write operation of the **auditpr** audit records. The buffered write option is useful for high-performance applications that generate many audit records.

The *AIX_AUDITBUFSZ* environment variable accepts decimal and hexadecimal values in the range 8192 bytes - 67 MB. Any other positive values outside the range of allowed values are rounded off to either the beginning of the range or the end of the range based on the nearest value. If this variable value is not set or this variable is assigned negative values or non-numerical values, the *AIX_AUDITBUFSZ* variable is ignored.

Flags

Item	Description
-h *field*[**,***field*]*	Selects the fields to display and the order in which to display them, by default **e**, **l, R**, **t**, and **c**. You can specify the following values: **e** The audit event. **l** The login name of the user. **R** The audit status. **t** The time the record was written. **c** The command name. **r** The real user name. **p** The process ID. **P** The ID of the parent process. **T** The kernel thread ID. This is local to the process; different processes may contain threads with the same thread ID. **h** The name of the host that generated the audit record. If there is no CPU ID in the audit record, the value **none** is used. If there is no matching entry for the CPU ID in the audit record, the 16 character value for the CPU ID is used instead. **i** The IDs or the names of roles of the audited process. **E** The effective privilege. **S** The effective sensitivity label (SL). **I** The effective integrity label (TL). **W** The workload partition name.
-i *inputfile*	Indicates the path to the audit trail file. If the **-i** flag is not specified, the **auditpr** command reads data from **stdin**.
-m "*Message***"**	Specifies a *Message* to be displayed with each heading. You must enclose the *Message* string in double quotation marks.
-r	Suppresses ID translation to the symbolic name.

Item	Description
-t {**0** | **1** | **2**}	Specifies when header titles are displayed. The default title consists of an optional message (see the **-m** flag) followed by the name of each column of output. **0** Ignores any title. **1** Displays a title once at the beginning of a series of records. **2** Displays a title before each record.
-v	Displays the trail of each audit record, using the format specifications in the **/etc/security/audit/events** file. The **-v** flag is mutually exclusive with the **-w** flag.
-w	Displays the trail and audit record in a single line, by using the format specified in the `/etc/security/audit/events` file. The **-w** flag is mutually exclusive with **-v** flag.
-X	Prints long user names at the end of the audit record when the **-X** flag is used with other flags that display the user names. The upper limit is determined by the **max_logname** Object Data Manager (ODM) attribute in the **PdAt** and **CuAt** object classes. If a user name is greater than the **max_logname** attribute, it is truncated to the number of characters minus 1 character, which is specified by the **max_logname** attribute.

Security

Access Control

This command should grant execute (x) access to the root user and members of the audit group. The command should be **setuid** to the root user and have the **trusted computing base** attribute.

Files Accessed

Mode	File
r	**/etc/security/audit/events**
r	**/etc/passwd**
r	**/etc/group**

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To read the system audit trail file with default header titles and fields and an audit trail, enter:

   ```
   /usr/sbin/auditpr  -v < /audit/trail
   ```

   The **/audit/trail** file must contain valid audit bins or records.
2. To format from an audit trail file all the audit events caused by user `witte`, enter:

   ```
   /usr/sbin/auditselect -e"login == witte"\
   /audit/trail | auditpr  -v
   ```

   The resulting record is formatted with the default values (**e**, **c**, **l**, **R**, and **t**) and includes a trail.

3. To read records interactively from the audit device, enter:

```
/usr/sbin/auditstream | /usr/sbin/auditpr -t0 -heRl
```

4. To enable the buffered write option for the audit records with a buffer size of 520000 bytes for auditing subsystem that is started in bin mode, enter the following command:

```
export AIX_AUDITBUFSZ=520000
/usr/sbin/auditpr -v -i /audit/trail > output
```

Files

Item	Description
/usr/sbin/auditpr	Specifies the path of the **auditpr** command.
/etc/security/audit/config	Contains audit system configuration information.
/etc/security/audit/events	Contains the audit events of the system.
/etc/security/audit/objects	Contains audit events for audited objects (files).
/etc/security/audit/bincmds	Contains auditbin backend commands.
/etc/security/audit/streamcmds	Contains auditstream commands.
/etc/security/audit/hosts	Contains the CPU ID to host name mappings.

auditselect Command

Purpose

Selects audit records for analysis according to defined criteria.

Syntax

auditselect { **-e** "*Expression*" | **-f** *File*} [**-m**] [*Trail*]

Description

The **auditselect** command is part of the audit subsystem. The command is called by the **auditbin** daemon if it is configured in the **/etc/security/audit/bincmds** file as a backend command for processing bin files.

The **auditselect** command selects audit records that match identified criteria and writes the records to standard output. With the **auditselect** command, you can filter the audit trail to obtain specific records for analysis or select specific records for long-term storage. The command takes stream or bin input from the file specified by the *Trail* parameter or from standard input. If you specify the **$bin** string as the value of the *Trail* parameter, the **auditbin** daemon substitutes the path name of the current bin file when it calls the **auditselect** command. The selection criteria can be entered as an expression or from the file specified by the **-f** flag. If the bin files are compressed, the **auditselect** command unpacks them prior to processing.

For stream data, configure both the **auditstream** command and the **auditselect** command in the **/etc/security/audit/streamcmds** file, or enter both commands from the command line.

The *AIX_AUDITBUFSZ* environment variable allows buffered write operation of the **auditselect** audit records. The buffered write option is useful for high-performance applications that generate many audit records.

The *AIX_AUDITBUFSZ* environment variable accepts decimal and hexadecimal values in the range 8192 bytes - 67 MB. Any other positive values outside the range of allowed values are rounded off to either the beginning of the range or the end of the range based on the nearest value. If this variable value is not

set or this variable is assigned negative values or non-numerical values, the *AIX_AUDITBUFSZ* variable is ignored.

Flags

Item	Description
-e "*Expression***"**	Defines the selection criteria. The *Expression* parameter consists of one or more terms joined by logical operators.
-f *File*	Specifies the *File* that contains the selection criteria.
-m	Specifies the output audit record with record extensions.

Creating Expressions

A valid expression consists of one or more terms joined by logical operators.

Logical Operators

Logical operators allow more than one term to be used in an expression. Normal precedence rules apply in evaluating expressions with more than one logical operator, and parentheses may be used to force the order of evaluation. The valid logical operators include the following:

Item	Description
&&	(And) The expression `term1 && term2` is true (selected) if both term1 and term2 are true.
||	(Or) The expression `term1 || term2` is true (selected) if either term1 or term2 is true.
!	(Not) The expression `!term1` is true (selected) if term1 is not true.

Terms

Each term of the expression has the following form:

```
Field Relational_Operator Value
```

Fields

Fields correspond to the information in the audit header of each record. Valid values for fields include the following:

Item	Description
`event`	Name of the audit event, for example, `FILE_Open`.
`command`	Name of the command that generated the audit event.

Item	Description
result	Status of the audit event. The value of the result field must be one of the following: • **OK** • **FAIL** • **FAIL_PRIV** • **FAIL_AUTH** • **FAIL_ACCESS** • **FAIL_DAC** Indicates the event failed because of a discretionary access control (DAC) denial. Access Control Lists are a form of information repository that contain data relative to the rights of access (permission) to shared resources/objects. ACLs are categorized on DAC mechanism. **FAIL** matches all other error codes.
login	ID of the login user of the process that generated the audit event.
real	ID of the real user of the process that generated the audit event.
pid	ID of the process that generated the audit event.
ppid	ID of the parent of the process that generated the audit event.
tid	ID of the kernel thread that generated the event.
time	Time of day the audit event was generated.
date	Date the audit event was generated.
host	Hostname of the machine that generated the record. The reserved name **UNKNOWN** can be used to match any machines that are not listed in the **/etc/security/audit/hosts** file.

Relational Operators

Relational operators are used to compare the field in the audit record to the specified value. Valid relational operators include:

Item	Description
==	Equal to
!=	Not equal to
<	Less than
>	Greater than
>=	Greater than or equal to
<=	Less than or equal to

Valid Terms

A valid term consists of a field, a relational operator, and a value. In addition, not all relational operators and values are valid for each field. The following are the valid combinations:

Field	Valid Operators	Valid Values
event	= =, ! =	Text string audit event name
result	= =, ! =	Text string audit status codes
command	= =, ! =	Text string command name
pid	all	Decimal integer process ID

Field	Valid Operators	Valid Values
ppid	all	Decimal integer process ID
login	all	Decimal integer user ID
login	= =, ! =	Text string user name
real	all	Decimal integer user ID
real	= =, ! =	Text string user name
tid	all	Decimal integer thread ID
time	all	String in the format specified by the current locale
date	all	String in the format specified by the current locale
host	= =, ! =	Text string host name or 16 character cpu ID
priv	= =, ! =	Privilege name
sl	= =, ! =	Sensitivity label name
tl	= =, ! =	Integrity label name
role	= =, ! =	Role name

Security

Access Control

This command should grant execute (x) access to the root user and members of the audit group. The command should be **setuid** to the root user and have the **trusted computing base** attribute.

RBAC Environment and

This command implements and can perform privileged operations. Only privileged users can run such privileged operations. To review the list of privileges and the authorizations associated with this command, refer to the **/etc/security/privcmds** database.

Examples

Configuration

1. To select bin-collected data records that match the USER_SU or USER_Login audit events, add the **auditselect** command to the **/etc/security/audit/bincmds** file by entering:

   ```
   /usr/sbin/auditselect -e "event== USER_SU || event== \
   USER_Login" $bin >> /audit/trail.login
   ```

   While auditing is enabled, the records for each initiation of a user session are read from the current bin file and written to the **/audit/trail.login** file.
2. To select stream-collected data records that match a user login that was unsuccessful, add the **auditselect** command to the **auditstream** stanza in the **/etc/security/audit/streamcmds** file by entering:

   ```
   /usr/sbin/auditstream -c authentication | \
   /usr/sbin/auditselect -e "event == \
   USER_Login && result == FAIL" | \
   /usr/sbin/auditpr -t 2 -v >> /dev/lpr2
   ```

   To produce a hardcopy audit trail, records of unsuccessful authentication events are written to the **/dev/lpr2** line printer.

Select authentication or login events

1. To search an audit trail file for all events that involve authentication errors:

```
/usr/sbin/auditselect -e "result == FAIL_AUTH"
/audit/oldtrail | /usr/sbin/auditpr -t -helt -v
```

   The records of events that were unsuccessful because authentication was denied are printed. The header titles will be printed once, followed by the event, login ID, and time fields, and then the audit trail.

2. To select audit records that are generated when `smith` logs in during prime working hours during the first week in May of 1987, enter:

```
/usr/sbin/auditselect  -f /aaa/bbb \
/audit/trail1987 | /usr/sbin/auditpr
```

   The `/aaa/bbb` file must contain the following line:

```
 command == login && login == smith &&
 time >= 08:00:00 && time <= 17:00:00 &&
 date >= 05/01/87 && date <= 05/05/87
```

String comparison

1. To compare the name of the audit event to the `USER_Login` string, enter one of the following:

```
"event == USER_Login"

"event != USER_Login"
```

2. To find out if the **passwd** command generated the audit event, use:

```
"command == passwd"
```

   To find out if the audit event was not generated by the **passwd** command, use:

```
"command != passwd"
```

3. To compare the audit status to the `OK` result string, enter:

```
"result == OK"
```

4. To compare the login or real user ID of the process that generated the audit event to a specific user ID (user ID `014` or the user name `carol`), enter one of the following:

```
"login == 014"
"login != carol"
"login == 014 || login != carol"
"real == carol"
```

5. To compare the ID of the process or the parent of the process that generated the audit event to the process ID `2006`, enter one of the following:

```
"pid == 2006"
"pid != 2006"
"ppid ==  2006"
```

   Note: Although login and real user IDs and process IDs can be compared with the inequality operators (**< =**, **> =**, **<**, **>**), it is normally unnecessary to do this.

6. To compare the time the audit event was generated to the `08:03:00` time string, enter one of the following:

```
"time == 08:03:00"
"time != 08:03:00"
"time <  08:03:00"
"time <= 08:03:00"
"time >  08:03:00"
"time >= 08:03:00"
```

Audit records are selected that fit the indicated comparison to the `08:03:00` time string. The time string must agree with the format specified by the current locale.

7. To compare the date that the audit event was generated to the `05/05/89` date string, enter one of the following:

```
"date == 05/03/89"
"date != 05/03/89"
"date <  05/03/89"
"date <= 05/03/89"
"date >  05/03/89"
"date >= 05/03/89"
```

Audit records are selected that fit the indicated comparison to the `05/05/89` date string. The date string must agree with the format specified by the current locale.

Note: The **auditselect** command does not support the *-r* flag for the recovery mode.

Buffered write option for audit records

1. To use the buffered write option for the audit records with a buffer size of 520000 bytes for auditing subsystem that is started in bin mode, enter the following command:

```
export AIX_AUDITBUFSZ=520000
/usr/sbin/auditselect -e "event== USER_SU || event==USER_Login" $bin >> /audit/trail.login
```

Files

Item	Description
/usr/sbin/auditselect	Specifies the path of the **auditselect** command.
/etc/rc	Contains the system initialization commands.
/etc/security/audit/config	Contains audit system configuration information.
/etc/security/audit/events	Contains the audit events of the system.
/etc/security/audit/objects	Contains audit events for audited objects (files).
/etc/security/audit/bincmds	Contains auditbin backend commands.
/etc/security/audit/streamcmds	Contains auditstream commands.
/etc/security/audit/hosts	Contains the CPU ID to hostname mappings.

auditstream Command

Purpose

Creates a channel for reading audit records.

Syntax

auditstream [**-m**] [**-c** *Class ...*]

Description

The **auditstream** command is part of the audit subsystem. This command reads audit records from the **/dev/audit** file (the audit device) and copies the records to standard output in binary format. You can select a subset of the audit records by specifying audit classes (defined in the **/etc/security/audit/config** file) with the **-c** flag; otherwise, all currently enabled audit classes are copied.

Audit stream data can be displayed and processed as it is generated. For example, the command output can be piped to an audit backend command for further processing or redirected to a file. Both

the **auditselect** command, which selects data records according to defined criteria, and the **auditpr** command, which formats the records for viewing or for printing, are examples of backend commands.

The **auditstream** command can be called from the command line or be configured to run multiple times as part of the audit system configuration. For information on configuring the **auditstream** command, refer to "**Setting up Auditing**" in *Security* and to the **/etc/security/audit/config** file.

Note: The **auditstream** command must be run in the background.

The *AIX_AUDITBUFSZ* environment variable allows buffered write operation of the **auditstream** audit records. The buffered write option is useful for high-performance applications that generate many audit records.

The *AIX_AUDITBUFSZ* environment variable accepts decimal and hexadecimal values in the range 8192 bytes - 67 MB. Any other positive values outside the range of allowed values are rounded off to either the beginning of the range or the end of the range based on the nearest value. If this variable value is not set or this variable is assigned negative values or non-numerical values, the *AIX_AUDITBUFSZ* variable is ignored.

Flags

Item	Description
-c *Class*	Specifies the audit classes to be copied. Each class must be configured in the **etc/security/audit/config** file as a list of comma-separated audit events. The default value is all the currently enabled audit events.
-m	Includes the processor ID, roles and privileges in each audit record.

Security

Access Control

This command should grant execute (x) access to the root user and members of the audit group. The command should be **setuid** to the root user and have the **trusted computing base** attribute.

Files Accessed

Mode	File
r	**/dev/audit**

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To configure the stream collection of audit data when the audit system is initialized, add the following to the stream stanza of the **/etc/security/audit/config** file:

   ```
   cmds  =  /etc/security/audit/streamcmds
   ```

   Then add the following to the start stanza:

   ```
   streammode=on
   ```

   Next, add to the /**etc/security/audit/streamcmds** file all the stream commands that should be executed when the auditing system is initialized. For example:

   ```
   /usr/sbin/auditstream  -c  authentication  |  \
   /usr/sbin/auditpr  -v  >  /dev/console
   ```

```
/usr/sbin/auditstream | /usr/sbin/auditselect -e \
"result == FAIL_ACCESS" | \
/usr/sbin/auditpr -t 2 -v > /dev/lpr2
```

The first command formats all records for events in the authentication class and writes them to the system console. The second command formats all records that resulted in an access denial and prints them on the printer **/dev/lp2**.

2. To record audit stream events on a line printer, enter:

```
/usr/sbin/auditstream | /usr/sbin/auditselect -e "event == \
USER_Login || event == USER_SU" | \
/usr/sbin/auditpr -v > /dev/lp0 &
```

This command formats and writes all user login and **su** events to the line printer.

3. To use the buffered write option for the audit records with a buffer size of 520000 bytes for auditing subsystem that is started in steam mode, enter the following command:

```
export AIX_AUDITBUFSZ=520000
/usr/sbin/audit start
```

Note: In stream mode, the *AIX_AUDITBUFSZ* environment variable must be set before the audit subsystem is started.

Files

Item	Description
/usr/sbin/auditstream	Specifies the path of the **auditstream** command.
/etc/rc	Contains the system startup routines.
/dev/audit	Specifies the audit device.
/etc/security/audit/config	Contains audit system configuration information.
/etc/security/audit/events	Contains the audit events of the system.
/etc/security/audit/objects	Contains audit events for audited objects (files).
/etc/security/audit/bincmds	Contains auditbin backend commands.
/etc/security/audit/streamcmds	Contains auditstream commands.
/etc/security/audit/hosts	Contains host and processor IDs.

authexec Command

Purpose

Runs a Role Based Access Control (RBAC) privileged command in a controlled manner.

Syntax

authexec *RBACcommandName*

Description

The **authexec** command runs a RBAC privileged command. When **authexec** is issued, users are authenticated against the roles defined in the **authroles** attribute for the RBAC command, *RBACcommandName*, in the RBAC privileged command database.

The **authexec** command in located in /usr/sbin/.

The user invoking **authexec** must have enough authorization to invoke the target command, *RBACcommandName*. The authenticating users should not be the same as the invoking user. The authenticating users must also have a valid non-blank password to successfully pass the authentication. No user can be authenticated more than once for any role. A maximum of sixteen roles can be configured for the RBAC privileged command.

A privileged command having the **authexec** attribute in the privileged command database cannot be run directly from shell or by using the exec subroutines in programs. Such commands have to be necessarily invoked using the **authexec** command.

This mechanism is not enforced when the command *RBACcommandName* is invoked by root in a root enabled RBAC system.

Parameters

Item	Description
RBACcommandName	Specifies the RBAC target command to run, including any flags or parameters. You must specify the absolute path of the target command, *RBACcommandName*.

Security

Access Control: All users can invoke this command.

Examples

If the command **usr/sbin/shutdown** is enabled for authenticated execution using the **authroles** attribute, then a user that is authorized to the shutdown command can run:

authexec /usr/sbin/shutdown

The following example shows the **usr/sbin/shutdown** command that is enabled for authenticated execution using the **authrole** attribute:

```
/usr/sbin/shutdown:
accessauths=aix.system.boot.shutdown
innateprivs=PV_AZ_ROOT,PV_DAC_O,PV_DAC_R,PV_DAC_W,
PV_DAC_X,PV_PROC_PRIV,PV_PROC_SIG
secflags=FSF_EPS
authroles=isso,so,sa
```

Before the **shutdown** command is run, three distinct users having one of the three roles listed in **authroles** attribute have to be authenticated. In this example, **authroles** attribute specifies the **isso**, **so**, and **sa** roles. This command requires the access authorization aix.system.boot.shutdown to invoke the **shutdown** command. This authorization is typically associated with the **so** role. A user, other than the user invoking the **shutdown** command, with the role **so** in addition to users with the **isso** and **sa** roles must authenticate to successfully issue the command.

Files

Item	Description
/etc/security/users	Contains the extended attributes of users.
/etc/security/roles	Contains the attributes of roles.
/etc/security/authorizations	Contains the attributes of authorizations
/etc/security/privcmds	Contains the attributes of RBAC privileged commands.

authrpt Command

Purpose

Reports the security capabilities of authorizations.

Syntax

authrpt [**-R***load_module*] [**-C**] [**-c** | **-f** | **-r** | **-u**] { auth1,auth2 ... }

Description

The **authrpt** command reports capability information of authorizations such as privileged commands, privileged files, role, and user information.

Either **–c**, **-f**, **-r** or **–u** flags can be specified.

When the **-c** option is specified, the privileged commands present in the **/etc/security/privcmds** database that can be executed by the authorizations is listed. The **–c** option can also be used to list the commands having ALLOW_ALL, ALLOW_GROUP and ALLOW_OWNER special authorizations.

When the **–f** option is specified, the list of privileged files present in the **/etc/security/privfiles** database that can be accessed by a user assigned the authorizations is listed.

When the **–u** option is specified, the list of users having the authorizations is displayed.

When the **–r** option is specified, the list of roles having the authorizations is listed.

The command takes a comma separated list of authorization names as input. When no option is specified, all the capability information such as commands, privileged files, roles and user information associated with the authorizations is listed.

Flags

Item	Description
-c	Specify that a report of privileged commands executable by the authorizations is to be obtained.
-f	Specify that a report of privileged file information accessible by the authorizations is to be obtained.
-u	Specify that a report of authorized users having the authorizations is to be obtained.
-r	Specify that a report of roles having the authorizations is to be obtained.
-R	Specifies the loadable module from which to obtain the report of authorization capabilities.
-C	Displays the authorization attributes in colon-separated records, as follows: `authorizaton:attribute1:attribute2: ...` `authorization1:value1:value2: ...` `authorization2:value1:value2: ...`

Exit status

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: This command should grant execute (x) access to the root user.

This command can be executed by root or an authorized user with the “aix.security.auth.list” authorization.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations.For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item
/etc/security/roles
/etc/security/authorizations
/etc/security/privcmds
/etc/security/privfiles

Examples

To report the commands associated with authorizations aix.fs and aix.system, use the following syntax:

```
authrpt –c aix.fs,aix.system
```

To report all capabilities of authorization aix.security, use the following syntax:

```
authrpt aix.security
```

To report all capabilities of authorization aix.security.user in colon separated format, use the following syntax:

```
authrpt –C aix.security.user
```

Information similar to the following appears:

```
#authorization:commands:privfiles:roles:users
aix.security.user:/usr/bin/mkuser,
/usr/bin/chuser:/etc/csh.cshrc,
/etc/csh.login:role1:Bob,Simon
```

authqry Command

Purpose

Queries the usage of authorizations over a time period.

Syntax

authqry { **-c** [**-s**] | **-q** [**-F** *<trailListFile>*] [**-t** *<time_period_in_days>*] } *user*

Description

The **authqry** command queries information about the authorizations used by a user over a specified time frame.

When the **–c** option is specified, the user is configured for the auditing of role and authorization information. A class **rbacqry** is added to the **/etc/security/audit/config** file with events for auditing authorizations and roles. If the user is already being audited (user entry present in the configuration file), then the **rbacqry** class is added to the user. Otherwise the username is added to the **/etc/security/audit/config** with the **rbacqry** class parameter.

When the **-s** option is specified, the auditing subsystem is started / restarted.

When the **–q** option is specified, the audit data is queried for authorization information.

When the **–t** option is specified, the usage of authorizations from the date (specified through the **–t** option) to the current system date is queried and obtained. Without **–t** option, authorization usage over the period from which auditing was enabled for that user is obtained. The command displays the entire set of authorizations used during this time frame.

Note: The **authqry** command makes use of the auditing feature in AIX. For the **authqry** command to work as expected, auditing must be turned on, audit configuration for the user must be enabled and a time frame must be specified in days.

Flags

Item	Description
-c	Specifies to configure the user for auditing of authorization usage.
-s	Start auditing subsystem if it is turned off. Restart if already turned on.
-q	Specifies to query audit data for authorization usage over a specified time period.
-F	The **–F** option reads the names of the audit trails to obtain audit information from the *trailListFile*. The names of audit trail files should be one per line of text. If the **–F** option is not specified, the system **/audit/trail** file is taken by default as the file to obtain audit information from.
-t	Specify the number of days from the current date to get the authorization usage.

Exit status

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: This command should grant execute (x) access to only the root user.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations.For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/etc/security/authorizations	
/audit/trail	

Examples

To query authorizations by Bob, use the following syntax:

```
authqry -q Bob
```

To query authorizations used by Simon for the past 20 days, use the following syntax:

```
authqry -q -t 20 Simon
```

autoconf6 Command

Purpose

Automatically configures IPv6 network interfaces at boot time.

Syntax

autoconf6 [**-a**] [**-A**] [**-i**] [**-s**] [**-6**] [**-M**] [**-O**] [**-R**] [**-c**] [**-v**] [**-m** *main_interface*] [*interface_name* ...]

Description

The **autoconf6** command is used at boot time to assign link-local addresses to ND-capable network interfaces. The **autoconf6** command initializes also the loopback interface, the automatic tunnels if needed, and adds some needed routes. It can also be used at any time to set link-local addresses and automatic tunnelling on newly configured ethernet-like interfaces.

Flags

Item	Description
-a	Configures and turns up all acceptable interfaces that are already configured with IPv4.
-A	Configures and turns up all acceptable interfaces.
-i	Configures and turns up the interfaces in the argument list. Without the **-a** and **-i** flags only the interfaces already up are configured.
-m *main_interface*	Specifies the main interface. You can also use the **no** command with the argument, **main_if6**.
-s	Installs the SIT interfaces and IPv4-compatible programs. Without this flag, the SIT interfaces are configured only if an SIT interface is already up.
-6	The SIT interface and IPv4-compatible interoperability are not installed or modified.
-M	(Debug) Do not modify existing IPv6 multicast routes.
-O	(Debug) Do not configure the loopback interface.
-R	(Debug) Do not install a default IPv6 route.
-c	Old compatibility flag for those who have bad LL addresses.
-v	Verbose output. The program displays what it is doing and/or what it is failing.

Item	Description
interface_name	Specifies the names of the interfaces that should be configured. This is used with the **-i** flag. If the **-i** flag is given and no interface_names are specified, no interfaces are configured. If an interface_name is given and the **-i** flag is not specified, a usage message is displayed. If ibX is specified as the interface name, the ibX interface is configured with an IPv6 address based on the EUI-64 for the InfiniBand port. To use the ibX interface with the **autoconf6** command, the ibX interface must be previously configured with an IPv4 address.

Messages

Messages indicate the different actions done and/or problems encountered by **autoconf6**.

automount Daemon

Purpose

Mounts automatic mount points.

Syntax

/usr/sbin/automount [**-m**] [**-n**] [**-v**] [**-t** *duration*] [**-i** *interval*] [**-f** *file*] [**-s** *timeout*] [**-D** *name=value*] ... [**-d** *value*]

Description

The **automount** command is used as an administration tool for **AutoFS**. It installs **AutoFS** mount points and associates an **automount** map with each mount point. The **AutoFS** file system monitors attempts to access directories within it and notifies the **automountd** daemon. The daemon uses the map to locate a file system, which it then mounts at the point of reference within the **AutoFS** file system.

The previous **automount** behavior can be specified if the **COMPAT_AUTOMOUNT** environment variable is set to any value before running the **automount** command. The current behavior became the default behavior in AIX 5.0.

If the file system is not accessed within an appropriate interval (ten minutes by default), the **automountd** daemon unmounts the file system.

If the **automountd** daemon has not been started the **automount** command attempts to start it using **SRC**.

Maps

Automount maps specify the mount points to be automatically mounted when accessed, and what should be mounted over those mount points. The **/etc/auto_master** map file specifies the initial mount points, known as *keys*, and their corresponding maps that determine which remote filesystem is mounted over it. The format of the **/etc/auto_master** file is:

```
/key    map
```

Note: The **/etc/auto_master** file is only read when the **automount** command is initially executed. Changes to it will not take effect until the **automount** command is run again.

The most common maps are direct maps, indirect maps, and host maps.

Direct maps require a special key (/-) in the **/etc/auto_master** file, and their map is a file with the following format:

```
/directkey     [-options]    server:/dir
```

When a user accesses the **/directkey** directory, the **automountd** daemon will mount **server:/dir** over **/directkey**.

Indirect maps have the following format:

```
indirectkey    [-options]    server:/dir
```

When a user accesses the **/key/indirectkey** directory, the **automountd** daemon will mount **server:/dir** over **/key/indirectkey**.

Host maps require a special map (-hosts) in the **/etc/auto_master** file. The **automountd** daemon will create a subdirectory under the **/key** directory for every server listed in the **/etc/hosts** file. When a user accesses the **/key/server** directory, the **automountd** daemon will mount the server's exported directories over the **/key/server** directory.

Alternate Map Locations

Automount maps might also be located on NIS and LDAP servers. The **automount** command will look for maps as files on the local system by default, unless the automount entry in the **/etc/irs.conf** file is changed. For example:

```
automount nis_ldap
```

It is possible to specify more than one name service, in the order that they will be used, by using a whitespace separated list. For example, to indicate that LDAP maps should be used first, followed by local files, the automount entry would be the following:

```
automount nis_ldap files
```

The valid values for the automount entry are `files`, `nis`, and `nis_ldap`.

Flags

Item	Description
-d *value*	Specifies the debug level of the autofs extension and **automount** daemon.
-D *name=value*	Specifies an environment variable and its value. You can specify multiple environment variables by using the **-D** flag multiple times.
-f *file*	Specifies a new master map file to use. The default is **/etc/auto_master**.
-i *Interval*	Specifies the amount of time, in seconds, that an inactive autofs mounted directory exists.
-m	Specifies not to search NIS for automount maps.
-n	Specifies the nobrowse option.
-s *timeout*	Specifies the amount of time, in seconds, before a new process is forked off if a mount takes too long. The minimum value is 30.
-t *Duration*	Specifies the amount of time, in seconds, that the auto unmount process sleeps before it starts to work again. The minimum value is 21. The default value is 120. The maximum value is 600.
-v	Displays on standard output verbose status and warning messages.

Examples

1. To specify the **`LocalOpts`**, **`LocalCaching`**, and **`Server`** environment variables for automatic mounting of mount points, enter the following command:

   ```
   automount -D LocalOpts=-rsize=16384,wsize=16384,timeo=15    \
   -D LocalCaching=-rsize=16384,wsize=16384,timeo=15 -D Server=autoserver
   ```

2. To use a master map file (`/etc/myFile`) instead of the default file (`/etc/auto_master`), enter the following command:

```
automount -f /etc/myFile
```

3. To set the interval time to 5 minutes, the timeout value to 30 seconds, and the duration time to one minute for the **automount** daemon, enter the following command:

```
automount -i 300 -s 30 -t 60
```

Files

Item	Description
/etc/auto_master	The default map file used to create the initial automount keys.
/etc/hosts	Specifies servers that will be used in automount host maps.
/etc/irs.conf	Specifies the location of the automount maps.

automountd Daemon

Purpose

AutoFS mount and unmount daemon.

Syntax

/usr/sbin/automountd [**-n**] [**-T**] [**-v**] [**-D** *name*=*value*]

Description

The **automountd** daemon is an RPC server that processes and answers requests from the local AutoFS filesystem kernel extension. It uses local files or name service maps to locate file systems to be mounted.

Maps

For a description on map files see the information on **Maps** in the **automount** daemon.

Flags

Item	Description
-D*name*=*Value*	Assigns a value to the indicated **automountd** daemon environment variable.
-n	Sets the **nobrowse** option on all maps by default.
-T	Traces RPC server calls, displaying it on standard output.
-v	Displays on standard output verbose status and warning messages.

autopush Command

Purpose

Configures lists of automatically **pushed STREAMS modules**.

Syntax

autopush -f *File*

autopush -r -M *Major* **-m** *Minor*

autopush -g -M *Major* **-m** *Minor*

Description

The **autopush** command configures the list of modules to be automatically pushed onto the stream when a device is opened. It can also remove a previous setting or obtain information on a setting.

Flags

Item	Description
-f *File*	Sets up the **autopush** configuration for each driver according to the information stored in the specified file. The file specified by the *File* parameter consists of lines consisting of at least four fields per line. Each field is separated by a character space as shown in the following example: `maj_ min_ last_min_ mod1 mod2 . . . modn` The first three fields are integers that specify the major device number, minor device number, and last minor device number. The subsequent fields represent the names of modules. If the value of the `min_` field is -1, then all minor devices of a major driver specified by the `maj_` field are configured and the value of the `last_min_` field is ignored. If the value of the `last_min_` field is 0, then only a single minor device is configured. To configure a range of minor devices for a particular major, the value of the `min_` field must be less than the value of the `last_min_` field. The last fields of a line in the **autopush** file represent the list of module names. Each module name is separated by a character space. The maximum number of modules that can be automatically pushed on a stream is eight, and they are pushed onto the stream in the order they are listed. Comment lines start with a # (pound sign).
-r	Removes the previous configuration setting of a particular major and minor device number.
-g	Obtains the current configuration setting of a particular major and minor device number. It also returns the starting minor device number if the request corresponds to a setting of a range.
-M *Major*	Specifies a major device number.
-m *Minor*	Specifies a minor device number.

This operating system provides an enhancement to the **autopush** command that makes it easier to specify major numbers. The name of a driver can be specified instead of its major number anywhere the major number is normally used.

Parameters

Item	Description
File	Contains at least the major device number, minor device number, last minor device number and modules.
Major	Specifies a major device number.
Minor	Specifies a minor device number.

Examples

1. To configure a list of automatically pushed Streams modules, type:

```
autopush -f File
```

2. To remove the previous configuration, type:

```
autopush -r -M Major -m Minor
```

3. To show the current configuration, type:

```
autopush -g -M Major -m Minor
```

awk Command

Purpose

Finds lines in files that match a pattern and performs specified actions on those lines.

Syntax

awk [**-u**] [**-F** *Ere*] [**-v** *Assignment*] ... { **-f** *ProgramFile* | **'***Program***'** } [[*File* ... | *Assignment* ...]] ...

Description

The **awk** command uses a set of user-supplied instructions to compare a set of files, one line at a time, to extended regular expressions supplied by the user. Then actions are performed upon any line that matches the extended regular expressions.

The pattern searching of the **awk** command is more general than that of the **grep** command, and it allows the user to perform multiple actions on input text lines. The **awk** command programming language requires no compiling, and allows the user to use variables, numeric functions, string functions, and logical operators.

The **awk** command is affected by the **LANG**, **LC_ALL**, **LC_COLLATE**, **LC_CTYPE**, **LC_MESSAGES**, **LC_NUMERIC**, **NLSPATH**, and **PATH** environment variables.

The following topics are covered in this article:

Input for the awk Command

The **awk** command takes two types of input: input text files and program instructions.

Input Text Files

Searching and actions are performed on input text files. The files are specified by:

- Specifying the *File* variable on the command line.
- Modifying the special variables **ARGV** and **ARGC**.
- Providing standard input in the absence of the *File* variable.

If multiple files are specified with the *File* variable, the files are processed in the order specified.

Program Instructions

Instructions provided by the user control the actions of the **awk** command. These instructions come from either the `*Program*' variable on the command line or from a file specified by the **-f** flag together with the *ProgramFile* variable. If multiple program files are specified, the files are concatenated in the order specified and the resultant order of instructions is used.

Output for the awk Command

The **awk** command produces three types of output from the data within the input text file:

- Selected data can be printed to standard output, without alteration to the input file.
- Selected portions of the input file can be altered.
- Selected data can be altered and printed to standard output, with or without altering the contents of the input file.

All of these types of output can be performed on the same file. The programming language recognized by the **awk** command allows the user to redirect output.

File Processing with Records and Fields

Files are processed in the following way:

1. The **awk** command scans its instructions and executes any actions specified to occur before the input file is read.

   The **BEGIN** statement in the **awk** programming language allows the user to specify a set of instructions to be done before the first record is read. This is particularly useful for initializing special variables.

2. One record is read from the input file.

   A record is a set of data separated by a record separator. The default value for the record separator is the new-line character, which makes each line in the file a separate record. The record separator can be changed by setting the **RS** special variable.

3. The record is compared against each pattern specified by the **awk** command's instructions.

   The command instructions can specify that a specific field within the record be compared. By default, fields are separated by white space (blanks or tabs). Each field is referred to by a field variable. The first field in a record is assigned the **$1** variable, the second field is assigned the **$2** variable, and so forth. The entire record is assigned to the **$0** variable. The field separator can be changed by using the **-F** flag on the command line or by setting the **FS** special variable. The **FS** special variable can be set to the values of: blank, single character, or extended regular expression.

4. If the record matches a pattern, any actions associated with that pattern are performed on the record.
5. After the record is compared to each pattern, and all specified actions are performed, the next record is read from input; the process is repeated until all records are read from the input file.
6. If multiple input files have been specified, the next file is then opened and the process repeated until all input files have been read.
7. After the last record in the last file is read, the **awk** command executes any instructions specified to occur after the input processing.

   The **END** statement in the **awk** programming language allows the user to specify actions to be performed after the last record is read. This is particularly useful for sending messages about what work was accomplished by the **awk** command.

The awk Command Programming Language

The **awk** command programming language consists of statements in the form:

Pattern { *Action* }

If a record matches the specified pattern, or contains a field which matches the pattern, the associated action is then performed. A pattern can be specified without an action, in which case the entire line containing the pattern is written to standard output. An action specified without a pattern is performed for every input record.

Patterns

There are four types of patterns used in the **awk** command language syntax:

- Regular Expressions
- Relational Expressions
- Combinations of Patterns
- BEGIN and END Patterns.

Regular Expressions

The extended regular expressions used by the **awk** command are similar to those used by the **grep** or **egrep** command. The simplest form of an extended regular expression is a string of characters enclosed in slashes. For an example, suppose a file named `testfile` had the following contents:

```
smawley, andy
smiley, allen
smith, alan
smithern, harry
smithhern, anne
smitters, alexis
```

Entering the following command line:

```
awk '/smi/' testfile
```

would print to standard output of all records that contained an occurrence of the string `smi`. In this example, the program `'/smi/'` for the **awk** command is a pattern with no action. The output is:

```
smiley, allen
smith, alan
smithern, harry
smithhern, anne
smitters, alexis
```

The following special characters are used to form extended regular expressions:

Character	Function
+	Specifies that a string matches if one or more occurrences of the character or extended regular expression that precedes the **+** (plus) are within the string. The command line: `awk '/smith+ern/' testfile` prints to standard output any record that contained a string with the characters `smit`, followed by one or more h characters, and then ending with the characters `ern`. The output in this example is: `smithern, harry` `smithhern, anne`

Character	Function
?	Specifies that a string matches if zero or one occurrences of the character or extended regular expression that precedes the ? (question mark) are within the string. The command line: `awk '/smith?/' testfile` prints to standard output of all records that contain the characters `smit`, followed by zero or one instance of the h character. The output in this example is: `smith, alan` `smithern, harry` `smithhern, anne` `smitters, alexis`
|	Specifies that a string matches if either of the strings separated by the | (vertical line) are within the string. The command line: `awk '/allen` `|` `alan /' testfile` prints to standard output of all records that contained the string `allen` or `alan`. The output in this example is: `smiley, allen` `smith, alan`
( )	Groups strings together in regular expressions. The command line: `awk '/a(ll)?(nn)?e/' testfile` prints to standard output of all records with the string ae or `alle` or anne or `allnne`. The output in this example is: `smiley, allen` `smithhern, anne`
{*m***}**	Specifies that a string matches if exactly *m* occurrences of the pattern are within the string. The command line: `awk '/l{2}/' testfile` prints to standard output `smiley, allen`
{*m***,}**	Specifies that a string matches if at least *m* occurrences of the pattern are within the string. The command line: `awk '/t{2,}/' testfile` prints to standard output: `smitters, alexis`

Character	Function
{*m, n***}**	Specifies that a string matches if between *m* and *n*, inclusive, occurrences of the pattern are within the string (where *m* <= *n*). The command line: `awk '/er{1, 2}/' testfile` prints to standard output: `smithern, harry` `smithern, anne` `smitters, alexis`
[*String***]**	Signifies that the regular expression matches any characters specified by the *String* variable within the square brackets. The command line: `awk '/sm[a-h]/' testfile` prints to standard output of all records with the characters `sm` followed by any character in alphabetical order from a to h. The output in this example is: `smawley, andy`
[^ *String***]**	A ^ (caret) within the [ ] (square brackets) and at the beginning of the specified string indicates that the regular expression *does not* match any characters within the square brackets. Thus, the command line: `awk '/sm[^a-h]/' testfile` prints to standard output: `smiley, allen` `smith, alan` `smithern, harry` `smithhern, anne` `smitters, alexis`
~,!~	Signifies a conditional statement that a specified variable matches (tilde) or does not match (tilde, exclamation point) the regular expression. The command line: `awk '$1 ~ /n/' testfile` prints to standard output of all records whose first field contained the character `n`. The output in this example is: `smithern, harry` `smithhern, anne`
^	Signifies the beginning of a field or record. The command line: `awk '$2 ~ /^h/' testfile` prints to standard output of all records with the character h as the first character of the second field. The output in this example is: `smithern, harry`

Character	Function
$	Signifies the end of a field or record. The command line: `awk '$2 ~ /y$/' testfile` prints to standard output of all records with the character y as the last character of the second field. The output in this example is: `smawley, andy` `smithern, harry`
. (period)	Signifies any one character except the terminal new-line character at the end of a space. The command line: `awk '/a..e/' testfile` prints to standard output of all records with the characters a and e separated by two characters. The output in this example is: `smawley, andy` `smiley, allen` `smithhern, anne`
***** (asterisk)	Signifies zero or more of any characters. The command line: `awk '/a.*e/' testfile` prints to standard output of all records with the characters a and e separated by zero or more characters. The output in this example is: `smawley, andy` `smiley, allen` `smithhern, anne` `smitters, alexis`
**** (backslash)	The escape character. When preceding any of the characters that have special meaning in extended regular expressions, the escape character removes any special meaning for the character. For example, the command line: `/a\/\//` would match the pattern a //, since the backslashes negate the usual meaning of the slash as a delimiter of the regular expression. To specify the backslash itself as a character, use a double backslash. See the following item on escape sequences for more information on the backslash and its uses.

Recognized Escape Sequences

The **awk** command recognizes most of the escape sequences used in C language conventions, as well as several that are used as special characters by the **awk** command itself. The escape sequences are:

Escape Sequence	Character Represented
\"	\" (double-quotation) mark
\/	/ (slash) character
ddd	Character whose encoding is represented by a one-, two- or three-digit octal integer, where *d* represents an octal digit
\\	\ (backslash) character

Escape Sequence	Character Represented
\a	Alert character
\b	Backspace character
\f	Form-feed character
\n	New-line character (see following note)
\r	Carriage-return character
\t	Tab character
\v	Vertical tab.

Note: Except in the **gsub**, **match**, **split**, and **sub** built-in functions, the matching of extended regular expressions is based on input records. Record-separator characters (the new-line character by default) cannot be embedded in the expression, and no expression matches the record-separator character. If the record separator is not the new-line character, then the new-line character can be matched. In the four built-in functions specified, matching is based on text strings, and any character (including the record separator) can be embedded in the pattern so that the pattern matches the appropriate character. However, in all regular-expression matching with the **awk** command, the use of one or more NULL characters in the pattern produces undefined results.

Relational Expressions

The relational operators < (less than), > (greater than), <= (less than or equal to), >= (greater than or equal to), = = (equal to), and ! = (not equal to) can be used to form patterns. For example, the pattern:

```
$1 < $4
```

matches records where the first field is less than the fourth field. The relational operators also work with string values. For example:

```
$1 =! "q"
```

matches all records where the first field is not a q. String values can also be matched on collation values. For example:

```
$1 >= "d"
```

matches all records where the first field starts with a character that is a, b, c, or d. If no other information is given, field variables are compared as string values.

Combinations of Patterns

Patterns can be combined using three options:

- Ranges are specified by two patterns separated with a , (comma). Actions are performed on every record starting with the record that matches the first pattern, and continuing through and including the record that matches the second pattern. For example:

  ```
  /begin/,/end/
  ```

  matches the record containing the string `begin`, and every record between it and the record containing the string `end`, including the record containing the string `end`.
- Parentheses ( ) group patterns together.
- The boolean operators || (or), && (and), and ! (not) combine patterns into expressions that match if they evaluate true, otherwise they do not match. For example, the pattern:

  ```
  $1 == "al" && $2 == "123"
  ```

matches records where the first field is `a1` and the second field is 123.

BEGIN and END Patterns

Actions specified with the **BEGIN** pattern are performed before any input is read. Actions specified with the **END** pattern are performed after all input has been read. Multiple **BEGIN** and **END** patterns are allowed and processed in the order specified. An **END** pattern can precede a **BEGIN** pattern within the program statements. If a program consists only of **BEGIN** statements, the actions are performed and no input is read. If a program consists only of **END** statements, all the input is read prior to any actions being taken.

Actions

There are several types of action statements:

- Action Statements
- Built-in Functions
- User-Defined Functions
- Conditional Statements
- Output Actions

Action Statements

Action statements are enclosed in { } (braces). If the statements are specified without a pattern, they are performed on every record. Multiple actions can be specified within the braces, but must be separated by new-line characters or ; (semicolons), and the statements are processed in the order they appear. Action statements include:

Arithmetical Statements
The mathematical operators + (plus), - (minus), / (division), ^ (exponentiation), * (multiplication), % (modulus) are used in the form:

```
Expression Operator Expression
```

Thus, the statement:

```
$2 = $1 ^ 3
```

assigns the value of the first field raised to the third power to the second field.

Unary Statements
The unary **-** (minus) and unary **+** (plus) operate as in the C programming language:

```
+Expression or -Expression
```

Increment and Decrement Statements
The pre-increment and pre-decrement statements operate as in the C programming language:

```
++Variable or --Variable
```

The post-increment and post-decrement statements operate as in the C programming language:

```
Variable++ or Variable--
```

Assignment Statements
The assignment operators += (addition), -= (subtraction), /= (division), and *= (multiplication) operate as in the C programming language, with the form:

```
Variable += Expression
```

```
Variable -= Expression
```

```
Variable /= Expression
```

```
Variable *= Expression
```

For example, the statement:

```
$1 *= $2
```

multiplies the field variable **$1** by the field variable **$2** and then assigns the new value to **$1**.

The assignment operators ^= (exponentiation) and %= (modulus) have the form:

```
Variable1^=Expression1
```

AND

```
Variable2%=Expression2
```

and they are equivalent to the C programming language statements:

```
Variable1=pow(Variable1, Expression1)
```

AND

```
Variable2=fmod(Variable2, Expression2)
```

where pow is the **pow** subroutine and fmod is the **fmod** subroutine.

String Concatenation Statements

String values can be concatenated by stating them side by side. For example:

```
$3 = $1 $2
```

assigns the concatenation of the strings in the field variables **$1** and **$2** to the field variable **$3**.

Built-In Functions

The **awk** command language uses arithmetic functions, string functions, and general functions. The close Subroutine statement is necessary if you intend to write a file, then read it later in the same program.

Arithmetic Functions

The following arithmetic functions perform the same actions as the C language subroutines by the same name:

Item	Description
atan2(*y*, *x* **)**	Returns arctangent of *y*/*x*.
cos(*x* **)**	Returns cosine of *x*; *x* is in radians.
sin(*x* **)**	Returns sin of *x*; *x* is in radians.
exp(*x* **)**	Returns the exponential function of *x*.
log(*x* **)**	Returns the natural logarithm of *x*.
sqrt(*x* **)**	Returns the square root of *x*.
int(*x* **)**	Returns the value of *x* truncated to an integer.
rand( )	Returns a random number *n*, with 0 <= *n* < 1.
srand([*Expr*] **)**	Sets the seed value for the **rand** function to the value of the *Expr* parameter, or use the time of day if the *Expr* parameter is omitted. The previous seed value is returned.

String Functions

The string functions are:

Item	Description
gsub(*Ere*, *Repl*, [*In*] **)**	Performs exactly as the **sub** function, except that all occurrences of the regular expression are replaced.
sub(*Ere*, *Repl*, [*In*] **)**	Replaces the first occurrence of the extended regular expression specified by the *Ere* parameter in the string specified by the *In* parameter with the string specified by the *Repl* parameter. The **sub** function returns the number of substitutions. An & (ampersand) appearing in the string specified by the *Repl* parameter is replaced by the string in the *In* parameter that matches the extended regular expression specified by the *Ere* parameter. If no *In* parameter is specified, the default value is the entire record (the **$0** record variable).
index(*String1*, *String2* **)**	Returns the position, numbering from 1, within the string specified by the *String1* parameter where the string specified by the *String2* parameter occurs. If the *String2* parameter does not occur in the *String1* parameter, a 0 (zero) is returned.
length [(*String***)]**	Returns the length, in characters, of the string specified by the *String* parameter. If no *String* parameter is given, the length of the entire record (the **$0** record variable) is returned.
blength [(*String***)]**	Returns the length, in bytes, of the string specified by the *String* parameter. If no *String* parameter is given, the length of the entire record (the **$0** record variable) is returned.
substr(*String*, *M*, [*N*] **)**	Returns a substring with the number of characters specified by the *N* parameter. The substring is taken from the string specified by the *String* parameter, starting with the character in the position specified by the *M* parameter. The *M* parameter is specified with the first character in the *String* parameter as number 1. If the *N* parameter is not specified, the length of the substring will be from the position specified by the *M* parameter until the end of the *String* parameter.
match(*String*, *Ere* **)**	Returns the position, in characters, numbering from 1, in the string specified by the *String* parameter where the extended regular expression specified by the *Ere* parameter occurs, or else returns a 0 (zero) if the *Ere* parameter does not occur. The **RSTART** special variable is set to the return value. The **RLENGTH** special variable is set to the length of the matched string, or to -1 (negative one) if no match is found.
split(*String*, *A*, [*Ere*] **)**	Splits the string specified by the *String* parameter into array elements *A*[1], *A*[2], . . ., *A*[*n*], and returns the value of the *n* variable. The separation is done with the extended regular expression specified by the *Ere* parameter or with the current field separator (the **FS** special variable) if the *Ere* parameter is not given. The elements in the *A* array are created with string values, unless context indicates a particular element should also have a numeric value.
tolower(*String* **)**	Returns the string specified by the *String* parameter, with each uppercase character in the string changed to lowercase. The uppercase and lowercase mapping is defined by the **LC_CTYPE** category of the current locale.
toupper(*String* **)**	Returns the string specified by the *String* parameter, with each lowercase character in the string changed to uppercase. The uppercase and lowercase mapping is defined by the **LC_CTYPE** category of the current locale.
sprintf(*Format***,** *Expr***,** *Expr***,** . . . **)**	Formats the expressions specified by the *Expr* parameters according to the **printf** subroutine format string specified by the *Format* parameter and returns the resulting string.

General Functions

The general functions are:

Item	Description
close(*Expression* **)**	Close the file or pipe opened by a **print** or **printf** statement or a call to the **getline** function with the same string-valued *Expression* parameter. If the file or pipe is successfully closed, a 0 is returned; otherwise a non-zero value is returned. The **close** statement is necessary if you intend to write a file, then read the file later in the same program.
system(*Command* **)**	Executes the command specified by the *Command* parameter and returns its exit status. Equivalent to the **system** subroutine.
Expression | **getline** [*Variable*]	Reads a record of input from a stream piped from the output of a command specified by the *Expression* parameter and assigns the value of the record to the variable specified by the *Variable* parameter. The stream is created if no stream is currently open with the value of the *Expression* parameter as its command name. The stream created is equivalent to one created by a call to the **popen** subroutine with the *Command* parameter taking the value of the *Expression* parameter and the *Mode* parameter set to a value of **r**. Each subsequent call to the **getline** function reads another record, as long as the stream remains open and the *Expression* parameter evaluates to the same string. If a *Variable* parameter is not specified, the **$0** record variable and the **NF** special variable are set to the record read from the stream.
getline [*Variable*] **<** *Expression*	Reads the next record of input from the file named by the *Expression* parameter and sets the variable specified by the *Variable* parameter to the value of the record. Each subsequent call to the **getline** function reads another record, as long as the stream remains open and the *Expression* parameter evaluates to the same string. If a *Variable* parameter is not specified, the **$0** record variable and the **NF** special variable are set to the record read from the stream.
getline [*Variable*]	Sets the variable specified by the *Variable* parameter to the next record of input from the current input file. If no *Variable* parameter is specified, **$0** record variable is set to the value of the record, and the **NF**, **NR**, and **FNR** special variables are also set.

Note: All forms of the **getline** function return 1 for successful input, zero for end of file, and -1 for an error.

User-Defined Functions

User-defined functions are declared in the following form:

```
function Name (Parameter, Parameter,...)  { Statements }
```

A function can be referred to anywhere in an **awk** command program, and its use can precede its definition. The scope of the function is global.

Function parameters can be either scalars or arrays. Parameter names are local to the function; all other variable names are global. The same name should not be used for different entities; for example, a parameter name should not be duplicated as a function name, or special variable. Variables with global scope should not share the name of a function. Scalars and arrays should not have the same name in the same scope.

The number of parameters in the function definition does not have to match the number of parameters used when the function is called. Excess formal parameters can be used as local variables. Extra scalar parameters are initialized with a string value equivalent to the empty string and a numeric value of 0 (zero); extra array parameters are initialized as empty arrays.

When invoking a function, no white space is placed between the function name and the opening parenthesis. Function calls can be nested and recursive. Upon return from any nested or recursive function call, the values of all the calling function's parameters shall be unchanged, except for array parameters passed by reference. The **return** statement can be used to return a value.

Within a function definition, the new-line characters are optional before the opening { (brace) and after the closing } (brace).

An example of a function definition is:

```
function average ( g,n)
  {
        for (i in g)
           sum=sum+g[i]
        avg=sum/n
        return avg
  }
```

The function average is passed an array, g, and a variable, n, with the number of elements in the array. The function then obtains an average and returns it.

Conditional Statements

Most conditional statements in the **awk** command programming language have the same syntax and function as conditional statements in the C programming language. All of the conditional statements allow the use of { } (braces) to group together statements. An optional new-line can be used between the expression portion and the statement portion of the conditional statement, and new-lines or ; (semicolon) are used to separate multiple statements in { } (braces). Six conditional statements in C language are:

Item	Description
if	Requires the following syntax: **if** (*Expression*) { *Statement }* [**else** *Action*]
while	Requires the following syntax: **while** (*Expression*) { *Statement* }
for	Requires the following syntax: **for** (*Expression* ; *Expression* ; *Expression*) { *Statement* }
break	Causes the program loop to be exited when the **break** statement is used in either a **while** or **for** statement.
continue	Causes the program loop to move to the next iteration when the **continue** statement is used in either a **while** or **for** statement.

Five conditional statements in the **awk** command programming language that do not follow C-language rules are:

Item	Description
for...in	Requires the following syntax: **for** (*Variable* **in** *Array*) { *Statement* } The **for**...**in** statement sets the *Variable* parameter to each index value of the *Array* variable, one index at a time and in no particular order, and performs the action specified by the *Statement* parameter with each iteration. See the **delete** statement for an example of a **for**...**in** statement.
if...in	Requires the following syntax: **if** (*Variable* **in** *Array*) { *Statement* } The **if**...**in** statement searches for the existence of the *Array* element. The statement is performed if the *Array* element is found.

Item	Description
delete	Requires the following syntax: **delete** *Array* [*Expression*] The **delete** statement deletes both the array element specified by the *Array* parameter and the index specified by the *Expression* parameter. For example, the statements: `for (i in g)` `  delete g[i];` would delete every element of the g[] array.
exit	Requires the following syntax: **exit** [*Expression*] The **exit** statement first invokes all **END** actions in the order they occur, then terminates the **awk** command with an exit status specified by the *Expression* parameter. No subsequent **END** actions are invoked if the **exit** statement occurs within an **END** action.
#	Requires the following syntax: **#** *Comment* The **#** statement places comments. Comments should always end with a new-line but can begin anywhere on a line.
next	Stops the processing of the current input record and proceeds with the next input record.

Output Statements

Two output statements in the **awk** command programming language are:

Item	Description
print	Requires the following syntax: **print** [*ExpressionList*] [*Redirection*] [*Expression*] The **print** statement writes the value of each expression specified by the *ExpressionList* parameter to standard output. Each expression is separated by the current value of the **OFS** special variable, and each record is terminated by the current value of the **ORS** special variable. The output can be redirected using the *Redirection* parameter, which can specify the three output redirections with the > (greater than), >> (double greater than), and the | (pipe). The *Redirection* parameter specifies how the output is redirected, and the *Expression* parameter is either a path name to a file (when *Redirection* parameter is > or >>) or the name of a command (when the *Redirection* parameter is a |).

Item	Description
printf	Requires the following syntax: **printf** *Format* [, *ExpressionList*] [*Redirection*] [*Expression*] The **printf** statement writes to standard output the expressions specified by the *ExpressionList* parameter in the format specified by the *Format* parameter. The **printf** statement functions exactly like the **printf** command, except for the c conversion specification (%c). The *Redirection* and *Expression* parameters function the same as in the **print** statement. For the c conversion specification: if the argument has a numeric value, the character whose encoding is that value will be output. If the value is zero or is not the encoding of any character in the character set, the behavior is undefined. If the argument does not have a numeric value, the first character of the string value will be output; if the string does not contain any characters the behavior is undefined.

Note: If the *Expression* parameter specifies a path name for the *Redirection* parameter, the *Expression* parameter should be enclosed in double quotes to insure that it is treated as a string.

Variables

Variables can be scalars, field variables, arrays, or special variables. Variable names cannot begin with a digit.

Variables can be used just by referencing them. With the exception of function parameters, they are not explicitly declared. Uninitialized scalar variables and array elements have both a numeric value of 0 (zero) and a string value of the null string (" ").

Variables take on numeric or string values according to context. Each variable can have a numeric value, a string value, or both. For example:

```
x = "4" + "8"
```

assigns the value of 12 to the variable x. For string constants, expressions should be enclosed in " " (double quotation) marks.

There are no explicit conversions between numbers and strings. To force an expression to be treated as a number, add 0 (zero) to it. To force an expression to be treated as a string, append a null string (" ").

Field Variables

Field variables are designated by a **$** (dollar sign) followed by a number or numerical expression. The first field in a record is assigned the **$1** variable , the second field is assigned to the **$2** variable, and so forth. The **$0** field variable is assigned to the entire record. New field variables can be created by assigning a value to them. Assigning a value to a non-existent field, that is, any field larger than the current value of **$NF** field variable, forces the creation of any intervening fields (set to the null string), increases the value of the **NF** special variable, and forces the value of **$0** record variable to be recalculated. The new fields are separated by the current field separator (which is the value of the **FS** special variable). Blanks and tabs are the default field separators. To change the field separator, use the **-F** flag, or assign the **FS** special variable a different value in the **awk** command program.

Arrays

Arrays are initially empty and their sizes change dynamically. Arrays are represented by a variable with subscripts in [ ] (square brackets). The subscripts, or element identifiers, can be numbers of strings, which provide a type of associative array capability. For example, the program:

```
/red/  { x["red"]++ }
/green/ { y["green"]++ }
```

increments counts for both the `red` counter and the `green` counter.

Arrays can be indexed with more than one subscript, similar to multidimensional arrays in some programming languages. Because programming arrays for the **awk** command are really one dimensional, the comma-separated subscripts are converted to a single string by concatenating the string values of the separate expressions, with each expression separated by the value of the **SUBSEP** environmental variable. Therefore, the following two index operations are equivalent:

```
x[expr1, expr2,...exprn]
```

AND

```
x[expr1SUBSEPexpr2SUBSEP...SUBSEPexprn]
```

When using the **in** operator, a multidimensional *Index* value should be contained within parentheses. Except for the **in** operator, any reference to a nonexistent array element automatically creates that element.

Special Variables

The following variables have special meaning for the **awk** command:

Item	Description
ARGC	The number of elements in the **ARGV** array. This value can be altered.
ARGV	The array with each member containing one of the *File* variables or *Assignment* variables, taken in order from the command line, and numbered from 0 (zero) to **ARGC** -1. As each input file is finished, the next member of the **ARGV** array provides the name of the next input file, unless: • The next member is an *Assignment* statement, in which case the assignment is evaluated. • The next member has a null value, in which case the member is skipped. Programs can skip selected input files by setting the member of the **ARGV** array that contains that input file to a null value. • The next member is the current value of **ARGV** [**ARGC** -1], which the **awk** command interprets as the end of the input files.
CONVFMT	The **printf** format for converting numbers to strings (except for output statements, where the **OFMT** special variable is used). The default is "%.6g".
ENVIRON	An array representing the environment under which the **awk** command operates. Each element of the array is of the form: **ENVIRON ["***Environment VariableName***"]** = *EnvironmentVariableValue* The values are set when the **awk** command begins execution, and that environment is used until the end of execution, regardless of any modification of the **ENVIRON** special variable.
FILENAME	The path name of the current input file. During the execution of a **BEGIN** action, the value of **FILENAME** is undefined. During the execution of an **END** action, the value is the name of the last input file processed.
FNR	The number of the current input record in the current file.

Item	Description
FS	The input field separator. The default value is a blank. If the input field separator is a blank, any number of locale-defined spaces can separate fields. The **FS** special variable can take two additional values: • With **FS** set to a single character, fields are separated by each single occurrence of the character. • With **FS** set to an extended regular expression, each occurrence of a sequence matching the extended regular expression separates fields.
NF	The number of fields in the current record, with a limit of 99. Inside a **BEGIN** action, the **NF** special variable is undefined unless a **getline** function without a *Variable* parameter has been issued previously. Inside an **END** action, the **NF** special variable retains the value it had for the last record read, unless a subsequent, redirected, **getline** function without a *Variable* parameter is issued prior to entering the **END** action.
NR	The number of the current input record. Inside a **BEGIN** action the value of the **NR** special variable is 0 (zero). Inside an **END** action, the value is the number of the last record processed.
OFMT	The **printf** format for converting numbers to strings in output statements. The default is "% .6g".
OFS	The output field separator (default is a space).
ORS	The output record separator (default is a new-line character).
RLENGTH	The length of the string matched by the **match** function.
RS	Input record separator (default is a new-line character). If the **RS** special variable is null, records are separated by sequences of one or more blank lines; leading or trailing blank lines do not result in empty records at the beginning or end of input; and the new-line character is always a field separator, regardless of the value of the **FS** special variable.
RSTART	The starting position of the string matched by the **match** function, numbering from 1. Equivalent to the return value of the **match** function.
SUBSEP	Separates multiple subscripts. The default is \031.

Flags

Item	Description
-f *ProgramFile*	Obtains instructions for the **awk** command from the file specified by the *ProgramFile* variable. If the **-f** flag is specified multiple times, the concatenation of the files, in the order specified, will be used as the set of instructions.
-u	Displays the output in an unbuffered mode. If this flag is used, the **awk** command does not buffer the output. Instead, it displays the output instantaneously. By default, the **awk** command displays the output in a buffered mode.
-F *Ere*	Uses the extended regular expression specified by the *Ere* variable as the field separator. The default field separator is a blank.

Item	Description
-v *Assignment*	Assigns a value to a variable for the **awk** command's programming language. The *Assignment* parameter is in the form of *Name = Value*. The *Name* portion specifies the name of the variable and can be any combination of underscores, digits, and alphabetic characters, but it must start with either an alphabetic character or an underscore. The *Value* portion is also composed of underscores, digits, and alphabetic characters, and is treated as if it were preceded and followed by a " (double-quotation character, similar to a string value). If the *Value* portion is numeric, the variable will also be assigned the numeric value. The assignment specified by the **-v** flag occurs before any portion of the **awk** command's program is executed, including the **BEGIN** section.
Assignment	Assigns a value to a variable for the **awk** command's programming language. It has the same form and function as the *Assignment* variable with the **-v** flag, except for the time each is processed. The *Assignment* parameter is processed just prior to the input file (specified by the *File* variable) that follows it on the command line. If the *Assignment* parameter is specified just prior to the first of multiple input files, the assignments are processed just after the **BEGIN** sections (if any). If an *Assignment* parameter occurs after the last file, the assignment is processed before the **END** sections (if any). If no input files are specified, the assignments are processed the standard input is read.
File	Specifies the name of the file that contains the input for processing. If no *File* variable is specified, or if a **-** (minus) sign is specified, standard input is processed.
'*Program***'**	Contains the instructions for the **awk** command. If the **-f** flag is not specified, the *Program* variable should be the first item on the command line. It should be bracketed by ' ' (single quotes).

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

You can alter the exit status within the program by using the **exit** [*Expression*] conditional statement.

Examples

1. To display the lines of a file that are longer than 72 characters, enter:

   ```
   awk  'length  >72'  chapter1
   ```

   This selects each line of the `chapter1` file that is longer than 72 characters and writes these lines to standard output, because no *Action* is specified. A tab character is counted as 1 byte.
2. To display all lines between the words `start` and `stop`, including `"start"` and `"stop"`, enter:

   ```
   awk  '/start/,/stop/'  chapter1
   ```
3. To run an **awk** command program, `sum2.awk`, that processes the file, `chapter1`, enter:

   ```
   awk  -f  sum2.awk  chapter1
   ```

The following program, `sum2.awk`, computes the sum and average of the numbers in the second column of the input file, `chapter1`:

```
    {
       sum += $2
    }
END {
       print "Sum: ", sum;
       print "Average:", sum/NR;
    }
```

The first action adds the value of the second field of each line to the variable `sum`. All variables are initialized to the numeric value of 0 (zero) when first referenced. The pattern **END** before the second action causes those actions to be performed after all of the input file has been read. The **NR** special variable, which is used to calculate the average, is a special variable specifying the number of records that have been read.

4. To print the first two fields in opposite order, enter:

```
awk '{ print $2, $1 }' chapter1
```

5. The following **awk** program

```
awk -f sum3.awk chapter2
```

prints the first two fields of the file `chapter2` with input fields separated by comma and/or blanks and tabs, and then adds up the first column, and prints the sum and average:

```
BEGIN  {FS = ",|[ \t]+"}
       {print $1, $2}
       {s += $1}
END    {print "sum is",s,"average is", s/NR }
```

b

The following AIX commands begin with the with the letter *b*.

back Command

Purpose

Starts the backgammon game.

Syntax

back

Description

The **back** command provides you with a partner for backgammon. You select one of the following three skill levels: beginner, intermediate, or expert. You can choose to roll your own dice during your turns, and you are asked if you want to move first.

Important locations on the computer-generated board are:

- 0 is the bar for removed white pieces.
- 1 is white's extreme inner table.
- 24 is brown's extreme inner table.
- 25 is the bar for removed brown pieces.

For details on how to make your moves, enter Y when prompted for `Instructions?` at the beginning of the game. During play, you are prompted for `move?`. Either enter a numerical move or press ? (question mark) key for a list of move choices.

When the game is finished, you are asked if you want to save game information. Entering Y stores game data in the **back.log** file in your current directory.

The **back** command plays only the forward game, even at the expert level. It objects if you try to make too many moves in a turn, but not if you make too few. Doubling is not permitted.

To quit the game, press the Interrupt (Ctrl-C) key sequence.

Files

Item	Description
/usr/games	Location of the system's games.
/usr/games/lib/backrules	Location of the rules file.
/tmp/b*	Location of the log temp file.
back.log	Contains data from previously played games.

backsnap Command

Purpose

Provides an interface to create a snapshot for a JFS2 file system and perform a backup of the snapshot.

Syntax

backsnap [**-R**] { **-m** *MountPoint* **-s size**=*Size* | **-n** *snapshotName* } [*BackupOptions*] *FileSystem*

Description

Provides an interface to create a snapshot for a JFS2 file system and perform a backup of the snapshot. The `restore` command can be used to retrieve the backup.

Flags

Item	Description
-m *MountPoint*	Specifies the path of where the external snapshot created should be mounted.
-R	Specifies that the snapshot created by this command will be removed when the backup completes.
-s size=*Size*	Specifies the size to create the new logical volume for the external snapshot. If *Size* is followed by an M, the value is treated as megabytes. If *Size* is followed by a G, the value is treated as gigabytes. Otherwise, the value is treated as 512-byte blocks.
-n *snapshotName*	Specifies the name of the internal snapshot to be created. The JFS2 file system must be enabled to use internal snapshots.

Parameters

Item	Description
BackupOptions	Any other options are passed to the **backup** command when the backup of the snapshot is performed. Minimally, it is required to specify the type of backup desired. For backup by name, the **-i** option must be specified along with the device for the backup. For **backup** by inode, the level option, -[0-9], must be specified along with the device for the backup. Use the `restore` command to retrieve the backup.
FileSystem	Specifies the JFS2 file system to create a snapshot of and backup.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Examples

1. To create a snapshot for the **/home/janet/sb** file system and perform a backup on the snapshot by name, enter:

   ```
   backsnap -m /tmp/snapshot/janetsb -s size=16M -i -f/dev/rmt0 /home/janet/sb
   ```

   This command creates a logical volume of size 16 megabytes and then creates a snapshot for the **/home/janet/sb** file system on the newly created logical volume. It then mounts the snapshot on **/tmp/snapshot/janetsb** and backs up the files and directories in that file system by name to the **/dev/rmt0** device.
2. To create a snapshot for the /home/janet/sb file system and perform a backup on the snapshot by inode, enter:

   ```
   backsnap -R -m /tmp/snapshot/janetsb -s size=16M -0 -f /dev/rmt0 /home/janet/sb
   ```

   This command creates a logical volume of size 16 megabytes and then creates a snapshot for the **/home/janet/sb** file system on the newly created logical volume. It then mounts the snapshot on **/tmp/snapshot/janetsb** and backs up the data in the snapshot by inode to the **/dev/rmt0** device. After the backup completes, the snapshot is deleted.

Files

Item	Description
/usr/sbin/backsnap	Contains the **backsnap** command.

backup Command

Purpose

Backs up files and file systems.

Syntax

To Back Up Files by Name

backup **-i** [**-b** *Number*] [**-p** [**-e** *RegularExpression*]] [**-E**{**force**|**ignore**|**warn**}] [**-f** *Device*] [**-l** *Number*] [-U] [**-O**] [**-o**] [**-q**] [**-v**] [**-Z**]

To Back Up File Systems by i-node

backup [[**-***Level*] [**-b** *Number*] [**-c**] [**-f** *Device*] [**-L** *Length*] [**-n** *snapshotName*] [-U] [**-O**] [**-u**]] [*FileSystem*] | [**-w** | **-W**] [**-Z**]

Description

The **backup** command creates copies of your files on a backup medium, such as a magnetic tape or diskette. The copies are in one of the two backup formats:

- Specific files backed up by name using the **-i** flag.
- Entire file system backed up by i-node using the *Level* and *FileSystem* parameters.

If you issue the **backup** command without any parameters, it defaults to a level 9 i-node backup of the root file system to the **/dev/rfd0** device. The default syntax is:

```
-9uf/dev/rfd0 /dev/rhd4
```

The default backup device is **/dev/rfd0**. If flags are specified that are not appropriate for the specified backup device, the **backup** command displays an error message and continues with the backup.

A single backup can span multiple volumes.

Notes:

1. Running the **backup** command results in the loss of all material previously stored on the selected output medium.
2. Data integrity of the archive may be compromised if a file is modified during system backup. Keep system activity at a minimum during the system backup procedure.
3. If a backup is made to a tape device with the device block size set to 0, it might be difficult to restore data from the tape unless the default write size was used with the **backup** command. The default write size for the **backup** command can be read by the **restore** command when the tape device block size is 0. In other words, the **-b** flag should not be specified when the tape device block size is 0. If the **-b** flag of the **backup** command is specified and is different from the default size, the same size must be specified with the **-b** flag of the **restore** command when the archived files are restored from the tape.
4. Do not attempt to back up a logical volume.

Backing Up Files by Name

To back up by name, use the **-i** flag. The **backup** command reads standard input for the names of the files to be backed up.

File types can be special files, regular files, or directories. When the file type is a directory, only the directory is backed up. The files under the directory are not backed up, unless they are explicitly specified.

Notes:

1. Files are restored using the same path names as the archived files. Therefore, to create a backup that can be restored from any path, use full path names for the files that you want to back up.
2. When backing up files that require multiple volumes, do not enter the list of file names from the keyboard. Instead, pipe or redirect the list from a file to the **backup** command. When you enter the file names from the keyboard and the backup process needs a new tape or diskette, the command "loses" any file names already entered but not yet backed up. To avoid this problem, enter each file name only after the archived message for the previous file has been displayed. The archived message consists of the character a followed by the file name.
3. If you specify the **-p** flag, only files of less than 2GB are packed.

Backing Up File Systems by i-node

To back up a file system by i-node, specify the **-***Level* and *FileSystem* parameters. When used in conjunction with the **-u** flag, the **-***Level* parameter provides a method of maintaining a hierarchy of incremental backups for each file system. Specify the **-u** flag and set the **-***Level* parameter to n to back up only those files that have been modified since the n-1 level backup. Information regarding the date, time, and level of each incremental backup is written to the **/etc/dumpdates** file. The possible backup levels are 0 to 9. A level 0 backup archives all files in the file system. If the **/etc/dumpdates** file contains no backup information for a particular file system, specifying any level causes all files in that file system to be archived.

The *FileSystem* parameter can specify either the physical device name (block or raw name) or the name of the directory on which the file system is mounted. The default file system is the root (**/**) file system.

Users must have read access to the file system device (such as **/dev/hd4**) or have Backup authorization in order to perform backups by i_node.

Notes:

1. You must first unmount a file system before backing it up by i-node. If you attempt to back up a mounted file system, a warning message is displayed. The **backup** command continues, but the created backup may contain inconsistencies because of changes that may have occurred in the file system during the backup operation.

2. Backing up file systems by i-node truncates the **uid** or **gid** of files having a **uid** or **gid** greater than 65535. When restored, these files may have different values for the **uid** and **gid** attributes. To retain the values correctly, always back up by name files having a **uid** or **gid** greater than 65535.
3. You can archive only JFS (Journaled File System) or JFS2 file systems when backing up by i-node. Back up any non-JFS or JFS2 file systems by file name or by using other archive commands, such as the **pax**, **tar**, or **cpio** command. In addition, backing up by i-node is not supported for file-systems located on disks that do not have a block-size of 512 bytes. These file-systems must be backed up using one of the other archive commands, such as the **pax**, **tar**, or **cpio** command.
4. The **-Z** flag is mandatory for backing up encrypted file systems.

Flags

Item	Description
-b *Number*	For backups by name, specifies the number of 512-byte blocks; for backups by i-node, specifies the number of 1024-byte blocks to write in a single output operation. When the **backup** command writes to tape devices, the default is 100 for backups by name and 32 for backups by i-node. The write size is the number of blocks multiplied by the block size. The default write size for the **backup** command writing to tape devices is 51200 (100 * 512) for backups by name and 32768 (32 * 1024) for backups by i-node. The write size must be an even multiple of the tape's physical block size. The value of the **-b** flag is always ignored when the **backup** command writes to diskette. In this case, the command always writes in clusters that occupy a complete track.
-c	Specifies that the tape is a cartridge, not a nine-track.
-e *RegularExpression*	Specifies that the files with names matching the regular expression are not to be packed. A regular expression is a set of characters, meta characters, and operators that define a string or group of strings in a search pattern. It can also be a string containing wildcard characters and operations that define a set of one or more possible strings. The **-e** flag is applied only when the **-p** flag is specified.
-E	For backups by name, the **-E** option requires one of the following arguments. If you omit the **-E** option, **warn** is the default behavior. **force** Fails the backup operation on a file if the fixed extent size or space reservation of the file cannot be preserved. **ignore** Ignores any errors in preserving extent attributes. **warn** Issues a warning if the space reservation or the fixed extent size of the file cannot be preserved.

Item	Description
-f *Device*	Specifies the output device. To send output to a named device, specify the *Device* variable as a path name (such as `/dev/rmt0`). To send output to the standard output device, specify a **-** (minus sign). The **-** (minus) feature enables you to pipe the output of the **backup** command to the **dd** command. You can also specify a range of archive devices. The range specification must be in the following format: `/dev/deviceXXX-YYY` where *XXX* and *YYY* are whole numbers, and *XXX* must always be less than *YYY*; for example, `/dev/rfd0-3`. All devices in the specified range must be of the same type. For example, you can use a set of 8mm, 2.3GB tapes or a set of 1.44MB diskettes. All tape devices must be set to the same physical tape block size. If the *Device* variable specifies a range, the **backup** command automatically goes from one device in the range to the next. After exhausting all of the specified devices, the **backup** command halts and requests that new volumes be mounted on the range of devices.
-i	Specifies that files be read from standard input and archived by file name. If relative path names are used, files are restored (with the **restore** command) relative to the current directory at restore time. If full path names are used, files are restored to those same names.
-l *Number*	(lowercase L) Limits the total number of blocks to use on the diskette device. The value specified must be a non-zero multiple of the number of sectors per diskette track. This option applies to by-name backups only. See the **format** command for information about sectors per diskette track.
-L *Length*	Specifies the length of the tape in bytes. This flag overrides the **-c**, **-d**, and **-s** flags. You can specify the size with a suffix of b, k, m, or g to represent Blocks (512 bytes), Kilo (1024 bytes), Mega (1024 Kilobytes), or Giga (1024 Megabytes), respectively. To represent a tape length of 2 Gigabytes, enter `-L 2g`. **Note:** Use the **-L** flag for i-node backups only.
-n *snapshotName*	Specifies the name of the internal snapshot to back up. You must mount the file system containing the snapshot. The **-n** flag is used for backups by the i-node only.
-o	Creates a Version 2-compatible backup by name. This flag is required for compatibility with Version 2 systems because backups by name that are created by a version higher than 2 cannot be restored on Version 2 systems. To create a Version 2-compatible backup by name, use the **-o** flag along with other flags required for backups by name. Files with attributes and values, such as user IDs and group IDs, that are too large for Version 2 systems will not be backed up. A message is displayed for each such file and each value that is too large.
-O	Creates a non-Trusted AIX security attributes backup. **Note:** The **-O** flag only applies for systems running Trusted AIX.

Item	Description
-p	Specifies that the files be packed, or compressed, before they are archived. Only files of less than 2GB are packed. **Note:** While using this option, it is recommended to keep the file system inactive. This option can be used on an active file system. However, if a file is modified at the time it is being backed up, there is an increased chance of the backup reporting a failure. You can omit this option while backing up to a tape device, which performs compression.
-q	Indicates that the removable medium is ready to use. When you specify the **-q** flag, the **backup** command proceeds without prompting you to prepare the backup medium and press the Enter key to continue. This option applies only to the first volume; you are prompted for subsequent volumes. The **-q** flag applies only to backups by name.
`-U`	Specifies to backup any ACLs or named extended attributes. Without this option the image will include only AIXC ACLs and PCLs in the archive along with the other regular file data. For files containing NFS4 ACLs, conversion to AIXC will happen by default during archival.
-u	Updates the **/etc/dumpdates** file with the raw device name of the file system and the time, date, and level of the backup. You must specify the **-u** flag if you are making incremental backups. The **-u** flag applies only to backups by i-node.
-v	Causes the **backup** command to display additional information about the backup. When using the **-v** flag, the size of the file as it exists on the archive is displayed in bytes. Additionally, a total of these file sizes is displayed when all files have been processed. Directories are listed with a size of 0. Symbolic links are listed with the size of the symbolic link. Hard links are listed with the size of the file, which is how hard links are archived. Block and character devices, if they were backed up, are listed with a size of 0. When the **-v** flag is not specified, the **backup** command displays only the names of the files being archived. This option is used only when backing up by file name.
-w	Currently disabled. If the **-w** flag is specified, no other flags are applied.
-W	Displays, for each file system in the **/etc/dumpdates** file, the most recent backup date and level. If the **-W** option is specified, no other flags are applied.
-*Level*	Specifies the backup level (0 to 9). The default level is 9.
-Z	Backs up the Encrypted File System (EFS) information for all of the files, directories, and file systems. The EFS information is extracted by default. **Note:** Archives created with **-Z** option can be restored only on AIX 6.1 or later releases.

Security

On Trusted AIX systems, only users with the **aix.fs.manage.backup** authorization can run the **backup** command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To backup all the files and subdirectories in the `/home` directory using full path names, enter:

   ```
   find /home -print | backup  -i  -f /dev/rmt0
   ```

   The **-i** flag specifies that files will be read from standard input and archived by file name. The **find** command generates a list of all the files in the `/home` directory. The files in this list are full path names. The | (pipe symbol) causes this list to be read from standard input by the **backup** command. The **-f** flag directs the **backup** command to write the files to the `/dev/rmt0` tape device. Because the files are archived using full path names, they will be written to the same paths when restored.

2. To backup all the files and subdirectories in the `/home/mike` directory using relative path names, enter:

   ```
   cd /home/mike
   find . -print | backup  -i  -v  -q
   ```

   Each file name in the list generated by the **find** command is preceded by **./** (dot, slash). Because the files are backed up using relative path names, they will be written to the current directory when restored. The **-v** flag causes the **backup** command to display additional information about the backup. The files are written to the default backup device **/dev/rfd0**.

3. To backup the **/** (root) file system, enter:

   ```
   backup  -0  -u  -f /dev/rmt0 /
   ```

   The `0` level specifies that all the files in the **/** (root) file system be backed up. The **-u** flag causes the **backup** command to update the **/etc/dumpdates** file for this backup.

4. To backup all the files in the **/** (root) file system that have been modified since the last level 0 backup, enter:

   ```
   backup  -1  -u  -f /dev/rmt0 /
   ```

   If the **/etc/dumpdates** file does not have an entry for a level 0 backup of the / (root) system, all the files in the file system are backed up.

5. To create an archive with Extended Attributes and ACLs, enter:

   ```
   ls /etc/passwd | backup -ivUf arch.bk
   ```

6. To create an archive without Trusted AIX security attributes, enter:

```
ls /etc/passwd | backup -ivOf arch.bk
```

Files

Item	Description
/etc/filesystems	Contains file system mount information.
/etc/dumpdates	Specifies log for incremental by i-node backups.
/dev/rfd0	Specifies default backup device.
/dev/rhd4	Specifies device where the default file system (root) is located.
/usr/sbin/backup	Contains the **backup** command.

banner Command

Purpose

Writes ASCII character strings in large letters to standard output.

Syntax

banner *String*

Description

The **banner** command writes ASCII character *String*s to standard output in large letters. Each line in the output can be up to 10 uppercase or lowercase characters in length. On output, all characters appear in uppercase, with the lowercase input characters appearing smaller than the uppercase input characters.

Each word you input appears on a separate line on the screen. When you want to display more than one word to a line, use quotation marks to specify which words will appear on one line.

Examples

1. To display a banner at the workstation, enter:

```
banner SMILE!
```

2. To display more than one word on a line, enclose the text in quotation marks, as follows:

```
banner "Out to" Lunch
```

This displays `Out to` on one line and `Lunch` on the next.

Files

Item	Description
/usr/bin/banner	Contains the **banner** command.

basename Command

Purpose

Returns the base file name of a string parameter.

Syntax

basename *String* [*Suffix*]

Description

The **basename** command reads the *String* parameter, deletes any prefix that ends with a / (slash) and any specified *Suffix* parameter, and writes the remaining base file name to standard output. The **basename** command applies the following rules in creating the base file name:

1. If the *String* parameter is a // (double slash), or if the *String* parameter consists entirely of slash characters, change the string to a single / (slash). Skip steps 2 through 4.
2. Remove any trailing / characters from the specified string.
3. If there are any / characters remaining in the *String* parameter, remove the prefix of the string up to and including the last / character.
4. If a *Suffix* parameter is specified and is identical to the characters remaining in the string, the string is not modified. For example, entering:

   ```
   K > basename /u/dee/desktop/cns.boo cns.boo
   ```

   results in:

   ```
   cns.boo
   ```

   If a *Suffix* parameter is specified and is not identical to all the characters in the string but is identical to a suffix in the string, the specified suffix is removed. For example, entering:

   ```
   K > basename /u/dee/desktop/cns.boo .boo
   ```

   results in:

   ```
   cns
   ```

   Failure to find the specified suffix within a string is not considered an error.

The **basename** and **dirname** commands are generally used inside command substitutions within a shell script to specify an output file name that is some variation of a specified input file name.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To display the base name of a shell variable, enter:

   ```
   basename $WORKFILE
   ```

   The command displays the base name of the value assigned to the shell variable `WORKFILE`. If the value of the `WORKFILE` variable is the **/home/jim/program.c** file, then the command displays `program.c`.
2. To construct a file name that is the same as another file name, except for its suffix, enter:

   ```
   OFILE=`basename $1 .c`.o
   ```

This command assigns to the OFILE file the value of the first positional parameter ($1), but with its `.c` suffix changed to `.o`. If $1 is the **/home/jim/program.c** file, OFILE becomes `program.o`. Because `program.o` is only a base file name, it identifies a file in the current directory.

Note: The ` (grave accent) specifies command substitution.

Files

Item	Description
/usr/bin/basename	Contains the **basename** command.

batch Command

Purpose

Runs jobs when the system load level permits.

Syntax

batch

Description

The **batch** command reads from standard input the names of commands to be run at a later time and runs the jobs when the system load level permits. The **batch** command mails you all output from standard output and standard error for the scheduled commands, unless you redirect that output. It also writes the job number and the scheduled time to standard error.

When the **batch** command is executed, it retains variables in the shell environment, and the current directory; however, it does not retain open file descriptors, traps, and priority.

The **batch** command is equivalent to entering the **at -q b -m now** command. The **-q b** flag specifies the **at** queue for batch jobs.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion
>0	An error occurred.

Examples

To run a job when the system load permits, enter:

```
batch <<!
longjob
!
```

This example shows the use of a "Here Document" to send standard input to the **batch** command.

Files

Item	Description
/usr/bin/batch	Contains the **batch** command.

Item	Description
/bin/batch	Symbolic link to the **batch** command.
/var/adm/cron	Indicates the main **cron** daemon directory.
/var/spool/cron/atjobs	Indicates the spool area.

battery Command

Purpose

Controls or queries battery information.

Syntax

battery [**-d**]

Description

The **battery** command controls or queries the battery. If the **battery** command is invoked without **-d** option, the following battery information is displayed:

```
battery type: NiCd or NiMH
current battery usage: charging, discharging, in use, fully charged
battery capacity
current remaining capacity
full charge count
```

If the **battery** command is invoked with **-d** option, the following battery information is also displayed:

```
discharge quantity
discharge time
```

If you use 50% of a battery's capacity and charge it every time (about 20 to 30 times), then the battery cannot be used at more than 50% of its capacity. This is called the *memory effect of battery*. If, then, the battery is discharged (made empty) and then recharged, the battery can be used at 100% again.

Flags

Item	Description
-d	Discharges the battery so you can reset the memory effect of battery.

Security

Access Control: Any User

Auditing Events: N/A

Examples

1. To show current battery status, enter:

   ```
   battery
   ```

   Something similar to the following displays:

   ```
   battery type: NiMH
   current battery usage: in use
   battery capacity: 3200 (mAH)
   ```

```
current remaining capacity: 1800 (mAH) [57%]
full charge count: 3
```

Files

Item	Description
/usr/bin/battery	Contains the **battery** command.

bc Command

Purpose

Provides an interpreter for arbitrary-precision arithmetic language.

Syntax

bc [**-c**] [**-l**] [*File* ...]

Description

The **bc** command is an interactive process that provides arbitrary-precision arithmetic. The **bc** command first reads any input files specified by the *File* parameter and then reads the standard input. The input files must be text files containing a sequence of commands, statements, or function definitions that the **bc** command can read and execute.

The **bc** command is a preprocessor for the **dc** command. It calls the **dc** command automatically, unless the **-c** (compile only) flag is specified. If the **-c** flag is specified, the output from the **bc** command goes to standard output.

The **bc** command allows you to specify an input and output base for operations in decimal, octal, or hexadecimal. The default is decimal. The command also has a scaling provision for decimal point notation. The **bc** command always uses the . (period) to represent the radix point, regardless of any decimal point character specified as part of the current locale.

The syntax for the **bc** command is similar to that of the C language. You can use the **bc** command to translate between bases by assigning the **ibase** keyword to the input base and the **obase** keyword to the output base. A range of 2-16 is valid for the **ibase** keyword. The **obase** keyword ranges from 2 up to the limit set by the **BC_BASE_MAX** value defined in the **/usr/include/sys/limits.h** file. Regardless of the **ibase** and **obase** settings, the **bc** command recognizes the letters A-F as their hexadecimal values 10-15.

The output of the **bc** command is controlled by the program read. Output consists of one or more lines containing the value of all executed expressions without assignments. The radix and precision of the output are controlled by the values of the **obase** and **scale** keywords.

Further information about the way in which the **bc** command processes information from a source file is described in the following sections:

- Grammar
- Lexical Conventions
- Identifiers and Operators
- Expressions
- Statements
- Function Calls
- Functions in -I Math Library

Grammar

The following grammar describes the syntax for the **bc** program, where `program` stands for any valid program:

```
%token  EOF NEWLINE STRING LETTER NUMBER

%token  MUL_OP
/*      '*', '/', '%'                           */

%token  ASSIGN_OP
/*      '=', '+=', '-=', '*=', '/=', '%=', '^='  */

%token  REL_OP
/*      '==', '<=', '>=', '!=', '<', '>'          */

%token  INCR_DECR
/*      '++', '--'                                */

%token  Define    Break    Quit    Length
/*      'define', 'break', 'quit', 'length'       */

%token  Return    For    If    While    Sqrt
/*      'return', 'for', 'if', 'while', 'sqrt'    */

%token  Scale    Ibase    Obase    Auto
/*      'scale', 'ibase', 'obase', 'auto'         */

%start   program

%%

program           : EOF
                  | input_item program
                  ;

input_item        : semicolon_list NEWLINE
                  | function
                  ;

semicolon_list    : /* empty */
                  | statement
                  | semicolon_list ';' statement
                  | semicolon_list ';'
                  ;

statement_list    : /* empty */
                  | statement
                  | statement_list NEWLINE
                  | statement_list NEWLINE statement
                  | statement_list ';'
                  | statement_list ';' statement
                  ;
```

```
statement         : expression
                  | STRING
                  | Break
                  | Quit
                  | Return
                  | Return '(' return_expression ')'
                  | For '(' expression ';'
                       relational_expression ';'
                       expression ')' statement
                  | If '(' relational_expression ')' statement
                  | While '(' relational_expression ')' statement
                  | '{' statement_list '}'
                  ;

function          : Define LETTER '(' opt_parameter_list ')'
                       '{' NEWLINE opt_auto_define_list
                       statement_list '}'
                  ;

opt_parameter_list:/* empty */
                  | parameter_list
                  ;

parameter_list    : LETTER
                  | define_list ',' LETTER
                  ;

opt_auto_define_list
                  : /* empty */
                  | Auto define_list NEWLINE
                  | Auto define_list ';'
                  ;

define_list       : LETTER
                  | LETTER '[' ']'
                  | define_list ',' LETTER
                  | define_list ',' LETTER '[' ']'
                  ;

opt_argument_list : /* empty */
                  | argument_list
                  ;

argument_list     : expression
                  | argument_list ',' expression
                  ;

relational_expression
                  : expression
                  | expression REL_OP expression
                  ;

return_expression : /* empty */
                  | expression
                  ;

expression        : named_expression
                  | NUMBER
                  | '(' expression ')'
                  | LETTER '(' opt_argument_list ')'
                  | '-' expression
                  | expression '+' expression
```

```
                  | expression '-' expression
                  | expression MUL_OP expression
                  | expression '^' expression
                  | INCR_DECR named_expression
                  | named_expression INCR_DECR
                  | named_expression ASSIGN_OP expression
                  | Length '(' expression ')'
                  | Sqrt '(' expression ')'
                  | Scale '(' expression ')'
                  ;
```

```
named_expression  : LETTER
                  | LETTER '[' expression ']'
                  | Scale
                  | Ibase
                  | Obase
                  ;
```

Lexical Conventions

The following lexical conventions apply to the **bc** command:

1. The **bc** command recognizes the longest possible lexical token or delimiter beginning at a given point.
2. Comments begin with /* (slash, asterisk) and end with */ (asterisk, slash). Comments have no effect except to delimit lexical tokens.
3. The newline character is recognized as the **NEWLINE** token.
4. The **STRING** token represents a string constant. The string begins with " (double quotation mark) and terminates with " (double quotation mark). All characters between the quotation marks are taken literally. There is no way to specify a string that contains " (double quotation mark). The length of each string is limited to the maximum bytes set in the **BC_STRING_MAX** value, which is defined in the **limits.h** file.
5. Blank characters have no effect except as they appear in the **STRING** token or when used to delimit lexical tokens.
6. The \n (backslash, newline) character:
   - delimits lexical tokens.
   - is interpreted as a character sequence in **STRING** tokens.
   - is ignored when part of a multiline **NUMBER** token.
7. A **NUMBER** token uses the following grammar:

   ```
   NUMBER  : integer
           | '.' integer
           | integer '.'
           |integer '.' integer
           ;
   integer : digit
           | integer digit
           ;
   digit   : 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
           | 8 | 9 | A | B | C | D | E | F
           ;
   ```

   NUMBER token values are interpreted as numerals in the base specified by the **ibase** internal register value.
8. The value of a **NUMBER** token is interpreted as a numeral in the base specified by the value of the **ibase** internal register. Each of the digit characters has the value from 0 to 15 in the order listed here, and the period character presents the radix point. The behavior is undefined if digits greater than or equal to the value of the **ibase** register appear in the token. There is an exception for single-digit values being assigned to the **ibase** and **obase** registers themselves.
9. The following keywords are recognized as tokens:

```
auto    for     length   return sqrt
break   ibase   obase    scale  while
define  if      quit
```

10. Except within a keyword, any of the following letters are considered a **LETTER** token:

```
a b c d e f g h i j k l m n o p q r s t u v w x y z
```

11. The following single-character and two-character sequences are recognized as the **ASSIGN_OP** token:
    - = (equal sign)
    - += (plus, equal sign)
    - -= (minus, equal sign)
    - *= (asterisk, equal sign)
    - /= (slash, equal sign)
    - %= (percent, equal sign)
    - ^= (caret, equal sign)
12. The following single characters are recognized as the **MUL_OP** token:
    - * (asterisk)
    - / (slash)
    - % (percent)
13. The following single-character and two-character sequences are recognized as the **REL_OP** token:
    - == (double equal sign)
    - <= (less than, equal sign)
    - >= (greater than, equal sign)
    - != (exclamation point, equal sign)
    - < (less than)
    - > (greater than)
14. The following two-character sequences are recognized as the **INCR_DECR** token:
    - ++ (double plus sign)
    - -- (double hyphen)
15. The following single characters are recognized as tokens. The token has the same name as the character:

    <newline>

    ((left parenthesis)

) (right parenthesis)

    , (comma)

    + (plus)

    - (minus)

    ; (semicolon)

    [(left bracket)

    ] (right bracket)

    ^ (caret)

    { (left brace)

    } (right brace)
16. The **EOF** token is returned when the end of input is reached.

Identifiers and Operators

There are three kinds of identifiers recognized by the **bc** command: ordinary identifiers, array identifiers, and function identifiers. All three types consist of single, lowercase letters. Array identifiers are followed by [ ] (left and right brackets). An array subscript is required except in an argument or auto list. Arrays are singly dimensioned and can contain up to the amount specified by the **BC_DIM_MAX** value. Indexing begins at 0. Therefore an array is indexed from 0 up to the value defined by **BC_DIM_MAX -1**. Subscripts are truncated to integers. Function identifiers must be followed by ( ) (left and right parentheses) and possibly by enclosing arguments. The three types of identifiers do not conflict.

The Operators in a bc Program table summarizes the rules for precedence and associativity of all operators. Operators on the same line have the same precedence. Rows are in order of decreasing precedence.

Operator	Associativity
++, - -	not applicable
unary -	not applicable
^	right to left
*, /, %	left to right
+, binary -	left to right
=, +=, -=, *=, /=, ^=	right to left
==, <=, >=, !=, <, >	none

Each expression or named expression has a *scale,* which is the number of decimal digits maintained as the fractional portion of the expression.

Named expressions are places where values are stored. Named expressions are valid on the left side of an assignment. The value of a named expression is the value stored in the place named. Simple identifiers and array elements are named expressions; they have an initial value of zero and an initial scale of zero.

The internal registers **scale**, **ibase**, and **obase** are all named expressions. The scale of an expression consisting of the name of one of these registers is 0. Values assigned to any of these registers are truncated to integers. The **scale** register contains a global value used in computing the scale of expressions (as described below). The value of the **scale** register is limited to 0 <= **scale** <= {**BC_SCALE_MAX**} and has a default value of 0. The **ibase** and **obase** registers are the input and output number radix, respectively. The value of **ibase** is limited to 2 <= **ibase** <= 16. The value of **obase** is limited to 2 <= **obase** = {**BC_BASE_MAX**}

When either the **ibase** or **obase** registers are assigned a single-digit value from the list described in "Lexical Conventions" , the value is assumed in hexadecimal. For example:

```
ibase=A
```

sets to base ten, regardless of the current **ibase** register value. Otherwise, the behavior is undefined when digits greater than or equal to the value of the **ibase** register appear in the input. Both **ibase** and **obase** registers have initial values of 10.

Internal computations are conducted as if in decimal, regardless of the input and output bases, to the specified number of decimal digits. When an exact result is not achieved, for example:

```
scale=0; 3.2/1
```

the **bc** command truncates the result.

All numerical values of the **obase** register are output according to the following rules:

1. If the value is less than 0, output a - (hyphen).
2. Output one of the following, depending on the numerical value:

- If the absolute value of the numerical value is greater than or equal to 1, output the integer portion of the value as a series of digits appropriate to the **obase** register (described in step 3). Next output the most significant non-zero digit, followed by each successively less significant digit.
- If the absolute value of the numerical value is less than 1 but greater than 0 and the scale of the numerical value is greater than 0, it is unspecified whether the character 0 is output.
- If the numerical value is 0, output the character 0.

3. If the scale of the value is greater than 0, output a . (period) followed by a series of digits appropriate to the following **obase** register values. The digits represent the most significant portion of the fractional part of the value, and *s* represents the scale of the value being output:
   - If the **obase** value is 10, output *s* number of digits.
   - If the **obase** value is greater than 10, output the number less than or equal to *s*.
   - If the **obase** value is less than 10, output a number greater than or equal to *s*.
   - For **obase** values other than 10, this should be the number of digits needed to represent a precision of 10s.
   - For **obase** values from 2 to 16, valid digits are the first **obase** of the single characters:

     ```
     0   1   2   3   4   5   6   7   8   9   A   B   C   D   E   F
     ```

     which represent the values 0 through 15, respectively.
   - For bases greater than 16, each digit is written as a separate multidigit decimal number. Each digit except the most significant fractional digit is preceded by a single space character. For bases 17 to 100, the **bc** command writes two-digit decimal numbers, for bases 101 to 1000 the **bc** command writes three-digit decimal numbers. For example, the decimal number 1024 in base 25 would be written as:

     ```
     01 15 24
     ```

     in base 125, as:

     ```
     008 024
     ```

   Very large numbers are split across lines, with 70 characters per line in the POSIX locale. Other locales may split at different character boundaries. Lines that are continued must end with a \ (backslash).

Expressions

A numeric constant is an expression. The scale is the number of digits that follow the radix point in the input representing the constant, or 0 if no radix point appears.

The sequence (*expression*) is an expression with the same value and scale as *expression*. The parentheses can be used to alter the normal precedence.

The unary and binary operators have the following semantics:

Item	Description
-expression	The result is the negative of the expression. The scale of the result is the scale of the expression.
	The unary increment and decrement operators do not modify the scale of the named expression upon which they operate. The scale of the result is the scale of that named expression.
++named_expression	The named expression is incremented by 1. The result is the value of the named expression after incrementing.
- -named_expression	The named expression is decremented by 1. The result is the value of the named expression after decrementing.

Item	Description
named_expression++	The named expression is incremented by 1. The result is the value of the named expression before incrementing.
named_expression- -	The named expression is decremented by 1. The result is the value of the named expression before decrementing.

The exponentiation operator, ^ (caret), binds right to left.

Item	Description
expression ^*expression*	The result is the first *expression* raised to the power of the second *expression.* If the second expression is not an integer, the behavior is undefined. If a is the scale of the left expression and b is the absolute value of the right expression, the scale of the result is: `if b >= 0 min(a * b, max(scale, a))` `if b <     0 scale`

The multiplicative operators * (asterisk), / (slash), and % (percent) bind left to right.

Item	Description
expression * *expression*	The result is the product of the two expressions. If a and b are the scales of the two expressions, then the scale of the result is: `min(a+b,max(scale,a,b))`
expression / *expression*	The result is the quotient of the two expressions. The scale of the result is the value of **scale**.
expression % *expression*	For expressions a and b, a % b is evaluated equivalent to the following steps: 1. Compute a/b to current scale. 2. Use the result to compute: `a - (a / b) * b` to scale: `max(scale + scale(b), scale(a))` The scale of the result will be: `max(scale + scale(b), scale(a))` When **scale** is zero, the % operator is the mathematical remainder operator.

The additive operators + (plus) and - (minus) bind left to right.

Item	Description
expression + *expression*	The result is the sum of the two expressions. The scale of the result is the maximum of the scales of the expressions.
expression - *expression*	The result is the difference of the two expressions. The scale of the result is the maximum of the scales of the expressions.

The following assignment operators bind right to left:

- = (equal sign)
- += (plus, equal sign)
- -= (minus, equal sign)
- *= (asterisk, equal sign)
- /= (slash, equal sign)
- %= (percent, equal sign)
- ^= (caret, equal sign)

Item	Description
named-expression = *expression*	This expression results in assigning the value of the expression on the right to the named expression on the left. The scale of both the named expression and the result is the scale of the expression.

The compound assignment forms:

named-expression <*operator* >= *expression*

are equivalent to:

named-expression = *named-expression* <*operator* > *expression*

except that the named expression is evaluated only once.

Unlike all other operators, the following relational operators are only valid as the object of an **if** or **while** statement or inside a **for** statement:

- < (less than)
- > (greater than)
- <= (less than, equal sign)
- >= (greater than, equal sign)
- == (double equal sign)
- != (exclamation, equal sign)

Item	Description
expression1 < *expression2*	The relation is true if the value of *expression1* is strictly less than the value of *expression2.*
expression1 > *expression2*	The relation is true if the value of *expression1* is strictly greater than the value of *expression2.*
expression1 <= *expression2*	The relation is true if the value of *expression1* is less than or equal to the value of *expression2.*
expression1 >= *expression2*	The relation is true if the value of *expression1* is greater than or equal to the value of *expression2.*
expression1 == *expression2*	The relation is true if the values of *expression1* and *expression2* are equal.
expression1 != *expression2*	The relation is true if the values of *expression1* and *expression2* are unequal.

Statements

When a statement is an expression, unless the main operator is an assignment, execution of the statement writes the value of the expression followed by a newline character.

When a statement is a string, execution of the statement writes the value of the string.

Statements separated by semicolons or newline characters are executed sequentially. In an interactive invocation of the **bc** command, each time a newline character is read that satisfies the grammatical production:

```
input_item : semicolon_list NEWLINE
```

the sequential list of statements making up the **semicolon_list** is executed immediately, and any output produced by that execution is written without any buffer delay.

If an **if** statement (**if** (*relation*) *statement*), the *statement* is executed if the relation is true.

The **while** statement (**while** (*relation*) *statement*) implements a loop in which the *relation* is tested. Each time the *relation* is true, the *statement* is executed and the *relation* retested. When the *relation* is false, execution resumes after *statement.*

A **for** statement (**for** (*expression; relation; expression*) *statement*) is the same as:

```
first-expression
while   (relation) {
   statement
   last-expression
}
```

All three expressions must be present.

The **break** statement causes termination for a **for** or **while** statement.

The **auto** statement (**auto** *identifier* **[**,*identifier* **]** ...) causes the values of the identifiers to be pushed down. The identifiers can be ordinary identifiers or array identifiers. Array identifiers are specified by following the array name by empty square brackets. The **auto** statement must be the first statement in a function definition.

The **define** statement:

```
define   LETTER ( opt_parameter_list )  {
   opt_auto_define_list
   statement_list
}
```

defines a function named LETTER. If the LETTER function was previously defined, the **define** statement replaces the previous definition. The expression:

```
LETTER ( opt_argument_list )
```

invokes the LETTER function. The behavior is undefined if the number of arguments in the invocation does not match the number of parameters in the definition. Functions are defined before they are invoked. A function is considered defined within its own body, so recursive calls are valid. The values of numeric constants within a function are interpreted in the base specified by the value of the **ibase** register when the function is invoked.

The **return** statements (**return** and **return**(*expression*)) cause termination of a function, popping of its **auto** variables, and specify the result of the function. The first form is equivalent to return(0). The value and scale of an invocation of the function is the value and scale of the expression in parentheses.

The **quit** statement (**quit**) stops execution of a **bc** program at the point where the statement occurs in the input, even if it occurs in a function definition or in an **if**, **for**, or **while** statement.

Function Calls

A function call consists of a function name followed by parentheses containing a comma-separated list of expressions, which are the function arguments. A whole array passed as an argument is specified by the array name followed by [ ] (left and right brackets). All function arguments are passed by value. As a result, changes made to the formal parameters have no effect on the actual arguments. If the function

terminates by executing a **return** statement, the value of the function is the value of the expression in the parentheses of the **return** statement, or 0 if no expression is provided or if there is no **return** statement.

The result of **sqrt**(*expression*) is the square root of the expression. The result is truncated in the least significant decimal place. The scale of the result is the scale of the expression or the value of **scale**, whichever is larger.

The result of **length**(*expression*) is the total number of significant decimal digits in the expression. The scale of the result is 0.

The result of **scale**(*expression*) is the scale of the expression. The scale of the result is 0.

There are only two storage classes in a **bc** program, global and automatic (local). Only identifiers that are to be local to a function need be declared with the **auto** keyword. The arguments to a function are local to the function. All other identifiers are assumed to be global and available to all functions. All identifiers, global and local, have initial values of 0. Identifiers declared as **auto** are allocated on entry to the function and released on returning from the function. Therefore they do not retain values between function calls. The **auto** arrays are specified by the array name followed by [ ] (left bracket, right bracket). On entry to a function, the old values of the names that appear as parameters and as automatic variables are pushed onto a stack. Until the function returns, reference to these names refers only to the new values.

References to any of these names from other functions that are called from this function also refer to the new value until one of those functions uses the same name for a local variable.

Functions in -l Math Library

The following functions are defined when you specify the **-l** flag:

Item	Description
s(*expression***)**	Specifies the sine of *expressionx*, where *expression* is in radians.
c(*expression***)**	Specifies the cosine of *expressionx*, where *expression* is in radians.
a(*expression***)**	Specifies the arctangent of *expressionx*, where *expression* is in radians.
l(*expression***)**	Specifies the natural logarithm of *expression*.
e(*expression***)**	Specifies the exponential of *expression*.
j(*expression***,***expression***)**	Specifies the Bessel function of integer order.

The scale of an invocation of each of these functions is the value of the **scale** keyword when the function is invoked. The behavior is undefined if any of these functions is invoked with an argument outside the domain of the mathematical function.

Flags

Item	Description
-c	Compiles the *File* parameter, but does not invoke the **dc** command.
-l	(Lowercase L) Defines a library of math functions, and sets the **scale** variable to 20.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.

Item	Description
1	Encountered a syntax error or could not access the input file.
unspecified	Any other error occurred.

Examples

1. You can use the **bc** command as a calculator. Depending on whether you set the **scale** variable and with what value, the system displays fractional amounts. Entering:

   ```
   bc
   1/4
   ```

   displays only `0`. To set the **scale** variable and add a comment, enter:

   ```
   scale = 1 /* Keep 1 decimal place */
   1/4
   ```

   The screen displays `0.2`. Entering:

   ```
   scale = 3 /* Keep 3 decimal places */
   1/4
   ```

   displays `0.250`. Entering:

   ```
   16+63/5
   ```

   displays `28.600`. Entering

   ```
   (16+63)/5
   ```

   displays `15.800`. Entering

   ```
   71/6
   ```

   displays `11.833`.

   The **bc** command displays the value of each expression when you press the Enter key, except for assignments.

   When you enter the **bc** command expressions directly from the keyboard, press the End-of-File (Ctrl-D) key sequence to end the **bc** command session and return to the shell command line.

2. To write and run a C-like program, enter a command similar to the following:

   ```
   bc  -l prog.bc
   e(2) /* e squared */
   ma
   ```

   The screen displays `7.38905609893065022723`. If you enter:

   ```
   f(5)   /* 5 factorial */
   ```

   The screen displays `120`. If you enter:

   ```
   f(10)   /* 10 factorial */
   ```

   The screen displays `3628800`.

   This sequence interprets the **bc** program saved in the **prog.bc** file, and reads more of the **bc** command statements from the keyboard. Starting the **bc** command with the **-l** flag makes the math library available. This example uses the **e** (exponential) function from the math library, and `f` is defined in the **prog.bc** program file as:

```
/* compute the factorial of n */
define f(n) {
 auto i, r;

 r = 1;
 for (i=2; i<=n; i++) r =* i;
 return (r);
}
```

The statement following a **for** or **while** statement must begin on the same line. When you enter the **bc** command expressions directly from the keyboard, press the End-of-File (Ctrl-D) key sequence to end the **bc** command session and return to the shell command line.

3. To convert an infix expression to Reverse Polish Notation (RPN), enter:

```
bc  -c
(a * b) % (3 + 4 * c)
```

The screen displays:

```
lalb* 3 4lc*+%ps.
```

This sequence compiles the **bc** command infix-notation expression into an expression that the **dc** command can interpret. The **dc** command evaluates extended RPN expressions. In the compiled output, the `l` before each variable name is the **dc** subcommand to load the value of the variable onto the stack. The `p` displays the value on top of the stack, and the `s.` discards the top value by storing it in register `.` (dot). You can save the RPN expression in a file for the **dc** command to evaluate later by redirecting the standard output of this command. When you enter the **bc** command expressions directly from the keyboard, press the End-of-File (Ctrl-D) key sequence to end the **bc** command session and return to the shell command line.

4. To assign in the shell an approximation of the first 10 digits of pi to the variable *x*, enter:

```
x=$(printf "%s\n" 'scale = 10; 104348/33215' | bc)
```

The following **bc** program prints the same approximation of pi, with a label, to standard output:

```
scale = 10
"pi equals "
104348 / 33215
```

5. To define a function to compute an approximate value of the exponential function (such a function is predefined if the -l (lowercase L) option is specified), enter:

```
scale = 20
define  e(x){
        auto a, b, c, i, s
        a = 1
        b = 1
        s = 1
        for (i = 1; 1 == 1; i++){
                a = a*x
                b = b*i
                c = a/b
                if (c == 0) {
                        return(s)
                }
                s = s+c
        }
}
```

To print approximate values of the exponential function of the first 10 integers, enter:

```
for (i = 1; i <= 10; ++i) {
        e(i)
}
```

Files

Item	Description
/usr/bin/bc	Contains the **bc** command.
/usr/lib/lib.b	Contains the mathematical library.
/usr/bin/dc	Contains the desk calculator.

bdftopcf Command

Purpose

Converts fonts from Bitmap Distribution Format (bdf) to Portable Compiled Format (pcf).

Syntax

bdftopcf [**-i** | **-t**] [**-p** *Number*] [**-u** *Number*] [**-l** | **-m**] [**-L** | **-M**] [**-o** *PcfFile*] **font-file.bdf**

Description

The **bdftopcf** command is the font compiler which converts fonts from Bitmap Distribution Format to Portable Compiled Format. Fonts in Portable Compiled Format can be read by any architecture, although the file is structured to allow one particular architecture to read them directly without reformatting. This feature allows fast reading on the appropriate machine. In addition, the files remain portable to other machines, although they are read more slowly.

Flags

Item	Description
-p *Number*	Sets the font glyph padding. Each glyph in the font has each scanline padded into a multiple of bytes specified by the *Number* variable, where *Number* is the value of 1, 2, 4, or 8 bytes.
-u *Number*	Sets the font scanline unit. When the font bit order is different from the font byte order, the *Number* variable describes what units of data (in bytes) are to be swapped. The *Number* variable can be the value of 1, 2, or 4 bytes.
-m	Sets the font bit order to MSB (most significant bit) first. Bits for each glyph are placed in this order. Thus, the left-most bit on the screen is the highest valued bit in each unit.
-l	(lowercase L) Sets the font bit order to LSB (least significant bit) first. The left-most bit on the screen is the lowest valued bit in each unit.
-M	Sets the font byte order to MSB (most significant byte) first. All multibyte data in the file, including metrics and bitmaps, are written most significant byte first.
-L	Sets the font byte order to LSB (least significant byte) first. All multibyte data in the file, including metrics and bitmaps, are written least significant byte first.
-t	Converts fonts into *terminal* fonts whenever possible. A terminal font has each glyph image padded to the same size. The Xserver can usually render these font types more quickly.

Item	Description
-i	Inhibits the normal computation of ink metrics. When a font has glyph images that do not fill the bitmap image because the ``on'' pixels do not extend to the edges of the metrics, the **bdftopcf** command computes the actual ink metrics and places them in the **.pcf** file. **Note:** The **-t** option inhibits the behavior of this flag.
-o *PcfFile*	Specifies the name of an output file. By default, the **bdftopcf** command writes the **pcf** file to standard output.

Examples

1. To convert fonts into terminal fonts whenever possible, enter:

```
bdftocpf -t font-file.bdf
```

2. To set the glyph padding to a multiple of 4 bytes, enter:

```
bdftocpf -p 4 font-file.bdf
```

bdiff Command

Purpose

Uses the **diff** command to find differences in very large files.

Syntax

bdiff { *File1* | **-** } { *File2* | **-** } [*Number*] [**-s**]

Description

The **bdiff** command compares the files specified by the *File1* and *File2* parameters and writes information about their differing lines to standard output. If either file name is **-** (minus), the **bdiff** command reads standard input. The **bdiff** command is used like the **diff** command to find lines that must be changed in two files to make them identical. The primary purpose of this command is to permit processing of files that are too large for the **diff** command.

The **bdiff** command ignores lines common to the beginning of both files, splits the remainder of each file into segments of *Number* lines each, and calls the **diff** command to compare the corresponding segments. In some cases, the 3500 line default for the *Number* parameter is too large for the **diff** command. If the **diff** command fails, specify a smaller value for the *Number* parameter and try again.

The output of the **bdiff** command has the same format as that of the **diff** command. The **bdiff** command adjusts line numbers to account for the segmenting of the files. Note that because of the file segmenting, the **bdiff** command does not necessarily find the smallest possible set of file differences.

Flags

Item	Description
-s	Suppresses error messages from the **bdiff** command. (Note that the **-s** flag does not suppress error messages from the **diff** command).

Examples

To display the differences between the `chap1` file and the `chap1.bak` file:

```
bdiff chap1 chap1.bak
```

Files

Item	Description
/usr/bin/bdiff	Contains the **bdiff** command.

bellmail Command

Purpose

Sends messages to system users and displays messages from system users.

Syntax

To Display Messages

bellmail [**-e**] [**-f***File*] [**-p**] [**-q**] [**-r**]

To Send Messages

bellmail [**-t**] *User* ...

Description

The **bellmail** command with no flags writes to standard output, one message at a time, all stored mail addressed to your login name. Following each message, the **bellmail** command prompts you with a ? (question mark). Press the Enter key to display the next mail message, or enter one of the **bellmail** subcommands to control the disposition of the message.

Use the *User* parameter to attach a prefix to messages you send. The **bellmail** command prefaces each message with the sender's name, date and time of the message (its postmark), and adds the message to the user's mailbox. Specify the *User* parameter by pressing End Of File (the Ctrl-D key sequence) or entering a line containing only a . (period) after your message.

The action of the **bellmail** command can be modified by manipulating the **/var/spool/mail/***UserID* mailbox file in two ways:

- The default permission assignment for *others* is `all permissions denied (660)`. You may change this permission to `read/write`. When you change permissions from the default, the system preserves the file, even when it is empty, to maintain the desired permissions. You can no longer remove the file.
- You can edit the file to contain as its first line:

  ```
  Forward to person
  ```

  This instruction causes all messages sent to the *User* parameter to be sent to the *Person* parameter instead. The `Forward to` feature is useful for sending all of a person's mail to a particular machine in a network environment.

To specify a recipient on a remote system accessible through UNIX-to-UNIX Copy Program (UUCP), preface the *User* parameter with the system name and an ! (exclamation mark). The [**-t**] *User*. . .**uucp** command contains additional information about addressing remote systems.

> **Note:** In order to use the remote mail function, UUCP must be completely configured.

If you are interested in writing your own third-party mail program, you may need to know the following locking mechanisms used by the **bellmail** command.

1. The **bellmail** command creates a *UserID***.lock** file in the **/var/spool/mail** directory that is opened by passing the **O_NSHARE** and **O_DELAY** flags to the **open** subroutine. If the *UserID***.lock** file is being held, your **bellmail** process sleeps until the lock is free.
2. The **bellmail** command locks **/var/spool/mail/***UserID* with the **lockf** subroutine.

Flags

Item	Description
-e	Does not display any messages. This flag causes the **bellmail** command to return an exit value of 0 if the user has mail, or an exit value of 1 if there is no mail.
-f*File*	Reads mail from the named *File* parameter instead of the default mail file, **/var/spool/mail/***UserID*.
-p	Displays mail without prompting for a disposition code. This flag does not delete, copy, or forward any messages.
-q	Causes the **bellmail** command to exit when you press Interrupt (the Ctrl-C key sequence). Pressing Interrupt (Ctrl-C) alone stops only the message being displayed. (In this case, the next message sometimes is not displayed until you enter the **p** subcommand.)
-r	Displays mail in first-in, first-out order.
-t	Prefaces each message with the names of all recipients of the mail. (Without this flag, only the individual recipient's name displays as addressee.)

The *User* parameter is a name normally recognized by the **login** command. If the system does not recognize one or more of the specified *User* parameters or if the **bellmail** command is interrupted during input, the **bellmail** command tries to save the message in the **dead.letter** file in the current directory. If the **bellmail** command cannot save the message to the **dead.letter** file, it saves the message in the **$HOME/dead.letter** file. Once in this file, the message can be edited and sent again.

Note: The **bellmail** command uses the **$MAIL** environment variable to find the user's mailbox.

Subcommands

The following subcommands control message disposition:

Item	Description
+	Displays the next mail message (the same as pressing the Enter key).
-	Displays the previous message.
!*Command*	Runs the specified workstation command.
*****	Displays a subcommand summary.
d	Deletes the current message and displays the next message.
m *User*	Forwards the message to the specified *User* parameter.
p	Displays the current message again.
q	Writes any mail not yet deleted to the **/var/spool/mail/***UserID* file and exits. Pressing End Of File (Ctrl-D) has the same effect.
s [*File*]	Saves the message in the named *File* parameter instead of in the default mail file, **$HOME/mbox**.
w [*File*]	Saves the message, without its postmark, in the specified *File* parameter instead of in the default mail file, **$HOME/mbox**.
x	Writes all mail unchanged to **/var/spool/mail/***UserID* and exits.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To send mail to other users, enter:

   ```
   bellmail tom rachel
   Don't forget the meeting tomorrow at 9:30 a.m.
   ```

   Press Ctrl-D at the end of the message. In this example, the system mails the message to users `tom` and `rachel`.
2. To send a file to another user, enter:

   ```
   bellmail lance <proposal
   ```

   In this example, the file `proposal` is sent to user `lance`.
3. To display your mail, enter:

   ```
   bellmail
   ```

   After the most recent message is displayed, a ? (question mark) indicates the **bellmail** command is waiting for one of the **bellmail** subcommands. Enter `help` or an * (asterisk) to list the subcommands available.
4. To save a message or a file to the default mail file, enter:

   ```
   bellmail
   ```

   This command displays each message mailed to you. Press the Enter key after the ? prompt until the desired file is displayed. When the appropriate file is displayed, enter:

   ```
   s
   ```

   In this example, the file is saved in the default mail file, **$HOME/mbox**.
5. To save a message or a file to a specific file, enter:

   ```
   bellmail
   ```

   This command displays each message mailed to you. Press the Enter key after the ? prompt until the desired file is displayed. When the appropriate file is displayed, enter:

   ```
   s mycopy
   ```

   In this example, the file is saved in a file named `mycopy`, instead of in the default mail file.

Files

Item	Description
$HOME/dead.letter	Unmailable text.
$HOME/mbox	Your personal mailbox.
/usr/mail/*.lock	Lock for mail directory.
/var/spool/mail/*UserID*	Default system mailbox for *UserID.*
/usr/bin/bellmail	Bellmail program.

bffcreate Command

Purpose

Creates installation image files in backup format.

Syntax

bffcreate [**-q**] [**-S**] [**-U**] [**-v**] [**-X**] [**-d** *Device*] [**-t** *SaveDir*] [**-w** *Directory*] [**-M** *Platform*] { [**-l** | **-L**] | [-c [-s *LogFile*] | *Package* [*Level*] ... | **-f** *ListFile* | **all** }

Description

The **bffcreate** command creates an installation image file in backup file format (bff) to support software installation operations.

The **bffcreate** command creates an installation image file from an installation image file on the specified installation media. Also, it automatically creates an installation image file from hyptertext images (such as those on the operating system documentation CD-ROMs). The **installp** command can use the newly created installation file to install software onto the system. The file is created in backup format and saved to the directory specified by *SaveDir*. The **.toc** file in the directory specified by the *SaveDir* parameter is updated to include an entry for the image file.

The **bffcreate** command determines the bff name according to this information:

Item	Description
Neutral Packages	*package.v.r.m.f.platform.installtype*
POWER processor-based platform Packages	*package.v.r.m.f.installtype*

Image Type	Target bff Name
Installation image for the POWER processor-based platform	*package.v.r.m.f.I*
Installation image for Neutral	*package.v.r.m.f.N.I*
3.1 update for the POWER processor-based platform	*package.v.r.m.f.service#*
3.2 update for the POWER processor-based platform	*package.v.r.m.f.ptf*
4.X** or later updates for the POWER processor-based platform	*package.part.v.r.m.f.U*
Update image for Neutral	*package.v.r.m.f.N.U*

** 4.X or later updates contain one *package* only. In addition, AIX Version 4 and later updates do not contain *ptf* IDs.

package = the name of the software package as described by the *PackageName* parameter

v.r.m.f = version.release.modification.fix, the level associated with the software package. The *PackageName* is usually not the same as the *fileset* name.

ptf = program temporary fix ID (also known as FixID)

The installation image file name has the form *Package.Level.I*. The *Package* is the name of the software package, as described for the *Package Name* parameter. *Level* has the format of *v.r.m.f*, where *v* = version, *r* = release, *m* = modification, *f* = fix. The *I* extension means that the image is an installation image rather than an update image.

Update image files containing an AIX 3.1 formatted update have a service number extension following the level. The *Servicenum* parameter can be up to 4 digits in length. One example is `xlccmp.3.1.5.0.1234`.

Update image files containing an AIX 3.2 formatted update have a *ptf* extension following the level. One example is `bosnet.3.2.0.0.U412345`.

AIX Version 4 and later update image file names begin with the *fileset* name, not the *PackageName*. They also have *U* extensions to indicate that they are indeed update image files, not installation images. One example of an update image file is `bos.rte.install.4.3.2.0.U`.

The **all** keyword indicates that installation image files are created for every installable software package on the device.

You can extract a single update image with the AIX Version 4 and later **bffcreate** command. Then you must specify the *fileset* name and the *v.r.m.f.* parameter. As in example 3 in the Examples section, the *PackageName* parameter must be the entire *fileset* name, `bos.net.tcp.client`, not just `bos.net`.

Attention: Be careful when selecting the target directory for the extracted images, especially if that directory already contains installable images. If a *fileset* at a particular level exists as both an installation image and as an update image in the same directory, unexpected installation results can occur. In cases like this, **installp** selects the image it finds first in the table of contents (**.toc**) file. The image it selects may not be the one you intended and unexpected requisite failures can result. As a rule of thumb, you should extract maintenance and technology levels to clean directories.

Flags

Item	Description
-c	Change image names to package name format.
-d *Device*	Specifies the name of the device where the original image resides. The device can be a CD, tape, diskette, or a directory. If the image is contained on tape, the tape device must be specified as no-rewind-on-close and no-retension-on-open (**/dev/rmt*.1** for high-density tape and **/dev/rmt*.5** for low-density tape). The default device is /**dev**/**rfd0**.
-f *ListFile*	Reads a list of *PackageNames* and *Levels* from *ListFile*. *PackageNames*, each optionally followed by a level, should appear one per line of text. Any text following the second set of spaces or tabs on a line is ignored.
-l	Lists the *Package*, *Level*, *Image Type* (*I* for installation images and *U* for update images), and *Part(s)* of all packages on the media.
-M*Platform*	Specifies that any of the following *Platform* values may be used to list or to create backup file format (bff) images of installable software products for a specific platform: **A** Specifies all packages. **N** Specifies platform-neutral packages. **R** Specifies POWER processor-based platform packages only.
-q	Suppresses the request for media.
-s *LogFile*	Save changed image names in file indicated by *LogFile*.
-t *SaveDir*	Specifies the directory where the installation image files are to be created. The **bffcreate** command creates the specified directory if it does not exist. If the **-t** flag is not specified, the files are saved in the **/usr/sys/inst.images** directory.

Item	Description
-U	Upgrades the directory structure of the destination repository to the current standard, if necessary. The current standard requires images to be organized into subdirectories according to package type and architecture. For example, installp images reside in the ***SaveDir*/installp/ppc** directory. When copying from a source containing this structure, the destination is required to conform. Specifying the **-U** flag permits the **bffcreate** command to create the appropriate subdirectory structure in your repository and move any existing images into the appropriate locations. Unless invalid manual copying is performed thereafter, this flag should only need to be used once.
-v	Writes the name of the backup format file to standard output.
-w *Directory*	Specifies the directory where a temporary working directory can be created. The **bffcreate** command creates the specified directory if it does not exist. The default directory is **/tmp**.
-S	Suppresses multiple volume processing when the installation device is a CD-ROM. Installation from a CD-ROM is always treated as a single volume, even if the CD-ROM contains information for a multiple volume CD set. This same suppression of multiple volume processing is performed if the **INU_SINGLE_CD** environment is set.
-X	Automatically extends the file system if space is needed.
-L	Displays information as a list separated by colons.

Security

Access Control

You must have root authority to run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To create an installation image file from the **bos.net** software package on the tape in the **/dev/rmt0** tape drive and use **/var/tmp** as the working directory, type:

   ```
   bffcreate -d /dev/rmt0.1
    -w /var/tmp bos.net
   ```

2. To create an installation image file from the **package** software package on the diskette in the **/dev/rfd0** diskette drive and print the name of the installation image file without being prompted, type:

   ```
   bffcreate -q -v
   package
   ```

3. To create a single update image file from the **bos.net.tcp.client** software package on the CD in **/dev/cd0**, type:

   ```
   bffcreate -d
   /dev/cd0 bos.net.tcp.client 4.2.2.1
   ```

4. To list the packages on the CD in **/dev/cd0**, type:

   ```
   bffcreate -l
    -d /dev/cd0
   ```

5. To create installation and/or update images from a CD in **/dev/cd0** by specifying a list of *PackageNames* and *Levels* in a *ListFile* called my MyListFile, type:

   ```
   bffcreate  -d /dev/cd0
    -f MyListFile
   ```

6. To create installation or update images of all software packages on the CD-ROM media for the current platform, type:

   ```
   bffcreate -d /dev/cd0 all
   ```

7. To list fileset information for the bos.games software package from a particular device, type:

   ```
   bffcreate -d /usr/sys/inst.images/bos.games -l
   ```

8. To list all the Neutral software packages on the CD-ROM media, type:

   ```
   bffcreate -d /dev/cd0 -MN -l
   ```

Files

Item	Description
/usr/sbin/bffcreate	Contains the **bffcreate** command.
/usr/sys/inst.images	Contains files in backup format for use in installing or updating a complete set or subset of software packages.
/usr/sys/inst.images/.toc	The table of contents file for the default directory where a list of installation image files in the directory is maintained.

bfs Command

Purpose

Scans files.

Syntax

bfs [**-**] *File*

Description

The **bfs** command reads a file specified by the *File* parameter, but does not process the file. You can scan the file, but you cannot edit it.

The **bfs** command is basically a read-only version of the **ed** command with two exceptions: the **bfs** command can process much larger files and has additional subcommands.

Input files can be up to 32,767 lines long, with up to 255 characters per line. The **bfs** command is usually more efficient than the **ed** command for scanning a file because the file is not copied to a buffer. The **bfs** command is most useful in identifying sections of a large file that can be divided, using the **csplit** command, into more manageable pieces for editing.

If you enter the **P** subcommand, the **bfs** command prompts you with an * (asterisk). You can turn off prompting by entering a second **P** subcommand. The **bfs** command displays error messages when prompting is turned on.

The **bfs** command runs in both single- and multi-byte environments. The language environment is determined by the setting of the **LANG** environment variable (in the **/etc/environment** file) for the shell.

Forward and Backward Searches

The **bfs** command supports all of the address expressions described under the **ed** command. In addition, you can instruct the **bfs** command to search forward or backward through the file, with or without wraparound. If you specify a forward search with wraparound, the **bfs** command continues searching from the beginning of the file after it reaches the end of the file. If you specify a backward search with wraparound, the command continues searching backwards from the end of the file after it reaches the beginning. The symbols for specifying the four types of search are as follows:

Item	Description
/*Pattern***/**	Searches forward with wraparound for the *Pattern*.
?*Pattern***?**	Searches backward with wraparound for the *Pattern*.
>*Pattern***>**	Searches forward without wraparound for the *Pattern*.
<*Pattern***<**	Searches backward without wraparound for the *Pattern*.

The pattern-matching routine of the **bfs** command differs somewhat from the one used by the **ed** command and includes additional features described in the **regcmp** subroutine. There is also a slight difference in mark names: only lowercase letters a through z may be used, and all 26 marks are remembered.

Flags

Item	Description
-	Suppresses the display of file sizes. Normally, the **bfs** command displays the size, in bytes, of the file being scanned.

Subcommands

The **e**, **g**, **v**, **k**, **n**, **p**, **q**, **w**, **=**, **!**, and null subcommands operate as explained in the **ed** command. However, the **bfs** command does not support a space between the address and the subcommand. Subcommands such as **—**, **+++-**, **+++=**, **-12**, and **+4p** are accepted. **1**,**10p** and **1**,**10** both display the first ten lines. The **f** subcommand displays only the name of the file being scanned; there are no remembered file names. The **w** subcommand is independent of output diversion, truncation, or compression (the **xo**, **xt**, and **xc** subcommands, respectively). *Compressed Output* mode suppresses blank lines and replaces multiple spaces and tabs with a single space.

The following additional subcommands are available:

Item	Description
xf *File*	Reads the **bfs** subcommands from the specified file. When the **bfs** command reaches the end of file or receives an interrupt signal, or if an error occurs, the **bfs** command resumes scanning the file that contains the **xf** subcommand. These **xf** subcommands can be nested to a depth of 10.
xo [*File*]	Sends further output from the **p** and null subcommands to the named file, which is created with read and write permission granted to all users. If you do not specify a *File* parameter, the **bfs** command writes to standard output. Each redirection to a file creates the specified file, deleting an existing file if necessary.
:*Label*	Positions a label in a subcommand file. The label is ended with a newline character. Spaces between the **:** (colon) and the start of the label are ignored. This subcommand can be used to insert comments into a subcommand file, since labels need not be referenced.

Item	Description
[*Address1*[**,***Address2*]] **xb/***Pattern***/***Label*	Sets the current line to the line containing the specified pattern, and jumps to the specified label in the current command file if the pattern is matched within the designated range of lines. The jump fails under any of the following conditions: • The value of either the *Address1* or *Address2* parameter is not between the first and last lines of the file. • The *Address2* value is less than the *Address1* value. • The pattern does not match at least one line in the specified range, including the first and last lines. This subcommand is the only one that does not issue an error message on bad addresses, so it may be used to test whether addresses are bad before other subcommands are run. The subcommand: `xb/^/label` is an Unconditional Jump. The **xb** subcommand is allowed only if it is read from some place other than a workstation. If it is read from a pipe, only a Downward Jump is possible.
xt [*Number*]	Truncates output from the **p** subcommand and the null subcommands to the number of characters. The default value of the *Number* parameter is 192.

Item	Description
xv[*Digit*] [*Value*]	Assigns the specified *Value* to the *Digit* parameter. The value of the *Digit* parameter can be 0 through 9. You can put one or more spaces between *Digit* and *Value*. For example: `xv5 100` `xv6 1,100p` assigns the value `100` to the variable 5 and the value `1,100p` to the variable 6. To reference a variable, put a % (percent sign) in front of the variable name. Given the preceding assignments for variables 5 and 6, the following three subcommands: `1,%5p` `1,%5` `%6` each display the first 100 lines of a file. To escape the special meaning of %, precede it with a \ (backslash). For example: `g/".*\%[cds]/p` matches and lists lines containing **printf** variables (%c, %d, or %s). You can also use the **xv** subcommand to assign the first line of command output as the value of a variable. To do this, make the first character of the *Value* parameter an ! (exclamation point), followed by the command name. For example: `xv5 !cat junk` stores the first line of the `junk` file in the variable 5. To escape the special meaning of `!` as the first character of *Value*, precede it with a \ (backslash). For example: `xv7 \!date` stores the value `!date` in the variable 7.
xbz *Label*	Tests the last saved exit value from a shell command and jumps to the specified label in the current command file if the value is 0.
xbn *Label*	Tests the last saved exit value from a shell command and jumps to the specified label in the current command file if the value is not 0.
xc [*Switch*]	Turns compressed output mode on or off. (Compressed output mode suppresses blank lines and replaces multiple spaces and tabs with a single space.) If the *Switch* parameter has a value of 1, output from the **p** subcommand and the null subcommands is compressed. If the *Switch* parameter is 0, this output is not compressed. If you do not specify a value for the *Switch* parameter, the current value of the *Switch* parameter, initially set to 0, reverses.

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion without any file or command errors
>0	An error occurred.

Files

Item	Description
/usr/bin/bfs	Contains the **bfs** command.

bg Command

Purpose

Runs jobs in the background.

Syntax

bg [*JobID* ...]

Description

If job control is enabled (see **"Job Control in the Korn shell or POSIX shell"** in *Operating system and device management*), the **bg** command resumes suspended jobs in the current environment by running them as background jobs. If the specified job is already running in the background, the **bg** command has no effect and exits successfully. If no *JobID* parameter is supplied, the **bg** command uses the most recently suspended job.

The *JobID* parameter can be a process ID number, or you can use one of the following symbol combinations:

Item	Description
%*Number*	Refers to a job by the job number.
%*String*	Refers to a job whose name begins with the specified string.
%?*String*	Refers to a job whose name contains the specified string.
%+ OR **%%**	Refers to the current job.
%-	Refers to the previous job.

Using the **bg** command to place a job into the background causes the job's process ID to become known in the current shell environment. The **bg** command output displays the job number and the command associated with that job. The job number can be used with the **wait**, **fg**, and **kill** commands by prefixing the job number with a % (percent sign). For example, **kill %3**.

A job is suspended by using the **Ctrl-Z** key sequence. That job can be restarted in the background using the **bg** command. This is effective if the job expects no terminal input and if job output is redirected to non-terminal files. If a background job has terminal output, the job can be forced to stop by entering the following command:

```
stty tostop
```

A background job can be stopped by entering the following command:

```
kill -s stop JobID
```

The **/usr/bin/bg** command does not work when operating in its own command execution environment, because that environment does not have suspended jobs to manipulate. This would be the case in the following example:

```
Command | xargs bg
```

Each **/usr/bin/bg** command operates in a different environment and does not share the parent shell's understanding of jobs. For this reason, the **bg** command is implemented as a Korn shell or POSIX shell regular built-in.

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	An error occurred.

If job control is disabled, the **bg** command exits with an error, and no job is placed in the background.

Examples

If the output of the **jobs** command displays the following stopped job:

```
[2] + Stopped (SIGSTOP)  sleep 100 &
```

use the job number to resume the `sleep 100 &` job by entering:

```
bg %2
```

The screen displays the revised status of job 2:

```
[2] sleep 100 &
```

Files

Item	Description
/usr/bin/ksh	Contains the Korn shell **bg** built-in command.
/usr/bin/bg	Contains the **bg** command.

bicheck Command

Purpose

Syntax checker for user-modified **bosinst.data** files.

Syntax

bicheck *Filename*

Description

The **bicheck** command checks for the existence of the control flow, target_disk_data, and locale stanzas in the **bosinst.data** file. The parameter *Filename* indicates the **bosinst.data** file you want to verify. The value—if not blank—for each field in a stanza is confirmed to match an allowable value, if possible, and checked for length limitations and/or other possible limitations.

If a non-prompted install is specified, the existence of values for required fields is confirmed.

If a dump stanza exists and if the value is not blank, the value is determined to match an allowable value, if possible. It is also checked for length limitations and/or other possible limitations.

The **bicheck** command does not stop after the first error, but continues to list all problems it finds with the given **bosinst.data** file. All error messages are sent to standard error.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
1	An error occurred.

Files

/usr/lpp/bosinst/bicheck contains the **bicheck** command.

biff Command

Purpose

Enables or disables mail notification during the current session.

Syntax

biff [**y** | **n**]

Description

The **biff** command informs the system whether you want to be notified when mail arrives. When mail notification is enabled, From and Subject header lines and the first 7 lines or 560 characters of a message are displayed on the screen when mail arrives. Notification, specified by the **biff y** command, is often included in the **$HOME/.login** or **$HOME/.profile** file to be executed each time the user logs in. The **biff n** command disables notification.

Note: In addition to **y** and **n**, you can use **yes** and **no** to enable and disable mail notification.

The **biff** command operates asynchronously. To receive notification when mail arrives, ensure:

1. The message permission setting is on in your shell (`mesg y`).
2. **comsat** is running (started by the **inetd** daemon).
3. Notification is enabled (`biff y`).

For synchronous notification, use the **MAIL** variable of either the **ksh** command, **bsh** command, or the **csh** command.

Options

Item	Description
y	Enables mail notification.
n	Disables mail notification.

Examples

1. To display the current setting, enter:

   ```
   biff
   ```

2. To be notified during the current terminal session whenever mail arrives, enter the following statement in your **$HOME/.login** or **$HOME/.profile** file:

   ```
   biff y
   ```

   The From and Subject header lines and the first seven lines or 560 characters of the message will be displayed on the screen when mail arrives.

Files

Item	Description
$HOME/.login	Read by **login** shell at login.
$HOME/.profile	Controls start-up processes and daemons.
/usr/bin/biff	Contains **biff** command.

bindintcpu Command

Purpose

Assigns a bus interrupt level to be delivered only to the indicated CPUs.

Syntax

bindintcpu *Level CPU* [*CPU*...]

bindintcpu **-u** *Level*

bindintcpu **-q** *Level*

Description

The **bindintcpu** command lets system administrators direct interrupts from a specific hardware device at the specified bus interrupt *Level* to a specific *CPU* number, or sets of *CPU* numbers. Normally, on multiple CPU systems, hardware device interrupts can be delivered to any running CPU, and the distribution among the CPUs is determined by a predefined method. The **bindintcpu** command lets the system administrator bypass the predefined method, and control the interrupts distribution from a specific device to selected CPUs. This command is applicable only on selective hardware types.

If an interrupt level has been bound with certain CPUs, all interrupts coming from that level will be distributed only to specified CPUs until it is redirected by **bindintcpu** again. If the **-q** flag is used, this utility will instead list to which CPUs the interrupt Level is bound. With the **-u** flag, an administrator can unbind a specified interrupt from its CPUs, and that interrupt will once again be delivered to any running

CPU through some predefined method. However, interrupts bound to **CPU0** cannot be redirected again. If an interrupt level has been bound to **CPU0**, it stays on **CPU0** until the system is booted again.

Notes:

- Not all hardware models support one-to-many bindings, specifying multiple CPUs with **bindintcpu** results in errors on certain types of machines. For consistency, it is recommended to specify one CPU per **bindintcpu** whenever possible.
- The number of interrupts that can be bound to a CPU is dependent on the hardware model. Interrupt binding operations fail with the ENOSPC error code when the binding limit is reached for a CPU.
- To see the bus interrupt level for a specific adapter, use the **lsattr** command and reference the **busintr** field. For example, device ent0 below has busintr value of 6.

```
 lsattr -E -l ent0
busio           0xbff400       Bus I/O address                   False
busintr         6              Bus interrupt level               False
intr_priority   3              Interrupt priority                False
tx_que_size     256            TRANSMIT queue size               True
rx_que_size     256            RECEIVE queue size                True
rxbuf_pool_size 384            RECEIVE buffer poof size          True
media_speed     10_Half_Duplex Media Speed                       True
use_alt_addr    no             Enable ALTERNATE ETHERNET address True
alt_addr        0x000000000000 ALTERNATE ETHERNET address        True
ip_gap          96             Inter-Packet Gap                  True
```

Flags

Item	Description
-q	List to which CPUs the interrupt Level is bound.
-u	Unbinds a specified interrupt from its CPUs.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To direct all interrupts from bus interrupt level 6 to CPU1, enter the following command:

   ```
   bindintcpu 6 1
   ```

2. To direct all interrupts from buss interrupt level 6 to CPU2 and CPU3, enter the following command:

   ```
   bindintcpu 6 2 3
   ```

Files

Item	Description
/usr/sbin/bindintcpu	Contains the **bindintcpu** command.

bindprocessor Command

Purpose

Binds or unbinds the kernel threads of a process to a processor.

Syntax

bindprocessor *Process* [*ProcessorNum*] | **-q** | **-u** *Process*{*ProcessID* [*ProcessorNum*] | -u *ProcessID* | -s *SmtSetID* | -b *bindID ProcessorNum* | -q }

Description

The **bindprocessor** command binds or unbinds the kernel threads of a process, or lists available processors. The *Process* parameter is the process identifier of the process whose threads are to be bound or unbound, and the *ProcessorNum* parameter is the bind CPU identifier of the processor to be used. If the *ProcessorNum* parameter is omitted, the process is bound to a randomly selected processor.

If simultaneous multi-threading is enabled, each hardware thread of a physical processor is listed as a separate processor by the `bindprocessor` command. This allows software threads to be bound to each hardware thread separately. There are two hardware threads on a POWER5 processor, and they are referred to as the *primary hardware thread* and *secondary hardware thread*. The *SmtSetId* parameter is the simultaneous multi-thread set identifier value of a hardware thread and is defined to be `0` for primary hardware threads and `1` for secondary hardware threads. The `-s` flag can be used to list available processors that are all primary hardware threads or that are all secondary hardware threads. The `-b` flag lists all the available hardware threads on a single physical processor on which the *ProcessorNum* parameter is the bind CPU identifier of either the primary hardware thread or the secondary hardware thread on that processor. Refer to **Simultaneous Multi-Threading** in *General Programming Concepts: Writing and Debugging Programs* for more information.

The **bindprocessor** command will fail if the target process has a *Resource Attachment*.

Programs that use processor bindings should become Dynamic Logical Partitioning (DLPAR) aware.

It is important to understand that a process itself is not bound, but rather its kernel threads are bound. Once kernel threads are bound, they are always scheduled to run on the chosen processor, unless they are later unbound. When a new thread is created, it has the same bind properties as its creator. This applies to the initial thread in the new process created by the **fork** subroutine: the new thread inherits the bind properties of the thread which called **fork**. When the **exec** subroutine is called, thread properties are left unchanged.

The **-q** flag of the **bindprocessor** command lists the available bind CPU identifiers: you can use the logical numbers given as values for the *ProcessorNum* parameter. The **-u** flag unbinds the threads of a process, allowing them to run on any processor.

When simultaneous multi-threading is enabled, the `-s` flag of the `bindprocessor` command allows you to bind the threads of an application to separate physical processors by listing the processors separately. The `-b` flag is useful if you want to bind all the threads of an application to the hardware threads of the same physical processor.

Notes:

1. The **bindprocessor** command is meant for multiprocessor systems. Although it will also work on uniprocessor systems, binding has no effect on such systems.
2. You need root authority to bind or unbind threads in processes you do not own.
3. If you attempt to bind kernel processes such as **swapper** and **sched** from the user space, the operation fails with the **EPERM** error code. You can determine which kernel processes will fail by looking for the **SSCHEDPROC** flag in the process structure. If the **SSCHEDPROC** flag is set, binding the kernel process will fail.

Flags

Item	Description
-b	Binds all threads of an application to the hardware threads of the same physical processor.
-q	Displays the processors which are available.
-s	Binds all threads of an application to separate physical processors by listing the processors separately.
-u	Unbinds the threads of the specified process.

Examples

1. To see which processors are available (possible *ProcessorNum* values*)*, type:

   ```
   bindprocessor -q
   ```

   For a four processor system, the output is similar to:

   ```
   The available processors are: 0 1 2 3
   ```

2. To bind the threads in process 19254 to processor 1, type:

   ```
   bindprocessor 19254 1
   ```

3. To see all the available processors that are primary hardware threads, type:

   ```
   bindprocessor -s 0
   ```

   For a four-processor system with simultaneous multi-threading enabled, the output is similar to:

   ```
   The available processors are: 0 2 4 5
   ```

   To see all the available processors that are secondary hardware threads, type:

   ```
   bindprocessor -s 1
   ```

   The output is similar to:

   ```
   The available processors are: 1 3 6 7
   ```

   When simultaneous multi-threading is disabled using the `smtctl` command, or on systems with processors that do not support simultaneous multi-threading, the outputs would be:

   ```
   bindprocessor -s 0
   The available processors are: 0 1 2 3
   bindprocessor -s 1
   SmtSetId 1 is not available
   ```

4. To see all the available bind CPU IDs on a physical processor that has a hardware thread with a bind CPU ID of `0`, type:

   ```
   bindprocessor -b 0
   ```

   The output is similar to:

   ```
   The available processors are: 0 1
   ```

   Again, typing the command:

```
bindprocessor -b 1
```

will also result in the same output.

File

Item	Description
/usr/sbin/bindprocessor	Contains the **bindprocessor** command.

binld Daemon

Purpose

Implements a Preboot Execution Environment (PXE) boot server. Serves boot file transfer server addresses and determines the appropriate boot file for PXE clients.

Syntax

To serve boot file information to the PXE clients using the system resource controller:

startsrc -s **binld** [**-a**] ...

To serve boot file information to the PXE clients without using the system resource controller:

binld [**-f**] [**-i**]

Description

The BINLD server assigns boot files for PXE clients and informs the clients where they should download the boot file. The BINLD daemon runs in the background and maintains a database of boot files that it serves and the client information (client architecture, client machine identifier, major and minor version of the network identifier) that is appropriate for each boot file. The initial boot file database is specified by the configuration file. The configuration file also contains all the data needed to assign PXE clients their boot file information.

On startup, a BINLD server reads the configuration file and sets up its initial database of available boot files. The BINLD server accepts the **refresh** command or a SIGHUP signal to reread the configuration file.

Flags

Item	Description
-a	The argument to be supplied.
-f	ConfigurationFile. Specifies the configuration file to be used.
-i	IP address. Specifies to which DHCP server IP address the DHCPINFORM should be sent.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>	An error occurred.

Security

Access Control: You must have root authority to run this command.

Files

Item	Description
/usr/sbin/binld	Contains the BINLD daemon.

biod Daemon

Purpose

Handles client requests for files.

Syntax

/usr/sbin/biod *NumberOfBiods*

Description

The **biod** daemon is retained for compatibility with earlier versions with scripts that invoke it. It no longer plays an active role in management of the NFS client subsystem. Instead, the NFS client internally manages its resources for performing I/O operations to NFS servers.

The *NumberOfBiods* argument historically allowed control of NFS client thread resources for performing I/O operations. This no longer has any effect. The maximum number of `biod` threads for I/O operations can be set as a mount option. The `biod` daemon might be removed in future AIX releases.

Files

Item	Description
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.

bj Command

Purpose

Starts the blackjack game.

Syntax

bj

Description

The **bj** command invokes the blackjack game. Blackjack is a card game. The object of blackjack is to be dealt cards with a value of up to but not over 21 and to beat the dealer's hand. The computer plays the role of the dealer in blackjack.

You place bets with the dealer on the likelihood that your hand will come equal or closer to 21 than will the dealer's. The following rules apply to betting.

The bet is two dollars every hand. If you draw a natural blackjack, you win three dollars. If the dealer draws a natural blackjack, you lose two dollars. If you and the dealer both have natural blackjacks, you exchange no money (a push).

If the dealer has an ace showing, you can make an insurance bet on the chance that the dealer has a natural blackjack, winning two dollars if the dealer has a natural blackjack and losing one dollar if not.

If you are dealt two cards of the same value, you can double, that is, play two hands, each of which begins with one of these cards, betting two dollars on each hand. If the value of your original hand is 10 or 11, you can double down, that is, double the bet to four dollars and receive exactly one more card in that hand.

Under normal play, you can draw a card (take a hit) as long as your cards total 21 or less. If the cards total more than 21, you bust and the dealer wins the bet. When you stand (decide not to draw another card), the dealer takes hits until a total of 17 or more is reached. If the dealer busts, you win. If both you and the dealer stand, the one with the higher total below or equal to 21 wins. A tie is a push.

The computer deals, keeps score, and asks the following questions at appropriate times: `Do you want a hit? Insurance? Double? Double down?` To answer yes, press Y; to answer no, press the Enter key.

The dealer tells you whenever the deck is being shuffled and displays the action (total bet) and standing (total won or lost). To quit the game, press the Interrupt (Ctrl-C) or End Of File (Ctrl-D) key sequence; the computer displays the final action and score and exits.

Files

Item	Description
/usr/games	Location of the system's games.

bootauth Command

Purpose

Allows only the authorized user to boot the system.

Syntax

bootauth

Description

The **bootauth** command verifies that the system is being started by an authorized user.

The **bootauth** command prompts you for a user name and a password. If the user name and the password entered are not valid, or if the user name does not have the **aix.system.boot** authorization, the **bootauth** command reissues the prompt. After three unsuccessful attempts, the system is restarted.

Security

To start the system successfully, you must have the following authorization:

Item	Description
aix.system.boot	Required to start the system.

Files

Item	Description
/usr/sbin/bootauth	Contains the **bootauth** command.

bootlist Command

Purpose

Displays and alters the list of boot devices available to the system.

Syntax

bootlist [{ **-m** *Mode* } [**-r**] [**-o**] [[**-i**] [**-V**] [**-F**]| [[**-f** *File*] [*Device* [*Attr*=*Value* ...] ...]]] [**-v**]

Description

The **bootlist** command allows the user to display and alter the list of possible boot devices from which the system may be booted. When the system is booted, it will scan the devices in the list and attempt to boot from the first device it finds containing a boot image. This command supports the updating of the following:

- Normal boot list. The normal list designates possible boot devices for when the system is booted in normal mode.
- Service boot list. The service list designates possible boot devices for when the system is booted in service mode. How a system is booted in service mode is hardware-platform dependent. It may require a key switch to be turned to the Service position, a particular function key to be pressed during the boot process, or some other mechanism, as defined for the particular hardware platform.
- Previous boot device entry. This entry designates the last device from which the system booted. Some hardware platforms may attempt to boot from the previous boot device before looking for a boot device in one of the other lists.

Support of these boot lists may vary from platform to platform. A boot list can be displayed or altered only if the platform supports the specified boot list. It may even be the case that a particular hardware platform does not support any of the boot lists.

When searching for a boot device, the system selects the first device in the list and determines if it is bootable. If no boot file system is detected on the first device, the system moves on to the next device in the list. As a result, the ordering of devices in the device list is extremely important.

The **bootlist** command supports the specification of generic device types as well as specific devices for boot candidates. Possible device names are listed either on the command line or in a file. Devices in the boot device list occur in the same order as devices listed on the invocation of this command.

The devices to be entered into the boot list may be specified in a file. This makes an alterable record of the boot devices available for reference or future update. When the **-f** flag is used, the list of devices is taken from the file specified by the *file* variable. Devices from this list are then placed in the boot list in the order found in the file.

> **Attention:** Care must be taken in specifying the possible boot devices. A future reboot may fail if the devices specified in the device list become unbootable. The system must not be powered off or reset during the operation of the **bootlist** command. If the system is reset, or if power fails at a critical point in the execution of this command, the boot list may be corrupted or lost.

The selection of the boot list to display or alter is made with the **-m** *mode* option, where the *mode* variable is one of the keywords: **service**, **normal**, **both**, or **prevboot**. If the **both** keyword is specified, then both the normal boot list and the service boot list will be displayed, or if being altered, will be set to the same list of devices. If the **prevboot** keyword is specified, the only alteration allowed is with the **-i** (invalidate) flag . The **-i** flag invalidates the boot list specified by the **-m** flag.

The devices currently in the boot list may be displayed by using the **-o** flag. The list of devices that make up the specified boot list will be displayed, one device per line. If a device specified in the boot list is no longer present on the system, a `-' is displayed instead of a name. The output is in a form that can be captured in a file and used as input to the **bootlist** command with the **-f** flag. This may be useful for restoring a boot list after making a temporary change.

Note: When you add a hot plug adapter to the system, that adapter and its child devices might not be available for specification as a boot device when you use the **bootlist** command. You may be required to reboot your system to make all potential boot devices known to the operating system.

When you specify a disk device, additional information might need to be added to the disk by using an *attribute*=*value* pair. This extra information is required when the target disk has multiple instances of the AIX operating system installed on it, or it is required to indicate a path ID when you specify the boot device. When the target disk has multiple instances of the AIX operating system installed on it, identify the boot logical volume on the target disk that is to be included in the boot list by using the `blv` attribute.

The `blv` attribute can be used in all cases, but it is only required when the target disk has multiple instances of AIX installed. When `bootlist` displays information with the `-o` flag, the `blv` attribute is always included for each disk, even if there is only one instance of AIX on that disk.

When you specify a path ID, identify the path ID of the target disk by using the `pathid` attribute. You can specify one or more path IDs with the `pathid` attribute by entering a comma-separated list of the required paths to be added to the boot list. When the `bootlist` command displays information with the `-o` flag, the `pathid` attribute is included for each disk that has an associated path ID.

Device Choices

The device name specified on the command line (or in a file) can occur in one of two different forms:

- It can indicate a specific device by its device logical name.
- It can indicate a generic or special device type by keyword. The following generic device keywords are supported:

Item	Description
fd	Any standard I/O-attached diskette drive
scdisk	Any SCSI-attached disk (including serial-link disk drives)
badisk	Any direct bus-attached disk
cd	Any SCSI-attached CD-ROM
rmt	Any SCSI-attached tape device
ent	Any Ethernet adapter
tok	Any Token-Ring adapter
fddi	Any Fiber Distributed Data Interface adapter

Note: Some hardware platforms do not support generic device keywords. If a generic device keyword is specified on such a platform, the update to the boot list is rejected and this command fails.

When a specific device is to be included in the device list, the device's logical name (used with system management commands) must be specified. This logical name is made up of a prefix and a suffix. The suffix is generally a number and designates the specific device. The specified device must be in the Available state. If it is not, the update to the device list is rejected and this command fails. The following devices and their associated logical names are supported (where the bold type is the prefix and the *xx* variable is the device-specific suffix):

Item	Description
fd*xx*	Diskette-drive device logical names
hdisk*xx*	Physical-volume device logical names
cd*xx*	SCSI CD-ROM device logical names
rmt*xx*	Magnetic-tape device logical names
ent*xx*	Ethernet-adapter logical names

Item	Description
tok*xx*	Token-ring adapter logical names
fddi*xx*	Fiber Distributed Data Interface adapter logical names

Attribute Choices

Attributes are extra pieces of information about a device that the user supplies on the command line. Since this information is specific to a particular device, generic devices do not have attributes. Attributes apply to the device that immediately precedes them on the command line, which allows attributes to be interspersed among devices on the command line. Currently, only network devices have attributes. These are:

Item	Description
bserver	The IP address of the BOOTP server
gateway	The IP address of the gateway
client	The IP address of the client
speed	Network adapter speed
duplex	The mode of the network adapter
vlan_tag	The virtual local area network (VLAN) identification value. Valid values are 0 - 4094.
vlan_pri	The VLAN priority value. Valid values are 0 - 7.
filename	The name of the file that is loaded by Trivial File Transfer Protocol (TFTP) from the BOOTP server

These attributes can be combined in the following ways:

- The **hardware** attribute cannot be specified alone; it must be specified with the **bserver** or **gateway** attribute. When specified with **bserver** or **gateway**, it applies to the server or gateway, respectively; when both **bserver** and **gateway** are specified, **hardware** will apply to **gateway**.
- The **bserver** attribute can be specified alone, with **hardware**, and/or **gateway**.
- If the **gateway** attribute is specified, **bserver** and **client** must also be specified.
- The **client** attribute can only be specified with **gateway** and **bserver**.
- The **vlan_pri** attribute must be specified with the **vlan_tag** attribute. The **vlan_tag** attribute can be specified alone.

Some of these attributes may not be supported on some hardware platforms. Additional hardware platform restrictions may apply.

The syntax for specifying an attribute is *attr=value*, where *attr* is the attribute name, *value* is the value, and there are no spaces before or after the =.

File Format When Using the -f Flag

The file specified by the *file* variable should contain device names separated by white space:

```
hdisk0 hdisk1 cd1
```

or one device per line:

```
hdisk0
hdisk1
cd1
```

Error Handling

If this command returns with an error, the device lists are not altered. The following device list errors are possible:

- If the user attempts to display or alter a boot list that is not supported by the hardware platform, the command fails, indicating the mode is not supported.
- If an invalid keyword, invalid flag, or an unknown device is specified, the command fails with the appropriate error message.
- If a specified device is not in the Available state, the command fails with the appropriate error message.

If you add too many devices to the boot list, the command adds only the number of devices to the boot list that the system supports.

Flags

Item	Description
Device	Provides the names of the specific or generic devices to include in the boot list.
-f *File*	Indicates that the device information is to be read from the specified file name.
-F	Indicates that the boot list must be modified even if the validation of the **speed** and **duplex** attributes, if specified, is not possible.
-i	Indicates that the device list specified by the **-m** flag should be invalidated.
-m *Mode*	Specifies which boot list to display or alter. Possible values for the *mode* variable are **normal**, **service**, **both**, or **prevboot**.
-o	Indicates that the specified boot list is to be displayed after any specified alteration is performed. The output is a list of device names.
-r	Indicates that the specified boot list is to be displayed after any specified alteration is performed. The output is hardware-platform dependent. It may be a hexadecimal dump of the boot list or a list of device names. (This is normally used for problem determination.)
-V	Indicates that the speed and duplex attributes, if specified, are to be verified only. The boot list is not modified.
`-v`	Displays verbose output. This flag is for problem determination only.

Security

Privilege Control

Only the root user and members of the security group should have execute (x) access to this command.

Auditing Events

Event	Information
NVRAM_Config	File name

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To invalidate the Service mode boot list, enter:

   ```
   bootlist -m service -i
   ```

2. To make a boot list for Normal mode with devices listed on the command line, enter:

   ```
   bootlist -m normal hdisk0 hdisk1 rmt0 fd
   ```

3. To make a boot list for Normal mode with a device list from a file, enter:

```
bootlist -m normal -f /bootlist.norm
```

   where **bootlist.norm** is a file containing device names to be placed in the boot list for Normal mode. The device names in the **bootlist.norm** file must comply with the described format.
4. To invalidate the previous boot device entry, enter:

```
bootlist -m prevboot -i
```

5. To boot from a Token-Ring device in slot 2, enter:

```
bootlist -m normal tok0
```

6. To attempt to boot through a gateway using Ethernet, and then try other devices, enter:

```
bootlist -m normal ent0 gateway=129.35.21.1 bserver=129.12.2.10
\ client=129.35.9.23 hdisk0 rmt0 tok0 bserver=129.35.10.19
hdisk1
```

7. To specify boot logical volume hd5 on disk hdisk0 for a normal boot, type:

```
bootlist -m normal hdisk0 blv=hd5
```

8. To view the boot list set in the preceding example, type:

```
bootlist -m normal -o
hdisk0 blv=hd5
```

9. To specify booting in normal mode from the only boot logical volume on hdisk0, or the mb_hd5 boot logical volume on hdisk1, type:

```
bootlist -m normal hdisk0 hdisk1 blv=mb_hd5 cd0
```

10. To view the boot list set in the preceding example, type:

```
bootlist -m normal -o
hdisk0
hdisk1 blv=mb_hd5
cd0
```

11. To specify path ID 0 on disk hdisk0 for a normal boot operation, type:

```
bootlist -m normal hdisk0 pathid=0
```

12. To specify path ID 0 and path ID 2 on disk hdisk0 for a normal boot operation, type one of the following commands:

    - `bootlist -m normal hdisk0 pathid=0,2`
    - `bootlist -m normal hdisk0 pathid=0 hdisk0 pathid=2`

bootparamd Daemon

Purpose

Provides information for booting to diskless clients.

Syntax

/usr/sbin/rpc.bootparamd [**-d**]

Description

The **bootparamd** daemon is a server process that provides information necessary to diskless clients for booting. It consults either the **bootparams** database or the **/etc/bootparams** file if the NIS service is not running.

Flags

Item	Description
-d	Displays debugging information.

Files

Item	Description
/etc/bootparams	Contains the list of client entries that diskless clients use for booting.

bootpd Daemon

Purpose

Sets up the Internet Boot Protocol server.

Syntax

bootpd [**-s**] [**-t** *Integer*] [**-d** [**-d** ...]] [**-g**] [*ConfigFile* [*DumpFile*]]

Description

The **bootpd** command implements an Internet Boot Protocol server.

The **bootpd** daemon is normally started by the **inetd** daemon. The default **/etc/inetd.conf** file contains the following line:

```
bootps dgram udp wait root /usr/sbin/bootpd bootpd
```

By default, this entry is commented out. One way to add the **bootpd** daemon to the **inetd** daemon's list of available subservers is to use the System Management Interface Tool (SMIT). Another way to make the **bootpd** daemon available is to edit the **/etc/inetd.conf** file, uncomment the `bootps` entry, and enter `refresh -s inetd` or `kill -1 InetdPid` to inform the **inetd** daemon of the changes to its configuration file. Now, when a bootp request arrives, **inetd** starts the **bootpd** daemon. Once the daemon is started, **bootpd** continues to listen for boot requests. However, if the server does not receive a boot request within 15 minutes of the previous one, it exits to conserve system resources. This time-out value of 15 minutes can be changed using the **-t** flag.

To start the **bootpd** daemon without **inetd**, use the `-s` flag. In this mode, the **bootpd** daemon continues to listen for bootp requests until the daemon is killed.

Upon startup, the **bootpd** daemon looks in the **/etc/services** file to find the port numbers to use, and extracts the following entries:

Item	Description
bootps	The BOOTP server listening port.
bootpc	The destination port used to reply to clients.

Then, the **bootpd** daemon reads its configuration file. If a configuration file is not specified, the default file is **/etc/bootptab**. Once the configuration file is read, the **bootpd** daemon begins listening for and processing bootp requests. The **bootpd** daemon rereads its configuration file when it receives a **SIGHUP** hang-up signal, or when it receives a bootp request packet and detects that the file has been updated. Hosts may be added, deleted, or modified when the configuration file is reread.

Flags

Item	Description
-d	Increases the level of debugging output. This flag can be used many times. The following table displays the levels of debugging that are available: **Debug Level Syntax Message** 1 -d Only error messages. 2 -d -d Level 1 messages and messages indicating potential errors. 3 -d -d -d ... Level 1 and level 2 and general information messages. If the debug level is set to >0 and if the **syslogd** daemon is running, then all debug messages are printed in the **syslogd** log file.
-g	Keeps the same gateway IP address that is in **bootp** request in **bootp** reply.
-s	Runs the **bootpd** command in a stand-alone configuration. This mode is used for large network installations with many hosts. In this case, the **-t** flag has no effect since the **bootpd** command never exits.
-t	Specifies a different time-out value in minutes, such as `-t20`. A time-out value of `0` means forever. The default time-out value is 15 minutes.
ConfigFile	Specifies the configuration file. The default configuration file is **/etc/bootptab**.
DumpFile	Specifies the file into which the **bootpd** daemon dumps a copy of the bootp server database. The default dump file is **/etc/bootpd.dump**.

Examples

1. To start the bootpd daemon in a stand-alone mode, enter the following:

```
/usr/sbin/bootpd -s
```

2. To start the **bootpd** daemon in a stand-alone mode with a debug level of 3, with a configuration file of **/etc/newconfig**, and a dump file of **/etc/newdumpfile**, enter the following:

```
/usr/sbin/bootpd -s -d -d -d /etc/newconfig /etc/newdumpfile
```

Files

Item	Description
/etc/bootpd.dump	The default **bootpd** dumpfile
/etc/bootptab	The default **bootpd** configuration file.
/etc/services	Defines sockets and protocols used for Internet services.
/etc/inetd.conf	Contains the configuration information for the **inetd** daemon.

bootptodhcp Command

Purpose

To convert a BOOTP configuration file into a DHCP configuration file or to remove BOOTP configuration information for a particular host from the DHCP configuration file.

Syntax

To Convert a BOOTP Configuration File into a DHCP Configuration File

/usr/sbin/bootptodhcp [**-d** *DHCPFile*] [**-b** *BOOTPFile*]

To Remove a BOOTP Configuration Information From a DHCP Configuration File

/usr/sbin/bootptodhcp [**-d** *DHCPFile*] **-r** *HostName*]

Description

The **bootptodhcp** command has two functions. The first is to translate a BOOTP configuration file into a DHCP configuration. The default command with no arguments translates the **/etc/bootptab** file. The filenames may be changed by using the **-b** or **-d** flags to specify a different file names.

The second function of the **bootptodhcp** command is the removal of a BOOTP client's information from a DHCP configuration file. The **-r** flag specifies which client to remove from the file. If the **-d** flag is not used.

Flags

Item	Description
-b *BOOTPFile*	Specifies the BOOTP configuration file. The default is **/etc/bootptab**.
-d *DHCPFile*	Specifies the DHCP configuration file.
-r *HostName*	Specifies the hostname of a BOOTP section to delete from the DHCP configuration file.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: Any User

Files Accessed: Need appropriate access permissions for files

Files

Item	Description
/usr/sbin/bootptodhcp	Contains the **bootptodhcp** command.
/etc/bootptab	Contains the default configuration file for bootpd.

bosboot Command

Purpose

Creates boot image.

Syntax

For General Use:

bosboot -*Action* [**-d** *Device*] [**-***Options* ...]

To Create a Device Boot Image:

bosboot {**-a** **-v**} [**-d** *Device*] [**-p** *Proto*] [**-k** *Kernel*] [**-I**|**-D**] [**-l** *LVdev*] [**-L**] [**-M** { *primary*| *standby*|*both* }] [**-T** *Type*] [**-b** *FileName*] [**-q**]

Description

The **bosboot** command creates the boot image that interfaces with the machine boot ROS (Read-Only Storage) EPROM (Erasable Programmable Read-Only Memory).

The **bosboot** command creates a boot file (boot image) from a RAM (Random Access Memory) disk file system and a kernel. This boot image is transferred to a particular media that the ROS boot code recognizes. When the machine is powered on or rebooted, the ROS boot code loads the boot image from the media into memory. ROS then transfers control to the loaded images kernel.

The associated RAM disk file system contains device configuration routines that make the machine's devices and file systems available. The RAM disk file system contains differing configuration files depending upon the boot device. A **mkfs** prototype file is supplied for each type of device. (See note 6 below.) Currently supported devices are:

- CD-ROM
- Disk
- Tape
- Network

A network device may be a token ring, Ethernet, or Fiber-Distributed Data Interface (FDDI) used to boot from a network boot server over a local area network (LAN).

The boot image varies for each type of device booted and is compressed to fit on certain media and to lessen real memory requirements. The boot logical volume must be large enough for the boot image.

In addition to creating a boot image, the **bosboot** command always saves device configuration data for disk. It does not update the list of boot devices in the NVRAM (nonvolatile random access memory). You can modify the list with the **bootlist** command.

The **bosboot** command is usually called during the Base Operating System installation and by the **updatep** command when the operating system is upgraded.

Note:

1. You must have root user authority to use the **bosboot** command.
2. Do not reboot the machine if the **bosboot** command is unsuccessful with a message not to do so while creating a boot disk. The problem should be resolved and the **bosboot** command run to successful completion.
3. The **bosboot** command requires some space in the **/tmp** file system and the file system where the target image is to reside, if there is such an image.
4. The **bosboot** command requires that the specified physical disk contain the boot logical volume. To determine which disk device to specify, issue the following command:

   ```
   lsvg -M rootvg
   ```

   This command displays a map of all logical volumes. The default boot logical volume is **hd5**. Use the disk device that contains the boot logical volume.
5. When the device is not specified with the **-d** flag, the **bosboot** command assumes the default device is the disk the system is booted from. However, if the prototype file is specified with a **-p** flag, the device must also be specified with a **-d** flag.

6. The prototype file used by the **bosboot** command to build the RAM disk file system depends on the boot device and the hardware platform (**sys0**) type of the machine the boot image will run on.

   The hardware platform type is an abstraction which allows machines to be grouped according to fundamental configuration characteristics such as number of processors or I/O bus structure or both. Machines with different hardware platform types will have basic differences in the way their devices are dynamically configured at boot time. The hardware platform type **rs6k** in AIX 5.1 and earlier applies to all Micro Channel-based uni-processor models through AIX 5.1 only. The type **rs6ksmp** applies to all Micro Channel-based symmetric multi-processor models through AIX 5.1 only. The type **rspc** in AIX 5.1 and earlier applies to all ISA-bus models. As new models are developed, their hardware platform types will either be one of the aforementioned types or, if fundamental configuration differences exist, new types will be defined. Boot images for a given boot device type will generally be different for machines with different hardware platform types.

   "The prototype file used by **bosboot** is constructed by starting with a copy of the base prototype file for the platform type and boot device (for example, **/usr/lib/boot/chrp.disk.proto**). Next the **bosboot** command looks at the **pcfg** file for the platform type being used (for example, **/usr/lib/boot/chrp.pcfg**). The **pcfg** file contains entries which bosboot uses in a template to search for proto extension files. These files, located in the directory **/usr/lib/boot/protoext**, provide extensions to the prototype file under construction. For example, if the platform type is **chrp** and the boot device is **disk**, and the file **/usr/lib/boot/protoext/chrp.pcfg** contains the following:

   ```
   scsi.
   chrp.
   chrp_lpar.
   fcp.
   graphics.
   ide.
   isa_sio.
   pci.
   ssa.
   sys.pci.
   tty.
   usbif.
   ```

   The **bosboot** command will start with the base prototype file **/usr/lib/boot/chrp.disk.proto**, and search the directory **/usr/lib/boot/protoext** for any files that match the template **disk.proto.ext.scsi.***. The contents of these files are added to the prototype file under construction. Next, the contents of files matching the template **/usr/lib/boot/protoext/disk.proto.ext.scsi.*** are added to the prototype file under construction. This continues until all lines in the **pcfg** file have been processed. At this point the prototype file under construction is complete. The **bosboot** command passes this prototype file to the **mkfs** command which builds the RAM disk file system.

7. The prototype files used by the BOSBOOT command to build boot images are dependent on the boot device. In addition, the prototype files are dependent on the system device type (sys0) of the machine for which the boot image is built.

   This is reflected in the names of these prototype files:

   /usr/lib/boot/chrp.disk.proto

   /usr/lib/boot/chrp.cd.proto

   /usr/lib/boot/chrp.tape.proto

   /usr/lib/boot/network/chrp.ent.proto

   /usr/lib/boot/network/chrp.tok.proto

   /usr/lib/boot/network/chrp.atm.proto

   /usr/lib/boot/network/chrp.fddi.proto

   The system device type is an abstraction that allows machines to be grouped according to fundamental configuration characteristics, such as number of processors and I/O bus structure. The system device is the highest-level device in the system node, which consists of all physical devices in the system.

Machines with different system device types have basic differences in the way their devices are dynamically configured at boot time.

The **bosboot** command, by default, uses the prototype file that matches the system device type of the machine executing the command. The -p option allows you to specify the system device type of the prototype file.

8. If the boot disk is removed from a running system, thus leaving the system operating from a replacement copy of that disk, you may experience an error message when you run the **bosboot** command. The error message states that the boot logical volume does not exist on the disk. This happens because the **bosboot** command, when called without the *-d* argument, defaults to the disk that the system most recently booted from. In this scenario, since that disk is no longer available, you will need to call the **bosboot** command with the *-d* argument, and the name of the disk on which the boot logical volume now resides. This provides the **bosboot** command with the information that is needed for identifying the new location of the boot image.

Flags

Item	Description
-d *device*	Specifies the boot device. This flag is optional for hard disk.

The following flags are action flags. One and only one flag must be specified.

Item	Description
-a	Creates complete boot image and device.
-v	Verify, but do not build boot image.

The following flags are option flags:

Item	Description
-b *FileName*	Uses specified file name as the boot image name. This flag is optional.
-D	Loads the low level debugger. This flag is optional.
-I (upper case i)	Loads and invokes the low-level debugger. This flag is optional.
-k *Kernel*	Uses the specified kernel file for the boot image. This flag is optional, and if not specified, **/unix** is the default.
-L	Enables lock instrumentation for MP systems. This flag has no effect on systems that are not using the MP kernel.
-l (lower case L) *LVDev*	Uses target boot logical volume for boot image. This flag is optional.
-M *primary*|*standby*|*both*	Specifies which boot pointer table entry to update. The options are: **primary** Specifies the table entry that was most recently used. **standby** Specifies the table entry that was not most recently used. **both** Specifies both boot pointer table entries.
-p *Proto*	Uses the specified prototype file for the RAM disk file system. This flag is optional.

Item	Description
-q	Determines how much disk space is required in which file system to create the boot image. Boot image is not created. This flag is optional.
-T *Type*	Specifies the hardware platform type (see note 6). This causes the **bosboot** command to create a boot image for the hardware platform type specified. If the type is not specified, the **bosboot** command creates a boot image whose hardware platform type matches that of the currently running machine. This flag is optional.

Security

Access Control: Only the root user can read and execute this command.

Examples

1. To create a boot image on the default boot logical volume on the fixed disk from which the system is booted, type:

   ```
   bosboot -a
   ```

2. To create a bootable image called **/tmp/tape.bootimage** for a tape device, type:

   ```
   bosboot -ad /dev/rmt0 -b /tmp/tape.bootimage
   ```

3. To create a boot image file for an Ethernet boot, type:

   ```
   bosboot -ad /dev/ent0
   ```

4. To create a token ring boot image for a machine whose hardware platform type is **chrp** while you are running on a machine whose hardware platform type is **chrp**, type:

   ```
   bosboot -ad /dev/tok -T chrp
   ```

Files

Item	Description
/usr/sbin/mkboot	Specifies boot creation routine.
/usr/lib/boot/chrp.disk.proto	Specifies the disk RAM file system template.
/usr/lib/boot/chrp.cd.proto	Specifies the CD-ROM RAM file system template.
/usr/lib/boot/chrp.tape.proto	Specifies the tape RAM file system template.
/usr/lib/boot/network/chrp.ent.proto	Specifies the Ethernet RAM file system template.
/usr/lib/boot/network/chrp.tok.proto	Specifies the token-ring RAM file system template.
/usr/lib/boot/network/chrp.atm.proto	Specifies the ATM file system template.
/usr/lib/boot/network/chrp.fddi.proto	Specifies the FDDI RAM file system template.

bosdebug Command

Purpose

Enables, disables, and/or displays the status of debugging features of the system.

Syntax

bosdebug [**-b**] [**-D** | **-I**] [**-K on | off**] [**-M**] [**-n sizelist**] [**-R on | off**] [**-M**] [**-s sizelist** | **-S**]

bosdebug [**-f** | **-l** <*file*>]

bosdebug [**-h**]

bosdebug [**-L**]

bosdebug [**-o**]

Description

The **bosdebug** command enables, disables, and/or displays the status of debugging features of the system.

Item	Description
-b	Disables data collection of state information for backtracking faults. This information is useful for debugging certain kinds of kernel errors. Disabling state information data collection for backtracking faults can provide a slight performance improvement under certain rare workloads, but that disablement does not allow the preservation of data that might be critical for problem analysis.
-D	Causes the kernel debug program to be loaded on each subsequent reboot.
-I	Causes the kernel debug program to be loaded and invoked on each subsequent reboot.
-L	Displays the current settings for the kernel debug program and the memory overlay detection system. Note that the settings shown will not take effect until after the next time that the **bosboot -a** and **shutdown -r** commands are run. This is the default.
-K on | off	Sets the state of kernel extension allocation tracking.
-o	Turns off all debugging features of the system.
-R on | off	Activates or deactivates the real-time kernel option. When `-R on` is specified, the kernel proactively generates an extra interrupt to ensure rapid response to a cross-CPU preemption request when the preempting thread is considered a real-time thread. Without this extra interrupt (called an *MPC*), the preempted thread might continue to run uninterrupted until the next regularly scheduled timer tick, or generally up to 10 ms. Threads running with a fixed priority policy are considered real time by default. If `RT_MPC=ON` is exported in the environment before a process is started, that process's threads are also considered real time. Note that while the extra MPC interrupts reduce preemption latency, they also add overhead. Consider this additional overhead before exporting `RT_MPC=ON` in the default environment.
-l <file>	Loads a symbol file into kernel for the **kdb** debugger print facility. Loads the symbols immediately. Do not reboot. A symbol file to print LFS structures may be created as follows: `# echo '#include <sys/vnode.h>' > sym.c` `# echo 'main() { ; }' >> sym.c` `# cc -g -o sym sym.c -qdbxextra /* for 32 bit kernel */` `# cc -g -q64 -o sym sym.c -qdbxextra /* for 64 bit kernel */`
-f	Flushes all the symbols (loaded through **-l** option) from kernel memory. Flushed immediately. Does not require a reboot.
-M	Causes the memory overlay detection system to be enabled. Memory overlays in kernel extensions and device drivers will cause a system crash.

Item	Description
-s sizelist	Causes the memory overlay detection system to promote each of the specified allocation sizes to a full page, and allocate and hide the next subsequent page after each allocation. This causes references beyond the end of the allocated memory to cause a system crash. sizelist is a list of memory sizes separated by commas. Each size must be in the range from 16 to 2048, and must be a power of 2.
-S	Causes the memory overlay detection system to promote all allocation sizes to the next higher multiple of page size (4096), but does not hide subsequent pages. This improves the chances that references to freed memory will result in a crash, but it does not detect reads or writes beyond the end of allocated memory until that memory is freed.
-n sizelist	Has the same effect as the -s option, but works instead for network memory. Each size must be in the range from 32 to 2048, and must be a power of 2. This causes the **net_malloc_frag_mask** variable of the **no** command to be turned on during boot.
-h	Displays the usage message for this command.

Any changes made by this command will not take effect until the **bosboot** and **shutdown -r** commands have been run (except **-l** and **-f** options).

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

bs Command

Purpose

Compiles and interprets modest-sized programs.

Syntax

bs [*File* [*Arguments*]]

Description

The **bs** command is a compiler and interpreter for interactive program development and debugging. To simplify program testing, it minimizes formal data declaration and file manipulation, allows line-at-a-time debugging, and provides trace and dump facilities and run-time error messages.

The optional parameter *File* specifies a file of program statements that you create and that the compiler reads before it reads from standard input. Statements entered from standard input are normally executed immediately (see **compile** and **execute** statement syntax). By default, statements read from *File* are compiled for later execution.

Unless the final operator is assignment to a variable, the result of an immediate expression statement is displayed.

Additional command line *Arguments* can be passed to the program using the built-in functions **arg** and **narg**.

Program lines must conform to one of the following formats:

```
statement
label statement
```

The interpreter accepts labeled statements only when it is compiling statements. A *label* is a name immediately followed by a colon. A label and a variable can have the same name. If the last character of a line is a **** (backslash), the statement continues on the following physical line.

A statement consists of either an expression or a keyword followed by zero or more expressions.

> **Note:** To avoid unpredictable results when using a range expression in the international environment, use a character class expression rather than a standard range expression.

Statement Syntax

Item	Description
break	Exits the innermost **for** or **while** loop.
clear	Clears the symbol table and removes compiled statements from memory. A **clear** is always executed immediately.
compile [*Expression*]	Causes succeeding statements to be compiled (overrides the immediate execution default). The optional *Expression* is evaluated and used as a file name for further input. In this latter case, the symbol table and memory are cleared first. **compile** is always executed immediately.
continue	Transfers control to the loop-continuation test of the current **for** or **while** loop.
dump [*Name*]	Displays the name and current value of every global variable or, optionally, of the *Named* variable. After an error or interrupt, **dump** displays the number of the last statement and (possibly) the user-function trace.
exit [*Expression*]	Returns to the system level. The *Expression* is returned as process status.
execute	Changes to immediate execution mode (pressing the INTERRUPT key has the same effect). This statement does not cause stored statements to execute (see **run**).

Item	Description
for	Performs repeatedly, under the control of a named variable, a statement or a group of statements using one of the following syntaxes: `for name=Expression Expression statement` `next` OR `for name=Expression Expression` `statement . . .` `next` OR `for Expression, Expression, Expression statement` `next` OR `for Expression, Expression, Expression` `statement . . .` `next` The first format specifies a single statement where the variable takes on the value of the first expression and then is increased by one on each loop until it exceeds the value of the second expression. You can use the second format to do the same thing , but you can specify a group of statements. The third format requires an initialization expression followed by a test expression (such as true to continue) and a loop-continuation action expression. You can use the fourth format to do the same thing, but you can specify a group of statements. Use commas to separate the expressions in the third and fourth formats.
fun	Defines a user-written function using the following syntax: `fun f ([a, . . . ]) [v, . . . ]` `statement . . .` `nuf` *f* specifies the function name, *a* specifies any parameters, and *v* identifies any local variables for the user-written function. You can specify up to 10 parameters and local variables; however, they cannot be arrays or associated with I/O functions. You cannot nest function definitions.
freturn	Signals the failure of a user-written function. Without interrogation, **freturn** returns zero. (See the unary interrogation operator (**?**).) With interrogation, **freturn** transfers to the interrogated expression, possibly bypassing intermediate function returns.
goto *Name*	Passes control to the compiled statement with the matching label of *Name*.
ibase *n*	Sets the input base to *n*. The only supported values for *n* are 8**,** 10 (the default), and 16**.** Hexadecimal values 10-15 are entered as alphabetic characters a-**f.** A leading digit is required when a hexadecimal number begins with an alphabetic character (for example, f0a must be entered as 0f0a). **ibase** is always executed immediately.

Item	Description
if	Performs a statement in one of the following syntaxes: `if Expression statement` `[else` `statement . . . ]` `fi` OR `if Expression` `statement . . .` `[else` `statement . . . ]` `fi` The first format specifies a single statement and the second format specifies a group of statements to continue using if the expression evaluates to nonzero. The strings 0 and "" (null) evaluate as zero. In the second format, an optional **else** allows a group of statements to be performed when the first group is not. The only statement permitted on the same line with an **else** is an **if**. You can put **fis** only on the same line as another **fi**. You can combine **else** and **if** into **elif**. You can close an **if** . . . **elif** . . . [**else** . . .] sequence with a single **fi**.
include *Expression*	Evaluates an *Expression* to the name of a file containing program statements. Such statements become part of the program being compiled. The **include** statements are always executed immediately. Do not nest **include** statements.
obase *n*	Sets the output base to *n*. The only supported values for *n* are 8, 10 (the default), and 16**.** Hexadecimal values 10 through15 are entered as alphabetic characters a-f**.** A leading digit is required when a hexadecimal number begins with an alphabetic character (that is, f0a must be entered as 0f0a). Like **ibase**, **obase** is always executed immediately.
onintr	Provides program control of interrupts using one of the following syntaxes: **onintr** *Label* OR **onintr** In the first format, control passes to the *Label* given, just as if a **goto** had been performed when **onintr** was executed. The effect of the **onintr** statement is cleared after each interrupt. In the second format, pressing INTERRUPT ends the **bs** program.
return [*Expression*]	Evaluates the *Expression* and passes the result back as the value of a function call. If you do not provide an expression, the function returns zero.
run	Passes control to the first compiled statement. The random number generator is reset. If a file contains a **run** statement, it should be the last statement; **run** is always executed immediately.
stop	Stops execution of compiled statements and returns to immediate mode.

Item	Description
trace [*Expression*]	Controls function tracing. If you do not provide an *Expression* or if it evaluates to zero, tracing is turned off. Otherwise, a record of user-function calls/returns will be written. Each return decreases by one the **trace** expression value.
while	Performs repeatedly, under the control of a named variable, a statement or a group of statements using one of the following syntaxes: `while Expression statement` `next` OR `while Expression` `statement . . .` `next` The **while** statement is similar to the **for** statement except that only the conditional expression for loop continuation is given.
!*cmd*	Runs a command and then returns control to the **bs** program.
# *Comment*	Inserts a comment line.

Expression Syntax

Item	Description
Name	Specifies a variable or, when followed immediately by a colon, a label. Names are composed of a letter (uppercase or lowercase) optionally followed by letters and digits. Only the first six characters of a name are significant. Except for names declared locally in **fun** statements, all names are global. Names can take on numeric (double float) values or string values or be associated with input/output (see the built-in function **open**).
*Name***(**[*Expression*[, *Expression*] . . .]**)**	Calls function *Name* and passes to it the parameters in parentheses. Except for built-in functions, *Name* must be defined in a **fun** statement. Function parameters are passed by value.
*Name***[***Expression*[, *Expression*] . . . **]**	References either arrays or tables (see built-in function table). For arrays, each expression is truncated to an integer and used as a specifier for the name. The resulting array reference is syntactically identical to a name; a [1,2] is the same as a [1] [2]. The truncated expressions must be values between 0 and 32,767.
Number	Represents a constant numerical value. This number can be expressed in integer, decimal, or scientific notation (it can contain digits, an optional decimal point, and an optional **e** followed by a possibly signed exponent).
String	Represents a character string delimited by " " (double quotation marks). Within the string, you can use the \ (backslash) as an escape character that allows the double quotation mark (\"), new-line character (\n), carriage return(\r), backspace (\b), and tab (\t) characters to appear in a string. When not immediately followed by these special characters, \ stands for itself.
(*Expression*)	Alters the normal order of evaluation.

Item	Description
(*Expression***,** *Expression*[, *Expression*] . . . **) [***Expression***]**	Specifies to use the bracketed expression outside the parentheses as a subscript to the list of expressions within the parentheses. List elements are numbered from the left, starting at zero. The following expression has the value of True if the comparison is true: `(False, True) [a == b]`
Expression Operator Expression	Converts the operands to numeric form before the operator is applied unless the operator is an assignment, concatenation, or relational operator.

Unary Operators

Item	Description
? *Expression*	Tests for the success of *Expression* rather than its value. This interrogation operator is useful for testing: • The end of file • Result of the **eval** built-in function • Return from user-written functions (see **freturn**) An interrogation trap (end of file, for example), causes an immediate transfer to the most recent interrogation, possibly skipping assignment statements or intervening function levels.
- *Expression*	Negates *Expression*.
++ *Name*	Increases by one the value of the variable (or array reference).
— *Name*	Decreases by one the value of the variable.
! *Expression*	Specifies the logical negation of *Expression*.

Note: Unary operators treat a null string as a zero.

Binary Operators (in increasing precedence)

Item	Description
=	Specifies the assignment operator. The left operand must be a name or array element. It acquires the value of the right operand. Assignment binds right to left; all other operators bind left to right.
_	Specifies the concatenation operator. (It is the underline character).
& |	Specifies logical AND, logical OR. The result of: *Expression & Expression* is 1 (true) only if both of its parameters are non-zero (true); it is 0 (false) if one or both of its parameters are 0 (false). The result of: *Expression* | *Expression* is 1 (true) if one or both of its expressions are non-zero (true); it is 0 (false) only if both of its expressions are 0 (false). Both operators treat a null string as a zero.

Item	Description
< <= > >= == !=	Specifies the relational operators: • < for less than • <= for less than or equal to • > for greater than • >= for greater than or equal to • == for equal to • != for not equal to The relational operators return 1 if the specified relation is True; otherwise they return 0 (false). Relational operators at the same level extend as follows: **a>b>c** is the same as **a>b& b>c**. A string comparison is made if both operands are strings. The comparison is based on the collating sequence specified in the environment variable **LC_COLLATE**.
+ -	Specifies addition and subtraction.
*** / %**	Specifies multiplication, division, and remainder.
^	Specifies exponentiation.

Note: Binary operators treat a null string as a zero.

Functions Dealing With Arguments

Item	Description
arg(*i*)	Returns the value of the *i*-th actual argument at the current function call level. At level zero, **arg** returns the *i*-th command-line argument. For example, **arg(0)** returns **bs**.
narg( )	Returns the number of arguments passed. At level zero, it returns the command line argument count.

Mathematical Functions

Item	Description
abs(*x*)	Returns the absolute value of *x*.
atan(*x*)	Returns the arc tangent of *x*.
ceil(*x*)	Returns the smallest integer not less than *x*.
cos(*x*)	Returns the cosine of *x*.
exp(*x*)	Returns e raised to the power *x*.
floor(*x*)	Returns the largest integer not greater than *x*.
log(*x*)	Returns the natural logarithm of *x*.
rand( )	Returns a uniformly distributed random number between zero and one.
sin(*x*)	Returns the sine of *x*.
sqrt(*x*)	Returns the square root of *x*.

String Functions

Item	Description
size(*s*)	Returns the size (length in characters) of *s*.
bsize(*s*)	Returns the size (length in bytes) of *s*.

Item	Description
format(*f*, *a*)	Returns the formatted value of *a*, *f* being a format specification string in the style of the **printf** subroutine. Use only the **%...f**,**%...e**, and **%...s** formats.
index(*x*, *y*)	Returns a number that is the first position in *x* containing a character that any of the characters in *y* matches. 0 return if no match is found. For 2-byte extended characters, the location of the first byte is returned.
trans(*s*, *f*, *t*)	Translates characters in the source string *s* which match characters in *f* into characters having the same position in *t*. Source characters that do not appear in *f* are copied unchanged into the translated string. If string *f* is longer than *t*, source characters that match characters found in the excess portion of *f* do not appear in the translated string.
substr(*s*, *Start*, *Length*)	Returns the substring of *s* defined by *Start* position in characters and *Length* in characters.

Item	Description
match(*String*, *Pattern*) **mstring**(*n*)	Returns the number of characters in *string* that match *pattern*. The characters **.**, *****, **$**, **[**, **]**, **^** (when inside square brackets), **** (and ****) have the following special meanings: **Note:** See **ed** for a more detailed discussion of this special notation. **.** Matches any character except the new-line character. ***** Matches zero or more occurrences of the pattern element that it follows. For example, .* matches zero or more occurrences of any character except the new-line character. **$** Specifies the end of the line. **[.-.]** Matches any one character in the specified range ([.-.]) or list ([. . .]), including the first and last characters. **[^ .-.]** **[^ . . .]** Matches any character except the new-line character and any remaining characters in a range or list. A circumflex (^) has this special meaning only when it immediately follows the left bracket. **[].-.]** **[] . . .]** Matches] or any character in the list. The right square bracket does not terminate such a list when it is the first character within it (after an initial ^ , if any). **\(. . . \)** Marks a substring and matches it exactly.The pattern must match from the beginning of the string and the longest possible string. Consider, for example: `match ('a123ab123',".*\([a-z]\)") = 6` In this instance, .* matches a 123a (the longest string that precedes a character in the range a-z); \ ([a-z]\) matches b, giving a total of six characters matched in the string. In an expression such as [a-z], the minus means "through," according to the current collating sequence. A collating sequence may define equivalence classes for use in character ranges. See the "International Character Support Overview" for more information on collating sequences and equivalence classes. The **mstring** function returns the *n*th substring in the last call to **match** (*n* must be between 1 and 10 inclusive).

File-Handling Functions

open*(Name, File, Mode)*

Item	Description
close*(Name)*	Specifies the name, file type and file mode. *Name* must be a legal variable name (passed as a string). After a close, the name becomes an ordinary variable. For open, the *File* can be one of the following: • 0 for standard input • 1 for standard output • 2 for error output • A string representing a file name • A string beginning with an !, which represents a command to be run (using "sh -c") *Mode* must be specified with an r for read, w for write, W for write without the new line character, or a for append. The initial associations are: • open ("get", 0, "r") • open ("put", 1, "w") • open ("puterr", 2, "w")
access*(p, m)*	Performs the access subroutine. Parameter *p* is the path name of a file; *m* is a bit pattern representing the requested mode of access. This function returns a 0 if the system request is permitted, -1 if it is denied.
ftype(*s)*	Returns a single character indicating file type: f for regular file, p for FIFO (named pipe), d for directory, b for block special, or c for character special.

Table Functions

Item	Description
table*(Name, Size)*	Specifies an associatively accessed, one-dimensional array. "Subscripts" (called keys) are strings (numbers are converted). *Name* must be a **bs** variable name (passed as a string). *Size* sets the minimum number of elements to be allocated. On table overflow, **bs** writes an error message.
item*(Name, i)*	
key( )	Accesses table elements sequentially instead of in an orderly progression of key values. Where the item function accesses values, the key function accesses the "subscript" of the previous item call. Do not quote *Name*. Since exact table sizes are not defined, the interrogation operator should be used to detect end-of-table; for example: `table("t",100)` `.` `.` `.` `#If word contains "parity", the following expression` `#adds one to the count of that word:` `++t[word]` `.` `.` `.` `#To display the key/value pairs:` `for i = 0, ? (s = item (t, i)), ++i if key( ) put = key ( )_":"_s`
iskey*(Name, Word)*	Tests whether the key word exists in the table name and returns one for true, zero for false.

Miscellaneous Functions

Item	Description
eval(*string*)	Specifies to evaluate the string parameter as an expression. The function is handy for converting numeric strings to numbers. **eval** can also be used as a crude form of indirection, as in: `name = "x,y,z"` `eval("++"_name)` which increments the variable "x,y,z". In addition, when **eval** is preceded by ? (interrogation operator), you can control **bs** error conditions. For example: `?eval ("open(\"X\",\"XXX\", \"r\")")` returns the value zero if there is no file named "XXX" (instead of halting your program). The following performs a **goto** to the label "L:" (if it exists): `label = "L:"` `if! (?eval ("goto"_label))puterr="no label"`

Item	Description
plot(*request, args)*	Produces output on devices recognized by the **tplot** command. Some requests do not apply to all plotters. All requests except 0 and 12 are implemented by piping characters to **tplot**. The call requests are as follows: **plot(0, *term*)** Causes further plot output to be piped into **tplot** with a flag of **-T***term.* **plot(1)** Erases the plotter. **plot(2, *string*)** Labels the current point with *string* **plot(3, *x1, y1, x2, y2*)** Draws the line between *(x1, y1*) and (*x2, y2*). **plot(4, *x, y, r*)** Draws a circle with center(*x, y*) and radius *r*. **plot(5, x*1, y1, x2, y2, x3, y3*)** Draws an arc (counterclockwise) with center *(x1, y1)*, and end points *(x2,y2)* and *(x3, y3).* **plot(6)** Not implemented. **plot(7, *x, y*)** Makes the current point at *(x, y).* **plot(8, *x*, y)** Draws a line from the current point to *(x ,y).* **plot(9, *x, y*)** Draws a point at *(x, y).* **plot(10, string)** Sets the line mode to string **plot(11, *x1, y1, x2, y2*)** Makes *(x1, y1)* the lower left corner of the plotting area and (x2, y2) the upper right corner of the plotting area. **plot(12, *x1, y1, x2, y2*)** Causes subsequent *x(y*) coordinates to be multiplied by *x1 (y1)* and then added to *x2 (y2)* before they are plotted. The initial scaling is **plot(12, 1.0, 1.0, 0.0, 0.0).**
last ( )	Returns, in immediate mode, the most recently computed value.

Example

To execute the bs command and direct the result to a file called output, enter:

```
bs < input.n > output
```

OR

```
bs input.n > output
```

bsh Command

Purpose

The **bsh** command invokes the Bourne shell.

Syntax

bsh [**-i**] [**-r**] [{ **+** | **-** } { [**a**] [**e**] [**f**] [**h**] [**k**] [**n**] [**t**] [**u**] [**v**] [**x**] }]
[**-c** *String* | **-s** | *File* [*Parameter*]]

Note: Preceding a flag with a **+** (plus sign) rather than a **-** (minus sign) turns it off.

Description

The **bsh** command invokes the Bourne shell, an interactive command interpreter and command-programming language. The shell carries out commands either interactively from a terminal keyboard or from a file.

Flags

The Bourne shell interprets the following flags only when the shell is invoked at the command line.

Note: Unless you specify either the **-c** or **-s** flag, the shell assumes that the next parameter is a command file (shell script). It passes anything else on the command line to that command file.

Item	Description
-a	Marks for export all variables to which an assignment is performed. If the assignment precedes a command name, the export attribute is effective only for that command's execution environment, except when the assignment precedes one of the special built-in commands. In this case, the export attribute persists after the built-in command has completed. If the assignment does not precede a command name, or if the assignment is a result of the operation of the **getopts** or **read** command, the export attribute persists until the variable is unset.
-c *String*	Runs commands read from the *String* variable. Sets the value of special parameter `0` from the value of the *String* variable and the positional parameters ($1, $2, and so on) in sequence from the remaining *Parameter* operands. The shell does not read additional commands from standard input when you specify this flag.
-e	Exits immediately if all of the following conditions exist for a command: • It exits with a return value greater than 0. • It is not part of the compound list of a **while**, **until**, or **if** command. • It is not being tested using AND or OR lists. • It is not a pipeline preceded by the ! (exclamation point) reserved word.
-f	Disables file name substitution.
-h	Locates and remembers the commands called within functions as the functions are defined. (Normally these commands are located when the function is executed; see the **hash**command.)
-i	Makes the shell interactive, even if input and output are not from a workstation. In this case the shell ignores the **TERMINATE** signal, so that the **kill 0** command does not stop an interactive shell, and traps an **INTERRUPT** signal, so you can interrupt the function of the **wait** command. In all cases, the shell ignores the **QUIT** signal.
-k	Places all keyword parameters in the environment for a command, not just those preceding the command name.

Item	Description
-n	Reads commands but does not execute them. The **-n** flag can be used to check for shell-script syntax errors. An interactive shell may ignore this option.
-r	Invokes the restricted shell. Using this flag is the same as issuing the **Rsh** command, except the shell enforces restrictions when reading the **.profile** files.
-s	Reads commands from standard input. Any remaining parameters specified are passed as positional parameters to the new shell. Shell output is written to standard error, except for the output of built-in commands.
-t	Exits after reading and executing one command.
-u	Treats an unset variable as an error and immediately exits when performing variable substitution. An interactive shell does not exit.
-v	Displays shell input lines as they are read.
-x	Displays commands and their arguments before they are executed.

Note: Using a + (plus sign) rather than a - (minus sign) unsets flags. The $- special variable contains the current set of flags.

Files

Item	Description
/usr/bin/bsh	Specifies the path name to the Bourne shell.
/usr/bin/Rsh	Specifies the path name to the restricted Bourne shell, a subset of the Bourne shell.
/tmp/sh*	Contains temporary files that are created when a shell is opened.

bterm command

Purpose

Emulates terminals in bidirectional (BIDI) mode.

Syntax

bterm [**-maps** *Map*] [**-help**] [**-keywords**] [**-nobidi**] [**-symmetric**] [**-autopush**] [**-or** *Orientation*] [**-text** *TextType*] [**-nss** *NumShape*] [**-csd** *CharShape*] [**-tail**] [**-nonulls**]

Description

The **bterm** command emulates the IBM 3151, VT220, HFT and other terminals. It operates in BIDI mode on ASCII terminals. This command creates a BIDI shell that can run any BIDI application. You cannot initiate the **bterm** command recursively from within itself.

The maps that determine the keyboard mapping and the symmetric swapping of characters are specified by the **-maps** flag. You can specify other BIDI behaviors using the flags available to the **bterm** command or by setting them in the defaults files. Such behaviors include the default text mode, the default screen orientation, the default mode of Arabic character shaping, the default shape of numerals, whether the Symmetric Swapping mode is enabled and whether the Autopush mode is enabled or not. The behaviors specified with flags take precedence over the behaviors set in the defaults files.

The default files are searched in the following order:

1. The **.Bidi-defaults** file is searched for in your home directory.

2. If the file is not found, the **bterm** command searches for the **BTerm** resource file in the **/usr/lib/nls/bidi/$LANG/app-defaults** file.

Flags

Item	Description
-autopush	Enables the Autopush mode in visual text mode.
-csd *CharShape*	Specifies the shape of Arabic characters. The *CharShape* variable can be one of the following options: • **automatic** • **isolated** (visual text mode only) • **initial** (visual text mode only) • **middle** (visual text mode only) • **final** (visual text mode only) • **passthru** The default is automatic shaping.
-help	Lists the available parameters and their syntax.
-keywords	Lists the keywords available in defaults file.
-maps *Map*	Specifies the map used for keyboard mapping and symmetric swapping of characters. Each language has a different map, and the available options for the *Map* variable are in the **/usr/lib/nls/bidi/maps** directory. You must specify the environment variable **BIDIPATH** as follows: `export BIDIPATH=/usr/lib/nls/bidi`
-nobidi	Disables the BIDI mode.
-nonulls	Initializes the screen with spaces instead of nulls.
-nss *NumShape*	Specifies the shape of the numerals. Specify one of the following options for the *NumShape* variable: • **bilingual** • **hindi** • **arabic** • **passthru** The default is **bilingual**.
-or *Orientation*	Specifies screen orientation. The *Orientation* variable can be either **LTR** or **RTL**. The default is **LTR**.
-symmetric	Enables the Symmetric Swapping mode.
-tail	Writes the "seen," "sheen," "sad," and "dad" characters of the Arabic language in two cells instead of one cell.
-text *TextType*	Specifies the type of data stream. The *TextType* variable can be either **implicit** or **visual**. The default is **implicit**.

Key Combinations

To change the BIDI settings using key combinations, press the Ctrl+X key sequence to enter a BIDI command mode. Any key you type after this key sequence is interpreted as a BIDI command. Invalid keys sound a beep and exit the BIDI command mode. The following keys are valid BIDI commands:

Key	Purpose
r	Reverses the screen orientation.
n	Sets the language keyboard layer to National.
l	Sets the language keyboard layer to Latin.
a	Toggles the automatic shaping variable option of the Arabic characters (valid also for Implicit mode).
t	Displays the status.
space	Enters a required space (RSP).

For implicit mode only:

Key	Purpose
c	Toggles the column heading mode.

For visual mode only:

Key	Purpose
s	Initiates the Push mode.
e	Terminates the End Push mode.
p	Toggles the Autopush mode.
f	Shapes Arabic characters in their final forms.
i	Shapes Arabic characters in their initial forms.
b	Shapes Arabic characters in the Passthru mode.
o	Shapes Arabic characters in their isolated forms.
m	Shapes Arabic characters in their middle forms.

.Bidi-defaults Keywords

Use the following keywords to set the defaults for the **bterm** command.

.Bidi-defaults Keywords	
Keywords	**Value/Effect**
fScrRev	**on** Screen reverse function key is enabled. **off** Screen reverse function key is disabled.
fRTL	**on** RTL keyboard layer function key is enabled. **off** RTL keyboard layer function key is disabled.

.Bidi-defaults Keywords *(continued)*	
Keywords	**Value/Effect**
fLTR	**on** LTR keyboard layer function key is enabled. **off** LTR keyboard layer function key is disabled.
fPush	**on** Push function key is enabled. **off** Push function key is disabled.
fEndPush	**on** End Push function key is enabled. **off** End Push function key is disabled.
fAutoPush	**on** AutoPush function key is enabled. **off** AutoPush function key is disabled.
fASD	**on** Automatic Shape Determination function key is enabled. **off** Automatic Shape Determination function key is disabled.
fShapeIS	**on** Isolated Shape function key is enabled. **off** Isolated Shape function key is disabled.
fShapeIN	**on** Initial Shape function key is enabled. **off** Initial Shape function key is disabled.
fShapeM	**on** Middle Shape function key is enabled. **off** Middle Shape function key is disabled.
fShapeF	**on** Final Shape function key is enabled. **off** Final Shape function key is disabled.
textType	**implicit** Type of data stream is set to Implicit. **visual** Type of data stream is set to Visual.

.Bidi-defaults Keywords *(continued)*	
Keywords	**Value/Effect**
orientation	**LTR** Left-to-right default screen orientation. **RTL** Right-to-left default screen orientation.
symmetric	**on** Symmetric Swapping enabled. **off** Symmetric Swapping disabled.
numShape	**bilingual** Numeral shaping is set to bilingual. **hindi** Numerals are represented in Hindi. **arabic** Numeral shaping is set in Arabic/Hebrew. **passthru** Numerals are represented in passthru.
charShape	**automatic** Arabic characters are shaped automatically. **passthru** Arabic characters are displayed in passthru mode. **isolated** Arabic characters are displayed in isolated mode. **initial** Arabic characters are displayed in initial mode. **final** Arabic characters are displayed in final mode. **middle** Arabic characters are displayed in middle mode.
maps	Specifies the page code directory to be used for Keyboard. layering, input, output and symmetric character swapping.
expandTail	**on** Writes "seen"-like characters and their tails on two cells. **off** Writes "seen"-like characters and their tails on one cell.
nobidi	**on** Activates BIDI mode. **off** Turn off BIDI mode.
noNulls	**on** Replaces nulls by spaces. **off** Leaves nulls as null, no replacement of spaces.

bugfiler Command

Purpose

Automatically stores bug reports in specified mail directories.

Syntax

bugfiler [**-d**] [**-m** *MessageMode*] [**-b** *BugUserName*] [*MailDirectory*]

Description

The **bugfiler** command automatically intercepts bug reports, summarizes them, and stores them in the appropriate folders in the directory specified by the *MailDirectory* variable.

The mail delivery program starts the **bugfiler** command through a line in the **/etc/aliases** file. The line has the following format:

```
bugs:"|/usr/lib/bugfiler $HOME/bugstuff"
```

In the example, the bug reports are placed in the **$HOME/bugstuff** directory. If no directory is specified, the **bugfiler** command places the bug reports in the **$HOME/mail** default directory.

> **Note:** The **$HOME**/**mail** directory must be created for the **bugfiler** command to use as a default directory.

If the *BugUserName* is other than `bugs`, the entry in the **/etc/aliases** file should contain a **-b** *BugUserName* flag, as in the following example:

```
hadley:"|/usr/lib/bugfiler -b hadley"
```

In this example, `hadley` is declared the *BugUserName* and all bug reports are placed in the **/home/hadley/mail** default directory. All directories used by the **bugfiler** command must be owned by `hadley`.

The **bugfiler** command reads bug reports from standard input, checks the format of each report, then either sends a message acknowledging receipt ($HOME/*MailDirectory*/**.ack** file) or indicates improper format ($HOME/*MailDirectory*/**.format** file).

Improperly formatted bug reports are filed in the **errors** directory, which the **bugfiler** command creates as a subdirectory of the *MailDirectory* variable. Bug reports must be in the format found in the **/usr/lib/bugformat** file. Use the **sendbug** command to start the **/usr/lib/bugformat** file. The **bugfiler** command summarizes valid bug reports and files them in the folder specified in the `Index:` line of the report. The source directory name in the `Index:` line must match one of the directory names in the mail directory. The **bugfiler** command appends a line in the following format to the *MailDirectory*/**summary** file:

```
DirectoryName/MessageNumber IndexInformation SubjectInformation
```

> **Note:** The **bugfiler** command does not recognize forwarded mail. It notifies the forwarder, not the sender, unless a `Reply-To:` line is included in the header of the report.

Format of Bug Reports

Bug reports must be submitted in ARPA RFC 822 format. The **sendbug** command contains information to compose and mail bug reports in the correct format.

The reports require the following header lines for proper indexing:

Item	Description
`Date:`	Followed by the date the **bugfiler** command receives the report.
`From:`	Followed by the valid return address of the sender.

Item	Description
`Subject:`	Followed by a short summary of the problem.
`Index:`	Followed by the path of the source directory and source file, the version number, and optionally, the **Fix** keyword.

The body of the bug report requires the following lines:

Item	Description
`Description:`	Followed by a detailed description of the problem, suggestion, or complaint.
`Repeat-By:`	Followed by a procedure to repeat the problem.
`Fix:`	Followed by a **diff** command comparing the old and new source files or a description of how to solve the problem. Include the `Fix:` line only if the **Fix** keyword is specified in the `Index:` line.

Redistribution of Bug Reports

Bug reports can be redistributed according to index information in the *MailDirectory*/**.redist** file. The *MailDirectory*/**.redist** file is examined for a line beginning with an index name followed by a tab. Following the index name and tab is a comma-separated list of mail addresses to receive copies of bug reports. If the list continues on multiple lines, each line but the last must end with a \ (backslash). The following is an example of index information in the **.redist** file:

```
myindex   joe@hal,mary@mercutio,martha@banquo,sarah@mephisto,\
dee@hamlet,dewayne@ceasar
```

Flags

Item	Description
-b *BugUserName*	Specifies a new user ID. If the **-b** *BugUserName* flag is not specified, the **bugfiler** command defaults to the login name.
-d	Sets debugging on. When the **-d** flag is specified, the **bugfiler** command sends error messages to standard output.
-m *MessageMode*	Sets message protection. The **-m** *MessageMode* flag specifies file permissions, using hexadecimal format, for all files that the **bugfiler** command creates.

Examples

1. The syntax of the **bugfiler** command when used with all three flags and a specified *MailDirectory* variable is as follows:

   ```
   hadley:"|/usr/lib/bugfiler -d -m 755 -b hadley
   /home/hadley/bugdir"
   ```

   When placed in the **/etc/aliases** file, this line starts debugging, sets file permissions to `rwxr-xr-x`, declares `hadley` as the *BugUserName*, and specifies the **/home/hadley/bugdir** directory.

2. The following is an example of a bug report:

   ```
   Date: Mon, 27 Nov 89 11:26:15 -600
   From: a@B
   Subject: Read not setting errno correctly
   Index: LFS/rdwr.c workstation 3.1

   Description: Read not setting errno correctly

   Repeat-By: Start an NFS daemon and it receives errors. Errno is
   zero.
   ```

Files

Item	Description
/etc/ aliases	Contains system-wide aliases for the mail transport system.
usr/sbin/sendmail	Contains the mail delivery program.
MailDirectory/**summary**	Contains the bug report summaries.
BugUserName/*MailDirectory*/**.ack**	Contains the message sent in acknowledgment.
BugUserName/*MailDirectory*/**.format**	Contains the message sent when format errors are detected.
MailDirectory/**.redist**	Contains the redistribution list for bug reports.

burst Command

Purpose

Divides a message into separate, new messages.

Syntax

burst [**+***Folder*] [*Messages*] [**-inplace**] [**-noinplace**] [**-quiet**] [**-noquiet**] [**-verbose**] [**-noverbose**]

Description

The **burst** command allows you to divide a message into multiple, new messages. The **burst** command operates on digests, messages forwarded by the **forw** command, and blind carbon copies sent by the **forw** and **send** commands. Messages created using the **burst** command are numbered consecutively, beginning with the next highest number in the specified folder.

The **burst** command can create about 1000 messages from a single message. However, the **burst** command generally does not place a specific limit on the number of messages in a folder after bursting is complete.

The **burst** command uses encapsulation boundaries to determine where to separate the encapsulated messages. If an encapsulation boundary is located within a message, the **burst** command may split that message into two or more messages.

By default, the first message extracted from the first digest becomes the current message. If the **-inplace** flag is specified, the first new message becomes the current message.

Flags

Item	Description
+*Folder*	Specifies the folder containing the message to divide. By default, the system uses the current folder.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.

Item	Description
-inplace	Replaces each digest with a table of contents for the digest, places the messages contained in each digest directly after the digest's table of contents, and renumbers all subsequent messages in the folder to make room for the messages from the divided digest. **Attention:** The **burst** command does not place text displayed after the last encapsulated message in a separate message. When you specify the **-inplace** flag, the **burst** command loses this trailing text. In digests, this text is usually an End-of-Digest string. However, if the sender appended remarks after the last encapsulated message, the **burst** command loses these remarks.
Messages	Specifies the messages that you want to divide. This parameter may specify several messages, a range of messages, or a single message. Use the following references to specify messages: ***Number*** Number of the message. When specifying several messages, separate each number with a comma. When specifying a range, separate the first and last number in the range with a hyphen. ***Sequence*** A group of messages specified by the user. Recognized values include: **all** All messages in the folder. **cur or . (period)** Current message. This is the default. **first** First message in a folder. **last** Last message in a folder. **next** Message following the current message. **prev** Message preceding the current message.
-noinplace	Preserves each digest. This is the default.
-noquiet	Reports information about messages not in digest format. This flag is the default.
-noverbose	Prevents reporting of the actions the **burst** command performs while dividing the digests. This flag is the default.
-quiet	Prevents reporting of information about messages not in digest format.
-verbose	Reports the actions the **burst** command performs while dividing a digest.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Current-Folder:`	Sets the default current folder.
`Msg-Protect:`	Sets the protection level for your new message files.
`Path:`	Specifies a user's MH directory.

Examples

1. The user receives message 5 from `mickey@mouse` containing several messages in digest form:

   ```
   5+ 03/02 mickey@mouse
   6+ 03/02 disney@world
   ```

   To burst message 5 into several, separate messages, enter:

   ```
   burst 5
   5+ 03/02 mickey@mouse
   6 03/02 disney@world
   7 first message in digest
   8 second message in digest
   9 third message in digest
   ```

   The resulting new messages are appended to the end of the folder. Message 5 remains intact and still contains all four messages.
2. To burst message 5 using the **-inplace** flag, enter:

   ```
   burst 5 -inplace
   5+ 03/02 mickey@mouse
   6 first message in digest
   7 second message in digest
   8 third message in digest
   9 03/02 disney@world
   ```

   The resulting new messages are placed immediately after the digest, and the **burst** command renumbers all the messages that follow. Message 5 now contains only the header and text of the forwarded message.

Files

Item	Description
$HOME/.mh_profile	Contains the MH user profile.
/usr/bin/burst	Contains the executable form of the **burst** command.

c

The following AIX commands begin with the with the letter *c*.

caccelstat Command

Purpose

Reports statistics related to coherent accelerators for the entire system, or for each accelerator and process.

Syntax

System-wide aggregate statistics

```
caccelstat -a [-t count] [-i interval]
```

Context statistics

```
caccelstat -c [-p pid -d device] [-t count] [-i interval] [-@ wparname]
```

Aggregate device or Accelerator Function Unit (AFU) statistics

```
caccelstat -d [device] [-t count] [-i interval] [-@ wparname]
```

Aggregate process context statistics

```
caccelstat -p [pid] [-t count] [-i interval] [-@ wparname]
```

Aggregate device kernel context statistics

```
caccelstat -k [device] [-t count] [-i interval] [-@ wparname]
```

Description

The **caccelstat** command is used to monitor coherent accelerator activity in the system. It can report aggregate statistics for all Coherent Accelerator Processor Interface (CAPI)-enabled processes, or it can show statistics for each CAPI-enabled process. It can also show statistics for each Accelerator Function Unit (AFU), where AFU is a coherent accelerator device such as `/dev/cacce10`.

System-wide aggregate statistics

The system-wide aggregate statistics report contains the following information:

- Number of CAPI-enabled processes
- Number of accelerators in the system
- Number of user contexts
- Number of kernel contexts
- Number of master contexts
- Number of page faults
- Number of segment table (STAB) faults
- Number of AFU errors
- Number of AFU exceptions
- Number of AFU signals

- Number of AFU stalls

Context statistics

This report contains context statistics, one line per context. If the process ID (pid) and the device are not specified, all registered user contexts in the system are reported. Kernel contexts are omitted. If either the pid or the device is specified, contexts for that pid or device is reported. If both pid and device are specified, only the contexts pertaining to selected pid and device are reported.

Report contains following information:

State
Context state.

This field can have the following values:

S
The context is in suspended state.

R
The context is in running state.

D
The context is in detaching state.

A
The context is in attaching state.

E
The context is in error state.

For a master process, `M` is combined with the state. For example, if the context of the master process is in the running state, the state field displays `MR`.

Note: The running state (R) does not imply that the AFU is running this context. It means that it is not in any of the other states.

pid
Process ID. This column is skipped if pid is specified in the option.

pf
Number of page faults.

spf
Number of STAB faults.

pi
Number of page-ins.

err
Number of process errors.

exc
Number of process exceptions.

sig
Number of process signals.

stalls
Number of AFU stalls due to exception.

aur
Accelerator Utilization Register (AUR) value of this process. If AUR is not supported, this field shows 0.

device
Device name. This column is skipped if device is specified in the option.

Aggregate device (AFU) statistics

This report contains aggregate statistics for an AFU across all process contexts. If the device is not specified, statistics for all AFUs in the system is reported. This report includes following columns, one line per device:

nctx
Number of registered process contexts.

pf
Number of page faults.

spf
Number of STAB faults.

pi
Number of page-ins.

err
Number of AFU errors.

exc
Number of AFU exceptions.

sig
Number of AFU signals.

stalls
Number of AFU stalls due to exception queues being full.

device
Device name. This column is skipped if the device is specified in the option.

Aggregate process context statistics

This report contains aggregate and per process context statistics for a CAPI-enabled process. The pid is the AIX process ID. If the pid is not specified, statistics for all the CAPI-enabled processes in the system is reported.

This report includes following column, one per context:

pid
AIX Process ID. This column is skipped if the device is specified in the option.

nctx
Number of registered process contexts.

pf
Number of page faults.

spf
Number of STAB faults.

pi
Number of page-ins.

err
Number of AFU errors.

exc
Number of AFU exceptions.

sig
Number of AFU signals.

stalls
Number of AFU stalls due to exception queues being full.

aur
Accelerator Utilization Register value for this process. If AUR is not supported, this field shows 0.

Aggregate device kernel context statistics

This report contains aggregate or per device kernel context statistics. If no device is specified, statistics for all AFUs are reported. This report includes following column, one per device:

pf
Number of page faults.

spf
Number of STAB faults.

pi
Number of page-ins.

err
Number of AFU errors.

exc
Number of AFU exceptions.

device
Device name. This column is skipped if the device is specified in the option.

Aggregate statistics for each active workload partition (WPAR) in the systems

This report contains aggregate statistics for each active WPAR present in the system. This report includes following columns, one line per WPAR.

wpar
WPAR name.

nctx
Number of contexts.

pf
Number of page faults.

spf
Number of STAB faults.

pi
Number of page-ins.

err
Number of AFU errors.

exc
Number of AFU exceptions.

sig
Number of AFU signals.

stalls
Number of AFU stalls.

Flags

Flag	Description
-a	Reports system-wide aggregate statistics.
-c	Reports per context statistics.
-d	Reports aggregate AFU statistics.
-p	Reports aggregate process statistics.
-k	Reports aggregate device kernel statistics.
-t *count*	Specifies the number of times the statistics must be reported.
-i *interval*	Specifies the time interval in seconds after which the statistics must be reported.
-@ ALL	Reports aggregate statistics for each active WPAR that is present in the system.
-@ *wparname*	Reports aggregate statistics for the specified WPAR.

Examples

1. To report system-wide aggregate statistics, enter the following command:

```
caccelstat -a
```

2. To report system-wide aggregate statistics for 10 times at 1-second interval, enter the following command:

```
caccelstat -a -t 10 -i 1
```

3. To report all the context statistics for the CAPI-enabled processes in the system, enter the following command:

```
caccelstat -c
```

4. To report context statistics for the process 1234, enter the following command:

```
caccelstat -c -p 1234
```

5. To report context statistics for the device `/dev/memcopy0`, enter the following command:

```
caccelstat -c -d /dev/memcopy0
```

6. To report process statistics for all the CAPI-enabled processes in the system, enter the following command:

```
caccelstat -p
```

7. To report process statistics for the CAPI-enabled process 1234, enter the following command:

```
caccelstat -p 1234
```

8. To report device statistics for all the CAPI devices present in the system, enter the following command:

```
caccelstat -d
```

9. To report device statistics for the CAPI device `/dev/memcopy0`, enter the following command:

```
caccelstat -d /dev/memcopy0
```

10. To report device kernel statistics for all CAPI devices in the system, enter the following command:

```
caccelstat -k
```

11. To report device kernel statistics for CAPI device `/dev/memcopy0`, enter the following command:

```
caccelstat -k -d /dev/memcopy0
```

12. To report aggregate statistics for each active WPAR in the system, enter the following command:

```
caccelstat -@ ALL
```

13. To report process statistics for all the CAPI-enabled processes in the `testWpar` WPAR, enter the following command:

```
caccelstat -p -@ testWpar
```

14. To report device statistics for the CAPI device `/dev/memcopy0` in the `testWpar` WPAR, enter the following command:

```
caccelstat -d -@ testWpar
```

15. To report all the context statistics for the CAPI-enabled processes in the `testWpar`, enter the following command:

```
caccelstat -c -@ testWpar
```

cache_mgt Command

Purpose

Manages the infrastructure that provides caching on the solid-state drive (SSD) devices.

Syntax

```
cache_mgt object action [-I [level]] [-T [timeout]]
```

Cache device management command

```
cache_mgt device list [-l]
```

Cache pool management commands

```
cache_mgt pool list [-l]
cache_mgt pool create -d devName[,devName,...][-p poolName] [-f]
cache_mgt pool remove [-p poolName] [-f]
cache_mgt pool extend [-p poolName] -d devName[,devName,...] [-f]
```

Cache partition management commands

```
cache_mgt partition list [-l]
cache_mgt partition create [-p poolName] -s partitionSize [-P partitionName]
cache_mgt partition remove [-P partitionName] [-f]
cache_mgt partition extend [-P partitionName] -s partitionSize
cache_mgt partition assign [-P partitionName] -t targetDevName
cache_mgt partition unassign {-t targetDevName | [-P partitionName]} [-f]
```

Commands to manage caching on target devices

```
cache_mgt cache list
cache_mgt cache start {-t targetDevName -P partitionName | -t {targetDevName | all} | -f}
cache_mgt cache stop {-t {targetDevName | all} | -p {poolName | all}}
```

Statistics monitoring commands

```
cache_mgt monitor start
cache_mgt monitor stop
cache_mgt monitor get {-h -s | -h | -s}
```

Commands to manage cache engine

```
cache_mgt engine list [-l]
cache_mgt engine register -n cePath
cache_mgt engine unregister [-n cePath]
```

Description

The **cache_mgt** command is used to manage caching on SSD devices. This command provides the following functions:

- List available SSD devices that can be used to create or extend cache pools on the system.
- Create, remove, extend, and list the cache pools on the system. A cache pool is a group of SSD devices. Cache partitions are created from the cache pool.

- Create, remove, extend, assign, unassign, and list the cache partitions on the system. A cache partition is a part of the cache pool. A cache partition must be assigned to a target device that needs to be cached.
- Start and stop caching of a target device. It also lists the cache partitions along with the assigned target devices and its caching status. A cache engine must be registered before the caching is initiated.
- Monitor cache statistics.
- Register and unregister the cache engine. It also lists the registered cache engine information.

Caching modes

Caching can be performed in one of the following modes:

Physical mode
Cache devices (or SSD devices) are directly assigned to a logical partition (LPAR). The **cache_mgt** commands can be used to manage cache pool, cache partitions, and caching of a target device.

Virtual mode
Cache devices (or SSD devices) are assigned to the Virtual I/O Server. The cache pool and cache partitions are managed on the Virtual I/O Server. Cache partitions on Virtual I/O Server can be virtualized (virtual cache devices) to client LPAR through virtual SCSI. Cache partition assignment and caching must be managed on the LPAR.

Cache engine information

The cache engine module caches the target devices on to the cache partitions. A cache engine is included by default and it is automatically registered on the client LPARs.

Only a single cache pool is supported in the physical mode and caching can be started only on a single cache partition.

Flags

Cache device management commands

Object	Action	Flags and parameters	Description
device	list	[**-l**]	Lists the SSD devices. If you use the **-l** flag, the command prints the associated cache pool name.

Cache pool management commands

Object	Action	Flags and parameters	Description
pool	list	[**-l**]	Lists the cache pools. If you use the **-l** flag, the command also prints the associated SSD devices.
pool	create	**-d** *devName*[,*devName*, ...] [**-p** *poolName*] [**-f**]	Creates a cache pool with the list of SSD devices that are specified with the **-d** flag. The pool name can also be specified with the **-p** flag. If the force (**-f**) flag is specified, a cache pool is created irrespective of the previous use of the devices.
pool	remove	[**-p** *poolName*] [**-f**]	Removes the cache pool. This action fails if a partition still exists in the pool. If the force (**-f**) flag is specified, all existing partitions within the pool are removed.
pool	extend	[**-p** *poolName*] **-d** *devName*[,*devName*, ...] [**-f**]	Extends an existing pool with the list of SSD devices that are specified with the **-d** flag. If the force (**-f**) flag is specified, a cache pool is extended irrespective of the previous use of the devices.

Cache partition management commands

Object	Action	Flags and parameters	Description
partition	list	[**-l**]	Lists the cache partitions (virtual cache devices). If you use the **-l** flag, the command prints the associated pool name if a partition is a logical volume and the associated target device names, if configured, in a comma-separated format.
partition	create	[**-p** *poolName*] **-s** *partitionSize* [**-P** *partitionName*]	Creates a cache partition in a pool. The pool name can be specified with the **-p** flag. The partition name can be specified with the **-P** flag. The partition size must be specified with the **-s** flag. Size of the partition must be in one of the following units: • B/b 512 byte blocks • K/k KB • M/m MB • G/g GB
partition	remove	[**-P** *partitionName*] [**-f**]	Removes a partition from a cache pool. The partition name can be specified with the **-P** flag. **Note:** You must unassign the partition before removing it. Or, use the force (**-f**) flag to remove it.
partition	extend	[**-P** *partitionName*] **-s** *partitionSize*	Extends an existing partition by the *partitionSize* value that is specified with the **-s** flag. The *partitionName* value can also be specified with the **-P** flag.
partition	assign	[**-P** *partitionName*] **-t** *targetDevName*	Creates the relationship between a cache partition name that is specified with the **-P** flag and a target device name that is specified with the **-t** flag in the configuration. The caching state remains 0 (stopped) and caching operation is not started. You must perform the **cache start -t** action to start the caching operation and change the caching state in the configuration.
partition	unassign	**-t** *targetDevName* [**-f**]	Removes the relationship between a cache partition and the target device name that is specified with the **-t** flag in the configuration. You must stop the caching for the partition before removing it or use the force (**-f**) flag before removing it.
partition	unassign	[**-P** *partitionName*] [**-f**]	Removes the relationship between a cache partition and all its target devices in the configuration. You must stop the caching for the partition before removing it or use the force (**-f**) flag before removing it.

Commands to manage caching on target devices

Object	Action	Flags and parameters	Description
cache	list		Lists the partitions for which the caching operation is started.
cache	start	**-t** *targetDevName* **-P** *partitionName*	Starts the caching operation of a target device name that is specified with the **-t** flag on a cache partition name that is specified with the **-P** flag.
cache	start	**-t** {*targetDevName* | all}	Starts the caching operation of a target device name that is specified with the **-t** flag for a previously assigned cache partition. The command starts the caching operation for all assigned target devices if you specify the **all** option with the **-t** flag.
cache	start	**-f**	Loads the cache engine even if there is no cache device.
cache	stop	**-t** {*targetDevName* | all}	Stops the caching operation of a target device name that is specified with the **-t** flag. The command stops the caching operation for all assigned target devices when the **all** option is specified with the **-t** flag. The cache partition assignment definition is not removed from the configuration.
cache	stop	**-p** {*poolName* | all}	Stops the caching operation of all target devices of a cache pool name that is specified with the **-p** flag. The command stops caching for all assigned target devices when the **all** option is specified with the **-p** flag. The cache partition assignment definition is not removed from the configuration.

Statistics monitoring commands

Object	Action	Flags and parameters	Description
monitor	start		Starts monitoring the caching operation.
monitor	stop		Stops monitoring the caching operation.
monitor	get	{**-h -s** | **-h** | **-s**}	Gets the caching I/O statistics. The command displays the statistics if the **-s** flag is specified. The command displays the header if the **-h** flag is specified.

Commands to manage the cache engine

Object	Action	Flags and parameters	Description
engine	list	[**-l**]	Lists the cache engine path that is set in the configuration. If the -l flag is specified, additional information about the cache engine and its capabilities are listed.
engine	register	**-n** *cePath*	Registers the cache engine (*cePath*) that is specified with the **-n** flag.

Object	Action	Flags and parameters	Description
engine	unregister	[**-n** *cePath*]	Unregisters the cache engine. If the *cePath* value is not specified with the **-n** flag, deactivate the cache engine. You must not remove its definition from the configuration.

Examples

1. To create a cache pool from a list of cache devices, enter the following command:

   ```
   cache_mgt pool create -d hdisk1,hdisk2,hdisk3 -p cmpool0
   ```

   The output is displayed similar to the following example:

   ```
   Pool cmpool0 created with device hdisk1.
   ```

2. To list the cache pool, enter the following command:

   ```
   cache_mgt pool list -l
   ```

   The output is displayed similar to the following example:

   ```
   cmpool0,hdisk1
   ```

3. To create a cache partition in a pool that has the partition size of 80 MB, enter the following command:

   ```
   cache_mgt partition create -p cmpool0 -s 80M -P part1
   ```

   The output is displayed similar to the following example:

   ```
   Partition part1 created in pool cmpool0.
   ```

4. To list the cache partitions, enter the following command:

   ```
   cache_mgt partition list -l
   ```

   The output is displayed similar to the following example:

   ```
   part1,cmpool0
   ```

5. To assign a cache partition to a target device, enter the following command:

   ```
   cache_mgt partition assign -t hdisk2 -P part1
   ```

   The output is displayed similar to the following example:

   ```
   Partition part1 assigned to target hdisk2.
   ```

6. To start caching of a target device, enter the following command:

   ```
   cache_mgt cache start -t hdisk2
   ```

   The output is displayed similar to the following example:

   ```
   Cache for target hdisk2 has been started.
   ```

7. To list all target devices that are started or for which caching is assigned, enter the following command:

   ```
   cache_mgt cache list
   ```

   The output is displayed similar to the following example:

```
hdisk2,part1,active
```

8. To extend an existing cache pool, enter the following command:

   ```
   cache_mgt pool extend -p cmpool0 -d hdisk5 -f
   ```

   The output is displayed similar to the following example:

   ```
   Pool cmpool0 extended with device hdisk5.
   ```

9. To extend an existing cache partition by size 120 MB, enter the following command:

   ```
   cache_mgt partition extend -P part1 -s 120M
   ```

   The output is displayed similar to the following example:

   ```
   Partition part1 extended by size 120M.
   ```

10. To stop the caching operation of a target device, and then to list the partitions, enter the following commands:

    ```
    cache_mgt cache stop -t hdisk2
    cache_mgt cache list
    ```

    The output is displayed similar to the following example:

    ```
    Cache for target hdisk2 has been stopped.
    hdisk2,part1,inactive
    ```

11. To unassign the target device from a cache partition, enter the following command:

    ```
    cache_mgt partition unassign -t hdisk2
    ```

    The output is displayed similar to the following example:

    ```
    Partition part1 unassigned from target hdisk2.
    ```

12. To remove a cache partition from a cache pool, enter the following command:

    ```
    cache_mgt partition remove -P part1
    ```

    The output is displayed similar to the following example:

    ```
    Partition part1 removed.
    ```

13. To remove the cache pool, enter the following command:

    ```
    cache_mgt pool remove -p cmpool0
    ```

    The output is displayed similar to the following example:

    ```
    Pool cmpool0 removed.
    ```

cachefslog Command

Purpose

Controls the logging of a cache file system.

Syntax

cachefslog [**-f***LogFile* |**-h**] *Cachefs_Mount_Point*

Description

The **cachefslog** command displays where CacheFS statistics are being logged. Optionally, it sets where CacheFS statistics are being logged, or it halts logging for a cache specified by *Cachefs_Mount_Point*. The *Cachefs_Mount_Point* argument is a mount point of a cache file system. All file systems cached under the same cache as *Cachefs_Mount_Point* are logged.

Flags

Item	Description
-f *LogFile*	Specifies the log file to be used. **Note:** You must have root authority in order to use this flag.
-h	Halts logging. **Note:** You must have root authority in order to use this flag.

Exit Status

The following exit values are returned:

Item	Description
0	success
non-zero	an error has occurred.

Examples

1. To checks if the directory **/home/sam** is being logged, type:

   ```
   cachefslog /home/sam
   ```

   The system displays the following:

   ```
   not logged: /home/sam
   ```

2. To change the *logfile* of **/home/sam** to **/var/tmp/samlog**, type:

   ```
   cachefslog -f /var/tmp/samlog /home/sam
   ```

   The system displays the following:

   ```
   /var/tmp/samlog: /home/sam
   ```

3. To halt logging for the **/home/sam directory**, type:

   ```
   cachefslog -h /home/sam
   ```

   The system displays the following:

   ```
   not logged: /home/sam
   ```

Files

Item	Description
/usr/sbin/cachefslog	Contains the **cachefslog** command.

cachefsstat Command

Purpose

Displays information about a cache file system.

Syntax

cachefsstat [**-z**] [*path...*]

Description

The **cachefsstat** command displays statistical information about the cache file system mounted on *path*. The statistical information includes cache hits and misses, consistency checking, and modification operations. If *path* is not specified, all mounted cache file systems are used. **cachefsstat** can also be used to reinitialize this information (see **-z** flag).

The statistical information includes the following:

Item	Description
`hit rate`	The percentage of cache hits over the total number of attempts, followed by the actual numbers of hits and misses.
`consistency checks`	The number of consistency checks performed, followed by the number that passed, and the number that failed.
`modifies`	The number of modify operations, including, for example, writes and creates.

Flags

Item	Description
-z	Reinitializes, zeros, statistics. Execute **cachefsstat -z** before running **cachefsstat** again to gather statistics on the cache performance. This flag can only be use by the superuser. The statistics printed reflect those just before the statistics are reinitialized.

Exit Status

The following exit values are returned:

Item	Description
0	success
non-zero	an error has occurred.

Examples

1. To display the cache file system statistics of the **/home/sam** directory, type:

```
cachefsstat /home/sam
```

The system displays the following:

```
cache hit rate: 73% (1234 hits, 450 misses) consistency checks: 700 (650 pass, 50 fail)
modifies: 321
```

Files

Item	Description
/usr/sbin/cachefsstat	Contains the **cachefsstat** command.

cachefswssize Command

Purpose

Displays the work space size for a cache file system.

Syntax

cachefswssize *LogFile*

Description

The **cachefswssize** command displays the work space size determined from *LogFile*. This includes the amount of cache space needed for each filesystem that was mounted under the cache, as well as a total.

Exit Status

The following exit values are returned:

Item	Description
0	success
non-zero	an error has occurred.

Examples

1. To display the work space size of the cache filesystems being logged in the file **/var/tmp/samlog**, type:

```
cachefswssize /var/tmp/samlog
```

The system displays similar to the following:

```
/home/sam
                                end size:    10688k
                           high water size:    10704k

/foo

                                end size:      128k
                           high water size:      128k

/usr/dist

                                end size:     1472k
                           high water size:     1472k

total for cache

                              initial size:   110960k
                                  end size:    12288k
```

```
                              high water size:    12304k
```

Files

Item	Description
/usr/sbin/cachefswssize	Contains the **cachefswssize** command.

cal Command

Purpose

Displays a calendar.

Syntax

cal [[*Month*] *Year*]

Description

The **cal** command displays a calendar of the specified year or month.

The *Year* parameter names the year for which you want a calendar. Since the **cal** command can display a calendar for any year from 1 through 9999, you must enter the full year rather than just the last two digits. The *Month* parameter identifies the month for which you want the calendar. It can be a number from 1 (indicating January) to 12 (indicating December). If you specify neither the *Year* nor the *Month* parameter, the **cal** command displays the current month. If you specify only one parameter, the **cal** command assumes the parameter is the *Year* parameter and displays the calendar for the indicated year.

Note: The **cal** command does not accept standard input.

The **cal** command uses the appropriate month and day names according to the locale settings.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To display a calendar for February, 1994, at your workstation, enter:

```
cal 2 1994
```

2. To print a calendar for 1994, enter:

```
cal 1994 | qprt
```

3. To display a calendar for the year 84, enter:

```
cal 84
```

Files

Item	Description
/usr/bin/cal	Contains the **cal** command.

calendar Command

Purpose

Writes reminder messages to standard output.

Syntax

calendar [**-**]

Description

The **calendar** command reads the **calendar** file and displays any line in the file that contains today's or tomorrow's date. The **calendar** file is user-created and must be in the same directory from which you run the **calendar** command. Typically, the **calendar** file resides in your home directory.

If you run the **calendar** command on a Friday, the **calendar** command displays all lines containing the dates for that Friday as well as the subsequent Saturday, Sunday, and Monday. The command does not recognize holidays.

The **calendar** command recognizes date formats such as *Month Day*, *Abbreviation Date*, and *MonthNumeral***/***Date.* Examples of these formats include December 7, Dec. 7 and 12/7. The **calendar** command also recognizes the special character * (asterisk) when followed by a / (slash). It interprets */7, for example, as signifying the seventh day of every month. The **calendar** command does not recognize formats such as 7/*, 7 December, 7/12, * 7 or DEC. 7.

If the system administrator has created a **calendar** file for all users, you can access this file by placing the following line at the beginning of your local **calendar** file:

```
#include <FileName>
```

The actual value of the *FileName* variable is determined by the system administrator. The name of this file does not have to be **calendar**. When you run the **calendar** command, it displays reminders that were stored in your local **calendar** file as well as those stored in the file specified by the *FileName* variable.

> **Note:** When the **calendar** file contains include statements, the **calendar** command runs the **calendar** file through the C preprocessor. To use include statements with the **calendar** file, the C preprocessor, which is contained in the **/usr/ccs/lib/cpp** file, must be installed on the operating system.

For you to get reminder service, your **calendar** file must have read permission for others. See the **chmod** command for information on setting permissions.

Flag

Item	Description
-	Calls the **calendar** command for everyone having a **calendar** file in the home directory. The **calendar** command sends reminders using the **mail** command instead of writing the results to standard output.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. A typical **calendar** file might look like the following:

```
*/25 - Prepare monthly report
Aug. 12 - Fly to Denver
aug 23 - board meeting
Martha out of town - 8/23, 8/24, 8/25
8/24 - Mail car payment
sat aug/25 - beach trip
August 27 - Meet with Simmons
August 28 - Meet with Wilson
```

   To run the **calendar** command, enter:

```
calendar
```

   If today is Friday, August 24, then the **calendar** command displays the following:

```
*/25 - Prepare monthly report
Martha out of town - 8/23, 8/24, 8/25
8/24 - Mail car payment
sat aug/25 - beach trip
August 27 - Meet with Simmons
```

2. A **calendar** file that contains an include statement might look like the following:

```
#include </tmp/out>
1/21 -Annual review
1/21 -Weekly project meeting
1/22 *Meet with Harrison in Dallas*
Doctor's appointment - 1/23
1/23 -Vinh's wedding
```

   To run the **calendar** command, enter:

```
calendar
```

   If today is Wednesday, January 21, then the **calendar** command displays the following:

```
Jan.21 Goodbye party for David
Jan.22 Stockholder meeting in New York
1/21 -Annual review
1/21 -Weekly project meeting
1/22 *Meet with Harrison in Dallas*
```

   The results of the **calendar** command indicate the `/tmp/out` file contained the following lines:

```
Jan.21 Goodbye party for David
Jan.22 Stockholder meeting in New York
```

Files

Item	Description
$HOME/calendar	Contains the **calendar** command.

Item	Description
/usr/lib/calprog	Contains the program that determines dates.
/usr/ccs/lib/cpp	Contains the C preprocessor.
/etc/passwd	Contains basic user attributes.

cancel Command

Purpose (line printer requests)

Cancels requests to a line printer.

Syntax (line printer requests)

cancel { *JobID ...* | *PrinterName* }

or

cancel *JobID QueueName*

Description (line printer requests)

The **cancel** command cancels line printer requests that were made by the **lp** command.

Specifying the following cancels the local print jobs:

- *JobID* cancels the print request, even if it is currently printing.
- *PrinterName* cancels the printing of your jobs on the specified queue. (If you have root user authority, all jobs on the queue are deleted.)

You can use the **-W** flag with the **enq**, **qchk**, **lpstat**, and **lpq** status commands to display more job number digits.

If your queue display shows duplicate 3-digit job numbers, use **qchk -W** to list job numbers with greater precision. You can then cancel a specific job.

For example, `qchk` might display job number 123 twice while, `qchk -W` would display job number 1123 and 2123. If you want to cancel job number 2123, specifying `cancel 123`, causes the **qdaemon** to cancel the first matching job number it finds in its internal list, which may be 1123. By having the additional information that the **-W** flag provides, you can cancel a specific job number.

And for remote print jobs, both the *JobID* and remote *QueueName* must be specified in order to explicitly cancel a job on a remote queue.

Notes:

1. You must have root-user authority, or be a member of the **print** group, to cancel print requests that were not submitted by your current ID.
2. The *JobID* must be a number.
3. If you enter `cancel -?`, the system displays the following error message:

```
enq: (FATAL ERROR): 0781-048: Bad queue or device name: -?
```

Exit Status (line printer requests)

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security (line printer requests)

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files (line printer requests)

Item	Description
/var/spool/qdaemon/*	Contains temporary copies of enqueued files.
/var/spool/lpd/qdir/*	Contains job description files for print jobs.
/usr/bin/cancel	Contains the command file.

Purpose (print requests)

Cancel print requests

Syntax (print requests)

cancel [*request-IDs*] [*printers*]

cancel -u *login-IDs* [*printers*]

Description (print requests)

The **cancel** command allows users to cancel print requests previously sent with the **lp** command. The first form of **cancel** permits cancellation of requests based on their *request-ID*. The second form of **cancel** permits cancellation of requests based on the *login-ID* of their owner.

Canceling a print request

The **cancel** command cancels requests for print jobs made with the **lp** command. The first form allows a user to specify one or more *request-IDs* of print jobs to be canceled. Alternatively, the user can specify one or more *printers*, on which only the currently printing job will be canceled if it is the user's job.

The second form of **cancel** cancels all jobs for users specified in *login-IDs*. In this form the *printers* option can be used to restrict the printers on which the users' jobs will be canceled. Note that in this form, when the *printers* option is used, all jobs queued by the users for those printers will be canceled. A printer class is not a valid argument.

A user without special privileges can cancel only requests that are associated with his or her own login ID; To cancel a request, a user issues the command:

```
cancel -u login-ID [printer]
```

This command cancels all print requests associated with the *login-ID* of the user making the request, either on all printers (by default) or on the printer specified.

Administrative users with the appropriate privileges can cancel jobs submitted by any user by issuing the following types of commands:

cancel -u "*login-ID-list*"
Cancels all requests (on all relevant printers) by the specified users, including those jobs currently being printed. Double quotation mark must be used around *login-ID-list* if the list contains blanks. The argument *login-ID-list* may include any or all of the following constructs:

login-ID
a user on the local system

system-name*!*login-ID
a user on system *system-name*

***system-name*!all**
all users on system *system-name*

all!*login-ID*
a user on all systems

all
all users on the local system

all!all
all users on all systems

A remote job can be canceled only if it originated on the client system; that is, a server system can cancel jobs that came from a client, and a client system can cancel jobs it sent to a server.

cancel -u "*login-ID-list*" *printer-1 printer-2 printer-n*
Cancels all requests by the specified users for the specified printers, including those jobs currently being printed. (For a complete list of printers available on your system, execute the **lpstat -p** command.)

In any of these cases, the cancellation of a request that is currently printing frees the printer to print the next request.

RBAC Environment

This command implements and can perform privileged operations. Only privileged users can run such privileged operations. To review the list of privileges and the authorizations associated with this command, refer to the **/etc/security/privcmds** database.

canonls Command

Purpose

Processes **troff** command output for the Canon LASER SHOT in LIPS III mode.

Syntax

canonls [**-eg***File*] [**-em***File*] [**-F***Directory*] [**-quietly**] [**-ug***File*] [**-um***File*] [*File* ...]

Description

The **canonls** command processes **troff** command output for the Canon LASER SHOT in LIPS III mode. This command is provided exclusively for Japanese language support.

The **canonls** command processes one or more files specified by the *File* parameter. If no file is specified, the **canonls** command reads from standard input.

The **canonls** command uses font files in the **/usr/lib/font/devcanonls** directory that have command names ending with **.out**. The **canonls** command does not produce correct output unless these files are provided.

Flags

Item	Description
-eg*File*	Specifies the Gothic font for the IBM Japanese extended character set. By default, the **canonls** command uses the Gothic font found in the **/usr/lib/X11/fonts/JP/IBM_JPN23G.snf** file.
-em*File*	Specifies the Mincho font for the IBM Japanese extended character set. By default, the **canonls** command uses the Mincho font found in the **/usr/lib/X11/fonts/JP/IBM_JPN23.snf** file.
-F*Directory*	Specifies a directory name as the place to find font files. By default, the **canonls** command looks for font files in the **/usr/lib/font/devvcanonls** directory.
-quietly	Suppresses all nonfatal error messages.
-ug*File*	Specifies the Gothic font for the user-defined characters of Japanese. By default, the **canonls** command uses the Gothic font found in the **/usr/lib/X11/fonts/JP/IBM_JPN23G.snf** file.
-um*File*	Specifies the Mincho font for the user-defined characters of Japanese. By default, the **canonls** command uses the Gothic font found in the **/usr/lib/X11/fonts/JP/IBM_JPN23.snf** file.

Example

To process the `reports` file for the Canon LASER SHOT printer, enter:

```
troff reports |canonls | qprt -dp
```

The **canonls** command first processes the output of the **troff** command, then sends the file to a print queue.

File

Item	Description
/usr/lib/font/devcanonls/*.out	Contains font files.

captoinfo Command

Purpose

Converts a **termcap** file to a **terminfo** descriptor file.

Syntax

captoinfo [**-w***Number*] [**-v**] [**-V**] [**-1**] [*FileName*...]

Description

The **captoinfo** command converts a **termcap** source file to a **terminfo** source file and displays it on the screen. The **termcap** file format is an older format. The **termcap** and **terminfo** files differ mainly in the capability names and the entry syntax. Therefore, the **captoinfo** command only makes the syntactical transformations and vocabulary substitutions. The command also strips obsolete **termcap** capabilities such as nc, and 2-character **termcap** names like D3.

By default, the **captoinfo** command converts the **termcap** description for the terminal specified by the **TERM** environment variable. The command reads the description of the terminal from the **/etc/termcap**

file and outputs a **terminfo**-style description. If you specify the *Filename* parameter, the command converts all the descriptions in the file to **terminfo** format.

You can redirect the output of the **captoinfo** command to a file.

Flags

Item	Description
-v	Turns on the verbose mode.
-V	Displays the version number.
-w*Number*	Defines the line width of the **terminfo** entry. The **captoinfo** command fits as many **terminfo** fields in this width as is possible on the output line. A **terminfo** field consists of a capability name and a corresponding value. If you specify the **-w** flag, you must specify a *Number* parameter. By default, the line width is 60. **Notes:** 1. If the width you specify is too small to contain even one field, the command displays one field per line. 2. If the width you specify is zero or negative, the line width will be set to 60.
-1	Displays one **terminfo** field per line.

Examples

1. To convert the **termcap** file **Wyse50.tc** to a **terminfo** file and see the results on the display, enter:

   ```
   captoinfo   Wyse50.tc
   ```

2. To convert the **termcap** file **Wyse50.tc** to a **terminfo** file and save the results, enter:

   ```
   captoinfo   Wyse50.tc   >   Wyse50.ti
   ```

3. To display one **terminfo** field per line and see more information, enter:

   ```
   captoinfo   -1   -v   Wyse50.tc
   ```

4. To produce a **terminfo** description of an ibm3101 terminal defined by the **TERM** environment variable, enter:

   ```
   captoinfo -w 40
   ```

   The **captoinfo** command converts the ibm3101 description in the **/etc/termcap** file into a **terminfo** description and produces a description with a 40 character width. The output of the command is similar to the following:

   ```
   ibm|ibm3101|3101|i3101|IBM 3101-10,
            am, xon,
            cols#80, lines#24,
            bel=^G, clear=\EK, cr=\r, cub1=\b,
            cud1=\n, cuf1=\EC,
            cup=\EY%p1%'\s'%+%c%p2%'\s'%+%c,
            cuu1=\EA, ed=\EJ, el=\EI,
            home=\EH, ht=\t, ind=\n,
            kcub1=\ED, kcud1=\EB, kcuf1=\EC,
            kcuu1=\EA,
   ```

capture Command

Purpose

Allows terminal screens to be dumped to a file.

Syntax

capture [**-a**] [*File*]

Description

The **capture** command allows a user to dump everything printed on the user's terminal to a file. The screen is printed to the file specified by the *File* parameter or to the **screen.out** file if no file is specified. If the **-a** flag is specified, the **capture** command appends the contents of the screen to the file.

In order to dump the screen to a file, the **capture** command creates a shell that emulates a VT100 terminal and maintains a record of what is being displayed on the screen. The **SHELL** environment variable determines the shell created. If the **SHELL** environment variable is not set, the **/usr/bin/bsh** shell is the default. The **TERM** environment variable is set to `TERM=vt100`. If, while running the **capture** command, the program asks for the terminal type in use, the user must enter `vt100`.

The Ctrl-P key sequence is the default keystroke to cause a screen dump to be performed. This can be changed by setting the **SCREENDUMP** environment variable to the 3-digit octal value of the desired screen dump key. For example, setting:

```
SCREENDUMP=014
```

changes the screen dump keystroke to Ctrl-L. Trying to set the **SCREENDUMP** environment variable by entering `^L` or `'\014'` results in an error message.

To stop the screen capture process, use the Ctrl-D key sequence or type `exit`. The system displays the message, `You are NO LONGER emulating a vt100 terminal`.

Flags

Item	Description
-a	Appends the screen contents to the specified file or, if no file is specified, to the **screen.out** file.

Files

Item	Description
/usr/bin/capture	Contains the **capture** command.

cat Command

Purpose

Concatenates or displays files.

Syntax

cat [**-** **q**] [**-r**] [**-** **s**] [**-** **S**] [**-** **u**][**-** **Z**] [**-** **n** [**-** **b**]] [**-** **v** [**-** **e**] [**-** **t**]] [**-** | *File* ...]

Description

The **cat** command reads each *File* parameter in sequence and writes it to standard output. If you do not specify a file name, the **cat** command reads from standard input. You can also specify a file name of **-** (dash) for standard input.

Attention: Do not redirect output to one of the input files using the redirection symbol, > (greater than symbol). If you do this, you lose the original data in the input file because the shell truncates the file before the **cat** command can read it.

Note: The I/O buffer size for the read and write system calls generated by this command can be configured by using the *AIX_STDBUFSZ* environment variable.

Flags

Item	Description
-b	Omits line numbers from blank lines, when specified with the **-n** flag.
-e	Displays a $ (dollar sign) at the end of each line, when specified with the **-v** flag.
-n	Displays output lines preceded by line numbers, numbered sequentially from 1.
-q	Does not display a message if the **cat** command cannot find an input file. This flag is identical to the **-s** flag.
-r	Replaces multiple consecutive empty lines with one empty line. This flag is identical to the **-S** flag.
-s	Does not display a message if the **cat** command cannot find an input file. This flag is identical to the **-q** flag. **Note:** Previously, the **-s** flag handled tasks now assigned to the **-S** flag.
-S	Replaces multiple consecutive empty lines with one empty line. This flag is identical to the **-r** flag.
-t	Displays tab characters as ^I if specified with the **-v** flag.
-u	Does not buffer output. The default is buffered output.
-v	Displays nonprinting characters as visible characters, with the exception of tabs, new-lines, and form-feeds. ASCII control characters (octal 000–037) are printed as ^*n*, where *n* is the corresponding ASCII character in the octal range 100–137 (@, A, B, C,..., X, Y, Z, [, \,], ^, and _); the DEL character (octal 0177) is printed as ^?. Other non-printable characters are printed as M-*x*, where *x* is the ASCII character specified by the low-order seven bits. When used with the -v option, the following options may be used: **-e** A $ character will be printed at the end of each line prior to a new line. **-t** Tabs will be printed as ^I and form feeds will be printed as ^L The -e and -t options are ignored if the -v option is not specified.
-	Allows standard input to the **cat** command.
Z	Dumps the contents of encrypted files in encrypted format. Access keys to the encrypted file are not required to do cat **-Z** on the file.

Exit Status

This command returns the following exit values:

Item	Description
0	All input files were output successfully.
>0	An error occurred.

Examples

Attention: Do not redirect output to one of the input files using the redirection symbol, > (caret).

1. To display a file at the workstation, enter:

   ```
   cat notes
   ```

   This command displays the data in the `notes` file. If the file is more than one less than the number of available display lines, some of the file scrolls off the screen. To list a file one page at a time, use the **pg** command.

2. To concatenate several files, enter:

   ```
   cat section1.1 section1.2 section1.3 >section1
   ```

   This command creates a file named `section1` that is a copy of `section1.1` followed by `section1.2` and `section1.3`.

3. To suppress error messages about files that do not exist, enter:

   ```
   cat  -q section2.1 section2.2 section2.3 >section2
   ```

   If `section2.1` does not exist, this command concatenates `section2.2` and `section2.3`. The result is the same if you do not use the **-q** flag, except that the **cat** command displays the error message:

   ```
   cat: cannot open section2.1
   ```

   You may want to suppress this message with the **-q** flag when you use the **cat** command in shell procedures.

4. To append one file to the end of another, enter:

   ```
   cat section1.4 >> section1
   ```

   The >> (two carets) appends a copy of `section1.4` to the end of `section1`. If you want to replace the file, use the > (caret).

5. To add text to the end of a file, enter:

   ```
   cat >>notes
   Get milk on the way home
   Ctrl-D
   ```

   This command adds `Get milk on the way home` to the end of the file called `notes`. The **cat** command does not prompt; it waits for you to enter text. Press the Ctrl-D key sequence to indicate you are finished.

6. To concatenate several files with text entered from the keyboard, enter:

   ```
   cat section3.1 - section3.3 >section3
   ```

   This command concatenates the file `section3.1` with text from the keyboard (indicated by the minus sign), and the file `section3.3`, then directs the output into the file called `section3`.

Files

Item	Description
/usr/bin/cat	Contains the **cat** command.

catman Command

Purpose

Creates the cat files for the manual.

Syntax

catman [**-n** | **-p** | **-w**] [**-M** *Path*] [*Section...*]

Description

The **catman** command creates the preformatted versions of the online manual from the **nroff** command input files. The **catman** command examines each manual page and re-creates those pages whose preformatted versions are missing or out of date. If any changes are made, the **catman** command re-creates the command **whatis** database.

Flags

Item	Description
-M *Path*	Updates manual pages located in the set of directories specified by the *Path* variable (the **/usr/share/man** directory by default).The *Path* variable has the form of a colon (:) separated by a list of directory names. For example: `'/usr/local/man:/usr/share/man'` If the environment variable **MANPATH** is set, its value is used for the default path. If the **nroff** command source file contains a line such as: `' .so manx/yyy.x'` a symbolic link is made in the **catx** directory to the appropriate preformatted manual page. This allows easy distribution of the preformatted manual pages among a group of associated machines using the **rdist** command. The **nroff** command sources need not be distributed to all machines, thus saving the associated disk space. For example, a local network of five machines (called mach1 through mach5) has mach3 with the manual page **nroff** command sources. Every night, mach3 runs the **catman** command by using the **cron** daemon and later runs the **rdist** command with a **distfile** file that looks like the following: `MANSLAVES = (mach1 mach2 mach4 mach5)` `MANUALS = (/usr/share/man/cat[1-8no] /usr/share/man/whatis)` `${MANUALS}    ->    ${MANSLAVES}` `install -R;` `notify root;`
-n	Prevents creation of the **whatis** command database.
-p	Prints the names of the manual pages that need to be recreated or updated without recreating or updating them.

Item	Description
-w	Reads the Berkeley Software Distribution (BSD) style manual pages in the **/usr/share/man/cat?/*.*** and **/usr/share/man/man?/*.*** files, and then reads the hypertext information bases and creates the **/usr/share/man/whatis** database. **Tip**: If the base EN_US documentation fileset is installed on the system, set the ilocale to en_US to build a complete **whatis** database.

Examples

To update manual sections 1, 2, and 3 only, enter:

```
catman 123
```

Files

Item	Description
/usr/lbin/getNAME	Contains the command to create the **whatis** database.
/usr/share/man	Specifies the default manual directory location.
/usr/share/man/man?/*.*	Contains the raw (the **nroff** command input) manual sections.
/usr/share/man/cat?/*.*	Contains preformatted manual pages.
/usr/share/man/whatis	Contains the **whatis** command database.
/usr/lbin/mkwhatis	Contains the command script to make the **whatis** command database.

cb Command

Purpose

Puts C source code into a form that is easily read.

Syntax

cb [**-s**] [**-l** *Length* | **-j**] [*File* ...]

Description

The **cb** command reads C programs from standard input or from specified files and writes them to standard output in a form that shows, through indentations and spacing, the structure of the code. When called without flags, the **cb** command does not split or join lines. Note that punctuation in preprocessor statements can cause indentation errors.

For best results, use this command on source code that is syntactically correct.

Flags

Item	Description
-j	Joins lines that are split. Ignored if **-l** flag is given.
-l *Length*	Splits lines that are longer than *Length* characters.
-s	Formats the source code according to the style of Kernighan and Ritchie in *The C Programming Language* (Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1978).

Example

To create a version of `pgm.c` called `pgm.pretty.c` that is easy to read, enter:

```
cb pgm.c > pgm.pretty.c
```

Files

Item	Description
/usr/ccs/bin/cb	Contains the **cb** command.
/usr/bin/cb	Symbolic link to the **cb** command.

cd Command

Purpose

Changes the current directory.

Syntax

cd [*directory*]

or

cd [*directorya directoryb*]

Description

The **cd** command sets the current working directory of a process. The user must have execute (search) permission in the specified directory.

If a directory parameter is not specified, the **cd** command sets the current working directory to the login directory (**$HOME** in the **ksh** and **bsh** environments, or **$home** in the **csh** environment). If the specified directory name is a full path name, it becomes the current working directory. A full path name begins with a / (slash) indicating root directory, a **.** (dot) indicating current directory, or a **..** (dot-dot) indicating parent directory. If the directory name is not a full path name, the **cd** command searches for it relative to one of the paths specified by the **$CDPATH** shell variable (or **$cdpath csh** variable). If the **cd** command is unsuccessful in searching the components, it throws the failure message of the last component it searched. This variable has the same syntax as, and similar semantics to, the **$PATH** shell variable (or **$path csh** variable).

Note: Running **/usr/bin/cd** from a shell does not change the shell's working directory. The shell's built-in **cd** command must be used.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To change the current working directory to the login (home) directory, type:

```
cd
```

2. To change to an arbitrary directory, type:

   ```
   cd /usr/include
   ```

   This changes the current directory to `/usr/include`.
3. To go down one level of the directory tree, type:

   ```
   cd sys
   ```

   If the current directory is `/usr/include` and it contains a subdirectory named `sys`, then `/usr/include/sys` becomes the current directory.
4. To go up one level of the directory tree, type:

   ```
   cd ..
   ```

   The special file name, `..` (dot-dot), refers to the directory immediately above the current directory.
5. Specifying two directory parameters substitutes the string **directoryb** for the string **directorya** in the current working directory, then makes the new path the current directory. For example, if the current working directory is

   ```
   /home/directorya/sub1/sub2/sub3/sub4
   ```

   the command

   ```
   cd directorya directoryb
   ```

   will set the current working directory to

   ```
   /home/directoryb/sub1/sub2/sub3/sub4
   ```

   if that directory exists. Additionally, if the current working directory is:

   ```
   home/directorya/sub1/sub2/sub3/sub4
   ```

   the command

   ```
   cd directorya directoryb/test
   ```

   will set the current working directory to

   ```
   home/directoryb/test/sub1/sub2/sub3/sub4
   ```

   if that directory exists. Likewise, if the current working directory is

   ```
   /home/directoryb/test/sub1/sub2/sub3/sub4
   ```

   the command

   ```
   cd directoryb/test directorya
   ```

   will set the current working directory to

   ```
   home/directorya/sub1/sub2/sub3/sub4
   ```

   if that directory exists.

Subdirectories must all have the same name.

cdc Command

Purpose

Changes the comments in a SCCS delta.

Syntax

cdc -r*SID* [**-m** [*ModificationRequestList*]] [**-y** [*Comment*]] *File ...*

Description

The **cdc** command changes the Modification Requests (MRs) and comments for the specified SCCS delta (the *SID* variable) for each named Source Code Control System (SCCS) file. If you specify a directory name, the **cdc** command performs the requested actions on all SCCS files in that directory (that is, all files with names that have the **s.** prefix). If you specify a **-** (minus) in place of *File*, the **cdc** command reads standard input and interprets each line as the name of an SCCS file.

You can change the comments and MRs for an SID only if you made the SID or you own the file and the directory.

Flags

Item	Description
-m[*ModificationRequestList*]	Supplies a list of MR numbers for the **cdc** program to add or delete in the SID specified by the **-r** flag. You can only use this flag if the specified file has the **v** header flag set. A null MR list has no effect. In the actual *ModificationRequestList* parameter, MRs are separated by blanks, tab characters, or both. To delete an MR, precede the MR number with an ! (exclamation point). If the MR you want to delete is currently in the list of MRs, it is changed into a comment line. The **cdc** command places a list of all deleted MRs in the comment section of the delta and precedes them with a comment line indicating that the MRs were deleted. If you do not specify the **-m** flag, and the **v** header flag is set, MRs are read from standard input. If standard input is a workstation, the **cdc** command prompts you for the MRs. The first new-line character not preceded by a backslash ends the list on the command line. The **cdc** command continues to take input until it reads an end-of-line character or a blank line. MRs are always read before comments (see the **-y** flag). If the **v** header flag has a value, the **cdc** command interprets the value as the name of a program that validates MR numbers. If the MR number validation program returns a nonzero exit value, the **cdc** command stops and does not change the MRs.
-r*SID*	Specifies the SCCS identification number of the delta for which the **cdc** command will change the comments or MRs.

Item	Description
-y[*Comment*]	Specifies comment text to replace an existing comment for the delta specified by the **-r** flag. The **cdc** command keeps the existing comments but precedes them by a comment line stating that they were changed. A null *Comment* value has no effect. If you do not specify the **-y** flag, the **cdc** command reads comments from standard input until it reads an end-of-file character. If the standard input is a workstation, the **cdc** command prompts for the comments and also allows a blank line to end input. If the last character of a line is a \ (backslash), the **cdc** command ignores it and continues to read standard input. **Note:** If the **cdc** command reads standard input for file names (that is, when you specify a file name of **-**), you must use the **-y** and **-m** flags.

Example

To change the comment for SID 1.3 of SCCS file `s.text.c` to `"new comment"`, enter:

```
cdc -r1.3 -y"new comment" s.test.c
```

Files

Item	Description
/usr/bin/cdc	Contains the path to SCCS **cdc** command**.**

cdcheck Command

Purpose

Asks **cdromd** daemon information about a device.

Syntax

cdcheck { **-a** | **-m** | **-u** | **-e** } [**-q**] [**-h** | **-?**] *DeviceName*

Description

The **cdcheck** command sends an appropriate command to the **cdromd** daemon to get information on a media or a device depending on the flag used.

The **cdcheck** command returns a zero (True) exit value and prints a message on **stdout** if the specified condition is true. Otherwise, the **cdcheck** command returns a nonzero (False) exit value and prints an error message on **stderr**.

To check if a device is managed by **cdromd** daemon, use the **cdcheck** command with the **-a** flag. If the **cdromd** daemon is running and the specified device is in its device list, the **cdcheck -a** command will return with a zero (True) exit value after printing the following message on **stdout**:

```
        cd<x> is managed by cdromd.
```

Note: An exit value of zero (True) with the **-a** flag means that a media will be automatically mounted when it is inserted. It does not mean that a media is currently mounted.

To check if a media is present and was mounted by **cdromd** daemon, use the **cdcheck** command with the **-m** flag. When a media is inserted in a drive, it can take several seconds or tens of seconds before it become ready and mounted. The **cdcheck -m** command waits until the end of the mount operation by the **cdromd** daemon. If this operation is successful, the **cdcheck -m** command returns with a zero (True) exit value after printing the mount point on **stdout**.

Note: If the media is damaged and can't be mounted by the **cdromd** daemon, the **cdcheck -m** command returns a nonzero (False) exit value and prints an error message on **stderr**.

To check if a media is present but was unmounted by the **cdumount** command, use the **cdcheck** command with the **-u** flag. If the **cdromd** daemon is running and the specified device is in in unmounted state, the **cdcheck -u** command will return with a zero (True) exit value after printing the following message on **stdout**:

```
        cd<x> is not mounted.
```

To check that there is no media present in the specified device, use the **cdcheck** command with the **-e** flag. If the **cdromd** daemon is running and there is no media present in the drive, the **cdcheck -e** command will return with a zero (True) exit value after printing the following message on **stdout**:

```
        No media present in cd<x>.
```

When using **cdcheck** in shell scripts, the **-q** flag can be added to the **cdcheck** command so that no messages are printed on **stdout** and **stderr**. The only exception is the **cdcheck** command with the **-m** flag, which always prints the mount point on **stdout** so that the shell script can get this mount point.

Flags

Item	Description
-a	Checks if a device is managed by **cdromd**.
-e	Checks if a media has been ejected from a device.
-h or **-?**	Displays the command usage message.
-m	Checks if a media is mounted on a device.
-q	Specifies silent mode: Doesn't print any information or error message. **Note:** If **-q** is used with the **-m** flag, the mount point will be printed to **stdout**.
-u	Checks if a media is not mounted on a device.
DeviceName	Specifies the name of the device.

Exit Status

This command returns the following exit values:

0
answer = yes.

>0
answer = no or error.

Examples

1. To ask cdromd if cd0 is managed enter:

```
cdcheck -a cd0
```

2. To ask **cdromd** if a media is mounted on **cd1** without any printed error messages, enter:

```
cdcheck -m -q cd1
```

3. To ask **cdromd** if a media is not mounted on **cd1** enter:

```
      cdcheck -u cd1
```

4. To ask **cdromd** if a media is not present on **cd0** enter:

```
cdcheck -e cd0
```

5. Shell script example:

```
DEVICE=$1

if [ cdcheck -a -q "$DEVICE" ]; then
    AUTO_MOUNT="ON"
else
    AUTO_MOUNT="OFF"
fi

# Other initializations
# ...

if [ "$AUTO_MOUNT" = "ON" ]; then
    MOUNT_POINT=`cdcheck -m -q $DEVICE`
else
    MOUNT_POINT="/tmp/MyProg_$$"
    mount -rv cdrfs $DEVICE $MOUNT_POINT
fi
if [ $? -ne 0 ]; then
    echo "mount $DEVICE failed"
    exit 1
fi

# Now extract data from $MOUNT_POINT...
# ...


# End of processing. Umount the media
if [ "$AUTO_MOUNT" = "ON" ]; then
    cdeject -q $DEVICE
else
    unmount $DEVICE
fi
if [ $? -ne 0 ]; then
    echo "unmount $DEVICE failed"
    exit 1
fi
```

cdeject Command

Purpose

Ejects a media from a CD drive managed by the **cdromd** daemon.

Syntax

cdeject [**-q**] [**-h** | **-?**] *DeviceName*

Description

The **cdeject** command sends an appropriate command to the **cdromd** daemon which unmounts (if necessary) the file system corresponding to the specified device and ejects the media from the drive specified by *DeviceName*.

Flags

Item	Description
-h or **-?**	Displays the command usage message.
-q	Specifies silent mode. If you specify this option, any information or error messages are not printed.
DeviceName	Specifies the name of the device.

Exit Status

This command returns the following exit values:

0
No error.

>0
An error occurred.

Examples

1. To eject a media from **cd0**, enter:

```
cdeject cd0
```

2. To eject a media from **cd1** without any printed error messages, enter:

```
cdeject -q cd1
```

cdmount Command

Purpose

Makes a file system available for use on a device managed by **cdromd**.

Syntax

cdmount [**-q**] [**-h** | **-?**] *DeviceName*

Description

The **cdmount** command sends an appropriate command to the **cdromd** daemon which mounts the file system on the device specified by *DeviceName* if it is not already mounted. This command can be used to mount a file system that was previously unmounted by the **cdumount** command.

The mount point used is either the one found in **/etc/cdromd.conf** file for the specified *DeviceName* or the default one (**/cdrom/cd0** for cd0, **/cdrom/cd1** for cd1, etc...).

The file system type and options used (**-o** and **-V** flag for **mount** command) are those found in **/etc/cdromd.conf** file or the default ones: "**-Vcdrfs -oro**" for a CD-ROM and "**-Vudfs -oro**" or "**-Vcdrfs -oro**" for DVD-ROM.

Flags

Item	Description
-h or **-?**	Displays the command usage message.
-q	Specifies silent mode: Doesn't print any information or error message.

Item	Description
DeviceName	Specifies the name of the device.

Exit Status

This command returns the following exit values:

0
No error.

>0
An error occurred.

Examples

1. To mount a file system on **cd0** enter:

```
cdmount cd0
```

2. To mount a file system on **cd1** without any printed error messages, enter:

```
cdmount -q cd1
```

cdpd Daemon

Purpose

Receives the incoming data packets by using the Cisco Discovery Protocol (CDP) and discovers the physically connected Cisco devices.

Syntax

```
startsrc -s cdpd
```

Description

The CDP is used by networking applications that run on Cisco devices to discover other Cisco devices that are directly connected to the device. Starting from AIX 7 with 7200-05, the CDP functions are implemented on an AIX logical partition by using the **cdpd** daemon. The **cdpd** daemon controls the incoming data packets or messages by using the CDP.

To configure the **cdpd** daemon and the network interface for an AIX logical partition, run the **cdpctl** command.

Note: The CDP data packets or messages can be received only by physical interfaces that are connected to the Cisco device and not by virtual interfaces.

Example

- To start the **cdpd** daemon through the System Resource Controller (SRC) subsystem, enter the following command:

```
startsrc -s cdpd
```

An output that is similar to the following example is displayed:

```
0513-059 The cdpd Subsystem has been started. Subsystem PID is 14745930.
```

Files

/usr/sbin/cdpd
Contains the **cdpd** daemon.

cdpctl Command

Purpose

Controls the **cdpd** daemon that receives the incoming data packets or messages by using the Cisco Discovery Protocol (CDP) and discovers the physically connected Cisco devices.

Syntax

```
cdpctl subcommand device_name
```

Description

The CDP is a Layer 2 (data link layer) protocol that is media-independent and network-independent. The CDP is used by networking applications that run on Cisco devices to discover other Cisco devices that are directly connected to the device. Starting from AIX 7 with 7200-05, the CDP functions are implemented on an AIX logical partition by using the **cdpd** daemon to receive the incoming data packets or messages by using the CDP. The System Resource Controller (SRC) subsystem starts the **cdpd** daemon when you run the following command:

```
startsrc -s cdpd
```

The **cdpctl** command controls the **cdpd** daemon by configuring the network interface on an AIX logical partition to receive the incoming data packets or messages by using the CDP that contains information about physically connected Cisco devices. The **cdpctl** command also discovers the physically connected Cisco devices and the associated configuration settings by detecting the data packets or messages that are broadcast from the connected Cisco devices.

Note: The CDP functions are implemented on AIX operating system to receive CDP messages and to discover physically connected devices only. The CDP data packets can be received only by physical interfaces that are connected to the Cisco device and not by virtual interfaces.

Subcommands

list
Displays list of network interfaces on the AIX logical partition that are CDP-capable. CDP-capable network interface means that the network interface can be configured to receive incoming data packets or messages by using CDP.

Syntax: `cdpctl list`

add
Adds a CDP-capable Cisco device or network interface to the AIX logical partition.

Syntax: `cdpctl add` *`device_name`*

show portlist
Displays list of network interfaces that are configured to receive incoming data packets or messages by using CDP.

Syntax: `cdpctl show portlist`

show port
Displays CDP information (such as switch and router capabilities) that is received on configured CDP-capable network interfaces.

Syntax: `cdpctl show port` *`device_name`*

remove
Removes the configured network interface from receiving any incoming data packets or messages by using the CDP from the connected Cisco devices.

Syntax: `cdpctl remove` *`device_name`*

Examples

- To display the usage of the **cdpctl** command, enter the following command:

```
cdpctl
```

- To list all the CDP-capable devices and network interfaces in the AIX logical partition, enter the following command:

```
cdpctl list
```

An output that is similar to the following example is displayed:

```
CDP capturable devices on the system:
en0
en2
```

- To add a network interface to the AIX logical partition that can receive data packets or messages from the physically connected Cisco devices by using the CDP, enter the following command:

```
cdpctl add en0
```

An output that is similar to the following example is displayed:

```
Successfully added port en0
Waiting for CDP advertise (default 60 seconds)......

Device ID                  : sw-yyyyyycisco92300.xxx.xxxx.xxx.com(XXXXXXXKQ)
Address                    : 9.x.y.z
Port ID                    : Ethernet1/49
Capabilities
                           : Router Level 3
                           : Level 2 Switch
                           : IGMP snooping
Cisco switch OS Version    : Cisco Nexus Operating System (NX-OS) Software, Version 9.2(3)
Platform                   : N9K-C92300YC
Native VLAN ID             : 1
Trusted Bitmap             : N/A
AVVID untrusted ports      : N/A
Duplex                     : Full
MTU                        : 1500
System Name                : sw-xxxxxx92300
System Object ID           : N/A
Management Addresses       : 9.x.y.z
```

- To list the network interfaces on the AIX logical partition that are configured to receive data packets and messages by using CDP, enter the following command:

```
cdpctl show portlist
```

An output that is similar to the following example is displayed:

```
en0
en1
```

Files

/usr/sbin/cdpd
Contains the **cdpd** daemon.

/usr/sbin/cdpctl
Contains the **cdpctl** command.

cdromd Command

Note: Use System Resource Controller (SRC) commands to control the **cdromd** daemon from the command line. To have the **cdromd** daemon enabled on each system startup, add the following line to **/etc/inittab**:

```
cdromd:23456789:wait:/usr/bin/startsrc -s cdromd
```

Purpose

Automatically mounts a CD-ROM or DVD-ROM when it is inserted in a device, and provides the server function for the **cdutil**, **cdcheck**, **cdmount**, **cdumount**, and **cdeject** commands.

Syntax

cdromd [**-d**]

Description

The **cdromd** daemon finds the device list it has to manage and their respective mount points in **/etc/cdromd.conf** file. If this file does not exist or is empty, **cdromd** manages all the CD-ROM and DVD-ROM devices available on the system, and the mount points are **/cdrom/cd0** for **cd0**, **/cdrom/cd1** for **cd1**, etc.

After its init phase **cdromd** periodically checks if a media is present in one of the managed drives (for devices that are not already mounted) and mounts it if there is a media.

cdromd also periodically checks its socket for requests coming from **cdutil**, **cdcheck**, **cdmount**, **cdumount** or **cdeject** commands.

The **cdromd** daemon should be controlled using the System Resource Controller (SRC). Entering **cdromd** at the command line is not recommended.

The cdromd daemon sends its error messages to the **syslogd** daemon.

The **cdromd** daemon can interfere with scripts, applications, or instructions that attempt to mount the CD or DVD device without first checking to see if the device is already enabled. A resource or device busy error will occur in such a condition. Use the **cdumount** or **cdeject** command to unmount the device so that you can mount the device as specified in the program or instructions. Alternatively, use the **cdcheck -m** or **mount** command to determine the current mount point of the device.

Manipulating the cdromd daemon with the System Resource Controller:

The **cdromd** daemon is a subsystem controlled by the System Resource Controller (SRC). Its subsystem name is **cdromd**. The **cdromd** daemon can be manipulated by the following SRC commands:

stopsrc
Stops a subsystem, group of subsystems, or a subserver.

startsrc
Starts a subsystem, a group of subsystems, or a subserver.

refresh
Requests a refresh of a subsystem or group of subsystems.

traceson
Turns on tracing of a subsystem, group of subsystems, or a subserver.

tracesoff
Turns off tracing of a subsystem, group of subsystems, or a subserver.

lssrc
Gets the status of a subsystem, group of subsystems, or a subserver.

In addition, the **cdromd** daemon can be controlled by issuing signals using the **kill** command. Sending a **SIGHUP** signal to cdromd is equivalent to the "`refresh -s cdromd`" command, and sending a **SIGTERM** signal to **cdromd** is equivalent to the "`stopsrc -s cdromd`" command.

Flags

Item	Description
-d	Sends debugging messages to **syslogd** daemon.

Exit Status

This daemon returns the following exit values:

0
The **cdromd** daemon was stopped by SRC or **SIGTERM** signal.

>0
An error occurred.

Examples

1. To stop the cdromd daemon normally, enter the following:

   ```
   stopsrc -s cdromd
   ```

   This command stops the daemon. The **-s** flag indicates that the specified subsystem is to be stopped.
2. To start the **cdromd** daemon, enter the following:

   ```
   startsrc -s cdromd
   ```

   This command starts the daemon. This command is in the **/etc/inittab** file and can be used on the command line. The -s flag indicates that the specified subsystem is to be started.
3. To get a short status report from the **cdromd** daemon, enter the following:

   ```
   lssrc -s cdromd
   ```

   This command returns the name of the daemon, the process ID of the daemon, and the state of the daemon (active or inactive).
4. To tell **cdromd** daemon its configuration file has changed, enter the following:

   ```
   refresh -s cdromd
   ```

   This command tells the **cdromd** daemon to read its configuration file again.

Files

Item	Description
/etc/cdromd.conf	Describes managed devices and supported file systems.

cdumount Command

Purpose

Unmounts a previously mounted file system on a device managed by **cdromd**.

Syntax

cdumount [**-q**] [**-h** | **-?**] *DeviceName*

Description

The **cdumount** command sends an appropriate command to the **cdromd** daemon which tries to unmount the file system on the device specified by *DeviceName*.

The **cdumount** command doesn't eject the media.

Flags

Item	Description
-h or **-?**	Displays the command usage message.
-q	Specifies silent mode: Doesn't print any information or error messages.
DeviceName	Specifies the name of the device.

Exit Status

This command returns the following exit values:

0
No error.

>0
An error occurred.

Examples

1. To unmount a file system on **cd0** enter:

```
cdumount cd0
```

2. To unmount a file system on **cd1** without any printed error messages, enter:

```
cdumount -q cd1
```

cdutil Command

Purpose

Tells the **cdromd** daemon to suspend or resume management of a device.

Syntax

cdutil { **-l** | **-r** | **-s** [**-k**] } [**-q**] [**-h** | **-?**] *DeviceName*

Description

The **cdutil** command sends an appropriate command to the **cdromd** daemon which suspends (**-s** flag) or resumes (**-r** flag) the management of the device specified by *DeviceName*.

A device managed by **cdromd** must be set in suspend state if it needs to be unconfigured (for example for a hotswap of the parent adapter).

The resume flag (**-r**) asks **cdromd** to restart polling the device.

Flags

Item	Description
-h or **-?**	Displays the command usage message.
-k	Do not eject the media when suspending a device.
-l	Load the media if one is present in the drive.
-q	Specifies silent mode: Doesn't print any information or error messages.
-r	Resumes device management by **cdromd**.
-s	Suspends device management by **cdromd**.
DeviceName	Specifies the name of the device.

Exit Status

This command returns the following exit values:

0
No error

>0
An error occurred

Examples

1. To suspend management of **cd0** by **cdromd**, type:

```
cdutil -s cd0
```

2. To suspend management of **cd0** by **cdromd** without ejecting the media, type:

```
cdutil -s -k cd0
```

3. To resume management of **cd1** by **cdromd** without any printed error messages, type:

```
cdutil -r -q cd1
```

certadd Command

Purpose

certadd stores a certificate into the local LDAP repository.

Syntax

certadd [**-c**|**-r**] [**-p** *privatekeystore*] [**-f** *file*] **-l** *label tag* [*username*]

Description

The **certadd** command stores a user-supplied certificate in the local LDAP repository.

If the **-c** (create only) option is used, it will return an error if the username and tag pair already exists as a named certificate. Otherwise, the existing certificate shall be replaced by the new certificate. If the **-r** (replace only) option is used, an error is returned if the username and tag pair does not already exist as a named certificate. These two options are mutually exclusive. The default behavior is to create the entry if it does not exist and to replace the existing certificate if it exists.

If the **-f** option is not given, the certificate shall be read from stdin. The certificate is in DER format. The **certadd** command is limited to `root` users, or users with the appropriate administrative roles, when the username parameter is other than the current user.

The **-l** option must always be specified. The label is a variable length text string that will be used to map a key in the keystore to the certificate which contains the matching public key. Make sure this label is the same as the one specified when the **certcreate** command is invoked.

If the **-p** option is not given, the default will be **file:/var/pki/security/keys/<*username*>**. If no protocol is specified, **file:** is assumed. Currently only URIs of type **file:** are supported. It is the responsibility of the invoker of this command to ensure that the private keystore contains the private key matching the public key in the certificate. If the certificate to be added is created using the **certcreate** command, then the private key is already in the private keystore. Alternatively, if the certificate is externally created, the user can later add the private key associated with the public key to the private keystore using the **keyadd** command.

The *tag* parameter is a variable length text string from the same character set as user names which is used to uniquely identify the certificate amongst all of the certificates owned by username. The *tag* ALL shall be reserved for the **certlist** command so that all certificates owned by a user may be viewed, therefore can not be used with the **certadd** command. It shall be also an error to replace a certificate named by the **auth_cert** attribute for a user. When an existing certificate is replaced with another one, the keys corresponding to the replaced certificate remain in the keystore until deleted by the user. These keys could be removed from the keystore using key management commands. Similarly, the keys for the new certificate could also be added to the keystore again using the key management commands. Only a certificate that is not revoked can be added, unless the system policy specifies otherwise.

The system revocation check policy is specified in the policy file, **/usr/lib/security/ pki/policy.cfg** under the stanza **crl**. When the **check** attribute is set to `yes`, the certificates are checked against a CRL. The certificate revocation list will be obtained using the Certificate Revocation Distribution Point information from the certificate and from the **/usr/lib/security/pki/ca.cfg file**. This file has an entry called **crl**, which one can use to specify the method of CRL retrieval. **ldap:**, **http:** and **file:** retrieval methods are supported. If more than one URI is specified, they must be delimited with a space. The certificate will not be added if the certificate revocation list could not be retrieved.

Flags

Item	Description
-c	Adds a new certificate.
-r	Replaces an existing certificate.
-l *label*	Specifies a label for the private key that matches the public key in certificate.
-p *privatekeystore*	Specifies the location of the private keystore.
-f *file*	Specifies a file that contains the DER-encoded certificate.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Security

This is a privileged (set-UID root) command.

`Root` and invokers belonging to group security can add certificates for anybody. A non-privileged user can only add certificates for themself.

Audit

This command records the following event information:

CERT_Add <*username*>

Examples

To add a certificate stored in **cert.der** to the local LDAP repository and associate it with user Bob, enter:

```
$ certadd -c -f cert.der -l signcert cert1 bob
```

or,

```
$ certadd -c -l signcert cert1 bob < cert.der
```

This will read the DER encoded certificate from file **cert.der** and assign **signcert** as the label and **cert1** as the tag and store it in LDAP as Bob's certificate. The default private keystore location will be **/var/pki/security/keys/bob**.

To replace Bob's **cert1** certificate with another certificate enter:

```
$ certadd -r -f newcert1.der -l newsigncert cert1 bob
```

Files

/usr/lib/security/pki/acct.cfg

/usr/lib/security/pki/ca.cfg

/usr/lib/security/pki/policy.cfg

certcreate Command

Purpose

certcreate requests a new certificate for the specified user.

Syntax

certcreate [**-S** *servicename*] [**-s** *startdate*] [**-e** *enddate*] { **-f** *file* | [**-b** | **-t**] } [**-p** *privatekeystore*] **-l** *label* [**-a** *subject_alt_name*] *subject_distinguished_name* [*user-name*]

Description

The **certcreate** command invokes the end-entity services and libraries and requests that a new certificate be created with the identifying information contained on the command line. Which service to use is specified by the **-S** option. Available services are defined in **/usr/lib/security/pki/ca.cfg**. Certificate requests without the **-S** option are created using the local service. It is an error to specify a servicename which does not have an entry in the **/usr/lib/security/pki/ca.cfg** file. The service entry in the **ca.cfg** file specifies which CA to send the request.

If the **-s** option is not given, the current day's date shall be used. If the **-e** option is not given, the validity value from the **policy.cfg** file will be used. If this value does not exist, then one year from the starting date shall be used as the validity period. Both `startdate` and `enddate` shall have the same format as the `expires` attribute used by the **chuser** command. The format is 10-character string in the `MMDDhhmmyy` form, where `MM` refers to month, `DD` refers to day, `hh` refers to hour, `mm` refers to minute, and `yy` refers to last 2 digits of the years 1939 through 2038. All characters are numeric.

If the **-f** option is given, the new certificate shall be DER encoded and stored in the named file in a binary format. Otherwise, it shall be DER encoded and output to **stdout**, either in binary or in hexadecimal

format. If **-b** option is given then the output will be displayed to **stdout** in binary, otherwise it will be hexadecimal. If neither **-b** nor **-t** is given, a binary format will be used.

The corresponding private key shall be stored in a private keystore or device, as required by the underlying commands or libraries. If **-p** option is given, the private key will be stored in private keystore specified. If **-p** option is not given the default will be **/var/pki/ security/keys/<*username*>**.

The **-l** option must be specified. The label is a variable length text string that will be used as an alias for the private key in the keystore.

The value of *subject_alt_name* will be an Internet electronic mail address (RFC2459 defines this to be a rfc822Name). This value is optional. If no value is provided, the certificate will not have an rfc822Name subject alternative name extension. *Subject_distinguished_name* shall be restricted to the valid set of values for PKI certificates. This is defined to be an X.501 type Name by RFC2459.

The **certcreate** command issues one or more prompts and request a password in order to generate the certificate and store it in the user's private keystore. If the user has an existing keystore, the user will be prompted once for the password. If the keystore does not exist, then it will be created and the user will be asked to re-enter the password again for confirmation. The command will fail if it is unable to open **/dev/tty** for the current process.

Flags

Item	Description
-S *servicename*	Specifies which service module to use.
-s *startdate*	Specifies the date on which the certificate will become valid.
-e *enddate*	Specifies the date on which the certificate will become invalid.
-f *file*	Specifies the file that certificate will be stored.
-p *privatekeystore*	Specifies the location of the private keystore.
-l *label*	Specifies the label of the private key in the keystore.
-a *subject_alt_name*	Specifies the subject alternative name of the certificate owner.
-b	Specifies the format of the certificate data to be binary.
-t	Specifies the format of the certificate data to be hexadecimal.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Security

This is a **setuid** command.

Root and invokers belonging to group security can create certificates for anyone. A non-privileged user can only create certificates for himself with the following rules while specifying a private keystore location:

- The invoker can specify the default private keystore: **/var/pki/security/keys/<*user-name*>**
- The invoker can specify a private keystore that they have access to write.

A non-privileged user can not request a certificate for others.

Audit

This command records the following event information:

CERT_Create <*username*>

Examples

```
$ certcreate -S local -s 0831112702 -e 1231235902 -f
cert.der -p file:/home/bob/bob.priv -l signcert
bob@ibm.com ou=finance,cn=Bob%20James bob
```

In the above example, the certificate will be valid from August 31, 2002 11:27 AM until December 31, 2002, 11:59 PM. The certificate will be placed in file **cert.der** and the private key will be stored in **bob.priv** with an alias **signcert**.

The following example uses the defaults for the start date, end date, and the private keystore.

```
$ certcreate -l signcert bob@ibm.com ou=finance,cn=Bob James > cert.der
```

Files

/usr/lib/security/pki/ca.cfg

/usr/lib/security/pki/policy.cfg

certdelete Command

Purpose

certdelete removes a certificate from the list of certificates associated with a user account and deletes the certificate from the local LDAP repository.

Syntax

certdelete tag [*username*]

Description

The **certdelete** command removes certificates associated with a user from the local LDAP repository. A deleted certificate could be added again using the **certadd** command. Note that the **certdelete** operation does not affect the certificates in CA's LDAP store where they are published.

The **tag** parameter uniquely identifies the certificate in the list of certificates owned by a user. It shall be an error to remove the certificate named by the **auth_cert** attribute for a user. Only a privileged (`root`) user, or a user belonging to group security may specify a user name other than their own.

If invoked without the username parameter, the **certdelete** command uses the name of the current user.

Specifying ALL as the value of tag will cause all of the certificates owned by a user to be removed. The command terminates on the first delete error it encounters while processing an ALL request. This leaves the rest of the certificates owned by the user undeleted. If the error is due to some temporary condition (such as local LDAP repository is inaccessible), the next **certdelete** will delete the remaining certificates. The user might query about the certificates that did not get deleted by using **certlist** command with a tag value of ALL.

Exit Status

Item	Description
0	Successful completion.
>0	An error occured.

Security

This is a privileged (set-UID root) command.

Root and invoker belonging to group security can delete certificates for anybody. A non-privileged user can only delete certificates for himself/herself.

Audit

This command records the following event information:

CERT_Create <*username*>

Examples

1. To remove a certificate with a tag value **signcert** belonging to Bob, enter:

```
$ certdelete signcert bob
```

2. To remove all the certificates from the local LDAP repository belonging to the current user, enter:

```
$ certdelete ALL
```

Files

/usr/lib/security/pki/acct.cfg

certget Command

Purpose

certget retrieves a single certificate from local LDAP repository.

Syntax

certget {**-f** *file* | [**-b** | **-t**]}*tag* [*username*]

Description

The **certget** command retrieves a single certificate from the local LDAP repository. This command retrieves a single certificate at a time. If the invoker wishes to retrieve all the certificates for a user, the **certlist** command may be used to first to obtain a list of the certificates and then perform the **certget** operation on the certificate list.

If the **-f** option is used, the certificate shall be written in binary format to the named file. Otherwise the certificate is output to **stdout** either in binary or hexadecimal. If the **-b** option is given, binary output is used (default). If the **-t** option is given, hexadecimal output is used. Certificates are output in DER format.

The **tag** parameter uniquely selects one of the user's certificates. The **username** parameter specifies which AIX user is to be queried. If invoked without the **username** parameter, the **certdelete** command uses the name of the current user.

Flags

Item	Description
-f	Specifies the file that the DER encoded certificate will be stored.
-b	Specifies the format of the certificate data to be binary.
-t	Specifies the format of the certificate data to be hexadecimal.

Exit Status

Item	Description
0	If successful.
EINVAL	If the command is ill-formed or the arguments are invalid.
ENOENT	If a) the user doesn't exist, b) the tag does not exist c) the file does not exist.
EIO	If unable to create/modify LDAP entry.
ENOCONNECT	If the service is not available.
errno	If system error.

Security

This command can be executed by anyone to retrieve a certificate belonging to a user from the local repository.

Audit

This command records the following event information:

CERT_Get <*username*>

Examples

1. To retrieve Bob's certificate tagged as **signcert** and store in **cert.der**, enter:

```
$ certget -f cert.der signcert bob
```

2. To store Bob's certificate **signcert** in hexadecimal in **cert.der**, enter:

```
$ certget -t signcert > cert.der
```

Files

/usr/lib/security/pki/acct.cfg

certlink Command

Purpose

certlink links a certificate in a remote repository to a user account.

Syntax

certlink [**-c**|**-r**] [**-p** *privatekeystore*] **-l** *label* **-o** *option tag* [*username*]

Description

The **certlink** command links a certificate in a remote repository to a user account. **certlink** is very similar to **certadd** except that the user provides a link to the certificate rather than providing the certificate itself.

If the **-c** (create only) option is given, it is an error if the {username, tag} pair already exists as a named certificate. Otherwise, an existing certificate shall be replaced by the new certificate. If the **-r** (replace only) option is given, it is an error if the {username, tag} pair does not already exist as a named certificate. These two options are mutually exclusive. The default behavior is to create the entry if it does not exist and to replace the existing certificate if it exists.

The **-l** option must be specified. The label is a variable length text string that will be used to map a key in the keystore to the certificate which contains the matching public key.

If the **-p** option is not given, the default will be **/var/pki/security/keys/<*username*>**. It is the responsibility of the invoker of this command to add the private key associated with the public key by using the **keyadd** command. Refer to the **certadd** command for more details on the use of the **-l** and **-p** flags. This information also applies to the **certlink** command.

The **-o** option is the URI where the certificate is stored. Currently only LDAP URIs are supported. The URI of the repository must be given in the format as specified in RFC 2255.

The **tag** parameter is a variable length text string from the same character set as user names which is used to uniquely identify the certificate among all of the certificates owned by **username**. The **ALL** tag shall be reserved for the **certlist** command so that all certificates owned by a user may be viewed. An error is also returned if a certificate named by the **auth_cert** attribute for a user is replaced.

When an existing certificate is replaced with another one, the keys corresponding to the replaced certificate remain in the keystore until deleted by the user. These keys can be removed from the keystore using key management commands. Similarly, the private key matching to a certificate can also be added to the keystore using the key management commands.

Only a certificate that is not revoked can be added unless the system policy specifies otherwise. The system revocation check policy is specified in the policy file **/usr/lib/security/ pki/policy.cfg**. The certificate revocation list will be obtained using the Certificate Revocation Distribution Point information in the certificate. If one is not given, the certificate distribution point information will be retrieved from the **/usr/lib/security/ pki/ca.cfg** file. The certificate will not be added, if the certificate revocation list could not be retrieved.

Flags

Item	Description
-c	Links a new certificate.
-r	Replaces an existing certificate.
-p	Specifies the location of the private keystore.
-l *label*	Specifies a label for the private key corresponding to the public key in certificate.
-o *option*	Specifies the URL where the certificate to be linked stored.

Exit Status

Item	Description
0	If successful.
>0	An error occured.

Security

This is a privileged (set-UID root) command.

Root and *invokers* belonging to group security can add certificates for anybody. A non-privileged user can only add certificates for themself.

Examples

To link a certificate stored in an external certificate repository and associate it with user Bob, enter:

```
$ certlink -c -l signcert -p /home/bob/keystore.p12 -o ldap://
cert.austin.ibm.com/o=ibm,ou=Finance,c=us?usercertificate??(
cn=Bob James)?X-serial=1A:EF:54 cert1 bob
```

Files

/usr/lib/security/pki/ca.cfg

/usr/lib/security/pki/policy.cfg

certlist Command

Purpose

certlist lists the contents of one or more certificates.

Syntax

certlist [**-c**] [**-a** attr [attr....]]*tag* [*username*]

Description

The **certlist** command lists the contents of one or more certificates. Using the **-c** option causes the output to be formatted as colon-separated data with the attribute names associated with each field on the previous line as follows:

```
# name: attribute1: attribute2: ...
User: value1: value2: ...
```

The **-f** option causes the output to be formatted in stanza file format with the username attribute given as the stanza name. Each `attribute=value` pair is listed on a separate line:

```
user:
     attribute1=value
     attribute2=value
     attribute3=value
```

When neither of these command line options are selected, the attributes are output as `attribute=value` pairs.

The **-a** option selects a list of one or more certificate attributes to output. In addition to the attributes supported by the load module, several pseudo-attributes shall also be provided for each certificate.

Those attributes are:

Item	Description
auth_user	User's authentication certificate.
distinguished_name	User's subject distinguished name in the certificate.
alternate_name	User's subject alternate name in the certificate.
validafter	The date the user's certificate becomes valid.
validuntil	The date the user's certificate becomes invalid.

Item	Description
tag	The name that uniquely identifies this certificate.
issuer	The distinguished name of the certificate issuer.
label	The label that identifies this certificate in the private keystore.
keystore	The location of the private keystore for the private key of the certificate.
serialnumber	The serial number of the certificate.
verified	`true` indicates that the user poved that he is in possession of the private key.

Flags

Item	Description
-c	Displays the output in colon-separated records.
-f	Displays the output in stanzas.
-a *attr*	Selects one or more attributes to be displayed.

The **tag** parameter selects which of the user's certificates is to be output. The reserved value ALL indicates that all certificates for the user are to be listed.

The **username** parameter specifies the name of the AIX user to be queried. If invoked without the **username** parameter, the **certdelete** command uses the name of the current user.

Exit Status

Item	Description
0	If successful.
EINVAL	If the command is ill-formed or the arguments are invalid.
ENOENT	If a) the user doesn't exist, b) the tag does not exist c) the file does not exist.
EACCES	If the attribute cannot be listed, for example, if the invoker does not have **read_access** to the user data-base.
EPERM	If the user identification and authentication fails.
errno	If system error.

Security

This command can be executed by any user in order to list the attributes of a certificate. Certificates listed may be owned by another user.

Audit

This command records the following event information:

CERT_List <*username*>

Examples

```
$ certlist -f -a verified keystore label signcert bob
bob:
      verified=false
      keystore=file:/var/pki/security/keys/bob
      label=signcert
$ certlist -c -a validafter validbefore issuer signcert bob
#name:validafter:validuntil:issuer
bob:1018091201:1018091301:c=US,o=xyz
```

```
$ certlist -f ALL bob
bob:
      auth_cert=logincert
      distinguished_name=c=US,o=xyz,cn=bob
      alternate_name=bob@xyz.com
      validafter=0921154701
      validuntil=0921154801
      issuer=c=US,o=xyz
      tag=logincert
      verified=true
      label=loginkey
      keystore=file:/var/pki/security/keys/bob
      serialnumber=03
bob:
      auth_cert=logincert
      distinguished_name=c=US,o=xyz,cn=bob
      alternate_name=bob@ibm.com
      validafter=1018091201
      validuntil=1018091301
      issuer=c=US,o=xyz
      tag=signcert
      verified=false
      label=signkey
      keystore=file:/var/pki/security/keys/bob
      serialnumber=02
```

Files

/usr/lib/security/pki/acct.cfg

/usr/lib/security/pki/policy.cfg

certrevoke Command

Purpose

certrevoke revokes a user certificate.

Syntax

certrevoke [**-S** *servicename*] { **-f** *file* **-l** *label* [**-p** *privatekeystore*] | tag [*user-name*]}

Description

The **certrevoke** command is used to revoke certificates issued by a certificate authority which is part of the system's domain. The **-S** option specifies which service to use while revoking a certificate. Available services are defined in **/usr/lib/security/pki/ca.cfg**. Certificate requests without the **-S** option are created using the local service. An error is returned if you specify a servicename which does not have an entry in the **/usr/lib/security/pki/ ca.cfg** file.

If the **-f** option is selected, the certificate shall be read from the named file, or **stdin** if the name is "**-**". Certificates must be in DER format. Whenever the user specifies the **-f** option, the label of the private key matching the public key must also be specified. If the user does not provide the location of the private keystore, the default location will be used.

If the **-f** option is not specified, the invoker must provide the tag value and optional username for the certificate to be revoked. If invoked without the username parameter, the **certrevoke** command will use the name of the current user.

The **-l** option will be used to retrieve the private key matching the public key in the certificate that is to be revoked. The **certrevoke** command will fail if the user is unable to demonstrate the ownership of the private key matching the public key that is to be revoked. The **certrevoke** command will ask the user a password before actually performing the certificate revocation. The command may fail if it is unable to open **/dev/tty** for the current process.

Flags

Item	Description
-S *servicename*	Specifies which service module to use.
-f *file*	Specifies that the certificate to be revoked will be read form file.
-l *label*	Specifies the label associated with the private key of the certificate to be revoked.
-p *privatekeystore*	Specifies the location of the private keystore.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Security

This is a **setuid** command.

`Root` and invokers belonging to group security can revoke anybody's certificate. `Root` will revoke the the certificate using the revocation passphrase. Revocation passphrase is specified in the **/usr/lib/security/pki/acct.cfg** file.

A non-privileged user can only revoke certificates that they own. They have to demonstrate that they own the private key matching to the public key in the certificate to be revoked.

Audit

This command records the following event information:

CERT_Revoke <*username*>

Examples

To revoke the certificate `signcert` owned by Bob, enter:

```
$ certrevoke signcert bob
```

To revoke a certificate in file **cert.der**, enter:

```
$ certrevoke cert.der
```

Files

/usr/lib/security/pki/ca.cfg

certverify Command

Purpose

certverify verifies that the invoker is in possession of the private key for the specified certificate.

Syntax

certverify [**-S** *servicename*] tag [*user-name*]}

Description

The **certverify** command verifies that the user is in possession of the private key for the specified certificate. Once the system verifies that the user is in possession of the private key, a signature is created for this certificate and associated with the certificate. A certificate that has not gone through this verification process is considered untrustworthy by AIX.

The **-S** option specifies which end-entity services and libraries to use while verifying the certificate. Available services are defined in **/usr/lib/security/pki/ca.cfg**. When invoked without **-S** flag, **certverify** will use the default service, **local**. It is an error to specify a service name which does not have an entry in the **/usr/lib/security/pki/ca.cfg** file. The `tag` parameter uniquely selects one of the user's certificates. The `username` parameter specifies which AIX user is to be queried. The **certverify** command will issue a password prompt and request the user to enter the password of the keystore. The command may fail if it is unable to open **/dev/tty** for the current process.

Flags

Item	Description
-S *servicename*	Specifies which service module to use.

Exit Status

Item	Description
0	Successful completion.
>0	An error occured.

Security

This is a **setuid** command.

A user must prove that he has the possession of the private key matching the certificate he owns by knowing the password of the private keystore and the label that identifies the private key in the keystore.

`Root` and invokers belonging to group security are allowed to perform the verification operation, however, they can only successfully complete this operation if they have the knowledge of the label and the password to the keystore.

A non-privileged user is allowed to verify the possession of the private key only for the certificates they own.

Audit

This command records the following event information:

CERT_Verify <*username*>

Examples

To verify Bob's `cert1` certificate, enter:

```
$ certverify cert1 bob
```

Files

/usr/lib/security/pki/acct.cfg

cfgif Method

Purpose

Configures or activates one or all network interface (IF) instance(s) defined in the system configuration database.

Syntax

cfgif [**-l** *InterfaceInstance*]

Description

The **cfgif** method configures or activates one or all IF instance(s) of TCP/IP defined in the system configuration database. The **cfgif** method performs the following steps:

1. Retrieves the attributes associated with the Interface Program from the customized database. The attributes may include network address, network mask, security level and other related information.
2. Invokes the **ifconfig** command to load the IF instance using the customized attributes. The **ifconfig** command will load the appropriate interface program if it has not already been loaded.
3. Calls the **ifconfig** command to attach a routine to establish a path between the interface instance and the adapter.
4. Sets the status of a particular IF instance to "AVAILABLE" in the customized database. All the IF instances are set to "DEFINED" at system reboot. When the **cfgif** method is invoked during boot time or from the command line, the IF instance(s) are then made available.

Flags

Item	Description
-l *InterfaceInstance*	Specifies the interface instance to configure. If the instance name is specified, only that Interface instance is configured. If this flag is not used, all Interface instances in the defined state are configured.
-2	Indicates that **ifconfig** will be invoked from the second phase of IPL so that a hex value will be shown on the front panel display. This flag should not be used during runtime.

Examples

1. To configure a particular token-ring IF instance, enter the following command. Note that `tr0` is the logical name for the token-ring IF instance. It should be defined using the **defif** method.

```
cfgif -l tr0
```

2. To configure all IF instances, use the following command:

```
cfgif
```

cfginet Method

Purpose

Loads and configures an Internet instance and its associated IF instances.

Syntax

cfginet [**-2**]

Description

The **cfginet** method loads and configures an instance of TCP/IP (an Internet instance) by performing the following steps:

1. Loads the protocol code.
2. Initializes entries in the Address Family Domain switch table and in the Network Input switch table.
3. Sets the status flag of the Internet instance to AVAILABLE.
4. Invokes the **hostname** command and the **route** command to set the hostname and static routes. The hostname and routing data are retrieved from the configuration database.

> **Note:** The **cfginet** method is a programming tool and should not be executed from the command line.

Flag

Item	Description
-2	Specifies the second phase of IPL device configuration. A predetermined hex value will be displayed on the front panel. This option should not be used during regular run-time operation.

Example

To configure an Internet instance on a host, enter the method in the following format:

```
cfginet
```

cfgmgr Command

Purpose

Configures devices and optionally installs device software by running the programs specified in the Configuration Rules object class.

Syntax

cfgmgr [**-f** | **-s** | **-p** *Phase*] [**-i** *Device*] [**-u** *Drc Name* | **-l** *Name*] [**-v**]

cfgmgr [**-f** | **-s** | **-p** *Phase*] [**-i** *Device*] [[**-l** *Name* | **-u** *Drc Name*] **-c** *Connection*] [**-v**]

Description

The **cfgmgr** command configures devices and optionally installs device software into the system. The configurable devices are controlled by the Configuration Rules object class, which is part of the Device Configuration database. Each configuration rule specifies the following:

- The full path name of an executable program to run
- When to run the program (in relation to the other rules)
- In which phase to run the program

During system boot, the **cfgmgr** command configures all the devices that are necessary to use the system. System boot is a two-step process:

1. Called phase 1, this step begins when the kernel is brought into the system and the boot file system is initialized. During this phase, the **cfgmgr** command is invoked, specifying this as phase 1 by using the **-f** flag. The **cfgmgr** command runs all of the phase 1 configuration rules, which results in the base devices being configured.
2. Phase 2 execution begins, and the **cfgmgr** command is called with the **-s** flag.

The **cfgmgr** command recognizes three phases of configuration rules:

- Phase 1
- Phase 2 (second boot phase for normal boot)
- Phase 3 (second boot phase for service boot)

The **cfgmgr** command runs all of the rules for the phase specified during invocation (for example, phase 1 rules for the **-f** flag). However, if the **-l** flag is used, the **cfgmgr** command configures only the named device and its children.

If the **cfgmgr** command is invoked without a phase option (for example, without the **-f**,**-s**, or **-p** flags), then the command runs the phase 2 rules. The only way to run the phase 3 rules is with the **-p** flag.

The configuration rules for each phase are ordered based on the values specified in the seq field. This field is an integer that specifies the priority in which to run this rule, relative to the other rules for this phase. The higher the number specified by the seq field, the lower the priority. For example, a value of 1 specified in the seq field is executed before a rule with a value of 10. There is one exception: a seq field value of 0 implies a "don't care" condition, and runs last. Therefore, a seq field value of 1 is the highest priority and runs first.

If there are any devices detected that have no device software installed when configuring devices, the **cfgmgr** command returns a warning message with the name or a list of possible names for the device package that must be installed. If the specific name of the device package is determined, it is displayed as the only package name on a line below the warning message. If the specific name cannot be determined, a colon-separated list of possible package names is displayed on a single line. A package name or list of possible package names is displayed for each of the devices, if more than one device is detected without its device software.

The system displays the following warning message when devices without their device software are detected:

```
cfgmgr: 0514-621 WARNING: The following  device packages are
        required for device support but are not currently
        installed.
devices.pci.22100020
devices.pci.14101800
devices.pci.scsi:devices.pci.00100300:devices.pci.NCR.53C825
```

In this example, two devices missing software were found, and the **cfgmgr** command displays the names of the device packages that must be installed. A third device that is also missing software was found, but in this case, the **cfgmr** command displays several possible device package names.

When more than one possible package name is identified for a device, only one of the names will actually correspond to a device package on the installation medium. This is the package you must install. However, in some cases, more than one of the names will correspond to actual device packages on the installation medium. In this case, the first package name in the list for which there is a device package on the install medium is the package that must be installed. If the **cfgmgr** command is used with the **-i** flag, then the correct packages will be installed.

If you invoke the **cfgmgr** command with the **-i** flag, the command attempts to install device software automatically for each new detected device. The *Device* variable of the **-i** flag specifies where to find the installation medium. The installation medium can be a hardware device (such as a tape or diskette drive), a directory that contains installation images, or the installation image file itself.

> **Attention:** To protect the Configuration database, the **cfgmgr** command is not interruptible. Stopping this command before it is complete could result in a corrupted database.

Flags

Item	Description
-c *Connection*	Specifies the connection information required to configure the specific targeted device. See the Targeted configuration of FC and FCoE devices instructions regarding the connection information for a specific device.
-u *Drc name*	Specifies the *Drc name* variable of the Peripheral Component Interconnect (PCI) or virtual slot to configure along with the children of the slot. You can get the *Drc name* variable of the device by using the **lsslot** command.
-f	Specifies that the **cfgmgr** command runs the phase 1 configuration rules. This flag is not valid at run time (after system start).
-i *Device*	Specifies the location of the installation medium.
-l *Name*	Specifies the named device to configure along with the children of the device.
-p *Phase*	Specifies that the **cfgmgr** command runs the specified phase.
-s	Specifies that the **cfgmgr** command runs the phase 2 configuration rules.
-v	Specifies verbose output. The **cfgmgr** command writes information about what it is doing to standard output.

Configuration Rules

Item	Description
phase	Specifies whether this rule belongs to phase 1, phase 2, or phase 3 (second boot phase for service mode).
seq	Specifies the relative priority of this rule as an integer.
rule	A string containing the full path name of a program to execute. It can also contain any flags, but they must follow the program name as the whole string run as if it was typed on the command line.

Security

Access Control: Only the root user and members of the system group should have execute (x) access to this command.

Auditing Event:

Event	Information
DEV_Configure	Device name

Examples

These examples are based on the configuration rules containing the following information:

```
phase   seq        rule


1     10      /usr/lib/methods/defsys
1     12      /usr/lib/methods/deflvm


2     10      /usr/lib/methods/defsys
2     12      /usr/lib/methods/deflvm
2     13      /etc/methods/startusb
2     17      /etc/methods/cfgvlan -2
2     18      /usr/lib/methods/cfgrcnet
2     19      /usr/lib/methods/ptynode
2     20      /etc/methods/vconnode
```

```
2     20      /usr/lib/methods/startlft
2     22      /etc/methods/startrcm
2     25      /usr/lib/methods/starttty
2     27      /etc/methods/startsgio
2      0      /usr/lib/methods/defaio
2      0      /usr/lib/methods/def_posix_aio
2      0      /usr/lib/perf/cfg_perfstat load
2      0      /usr/lib/perf/load_blockset_ext


3     10      /usr/lib/methods/defsys
3     12      /usr/lib/methods/deflvm
3     13      /etc/methods/startusb
3     15      /usr/lib/methods/starttty
3     19      /usr/lib/methods/ptynode
3     20      /usr/lib/methods/startlft
3     20      /etc/methods/vconnode
3     22      /etc/methods/startrcm
3     27      /etc/methods/startsgio
```

1. When the **cfgmgr** command is invoked with the **-f** flag, the command gets all of the configuration rules with phase = 1 and runs them in the following order:

   ```
   /usr/lib/methods/defsys
   /usr/lib/methods/deflvm
   ```

   Note: The **-f** flag cannot be used during run time.

2. When the **cfgmgr** command is run with the **-s** flag, the command gets all of the configuration rules with phase = 2 and runs them in the following order:

   ```
   /usr/lib/methods/defsys
   /usr/lib/methods/deflvm
   /etc/methods/cfgvlan -2
   /usr/lib/methods/cfgrcnet
   /usr/lib/methods/ptynode
   /etc/methods/vconnode
   /usr/lib/methods/startlft
   /etc/methods/startrcm
   /usr/lib/methods/starttty
   /etc/methods/startsgio
   /usr/lib/methods/defaio
   /usr/lib/methods/def_posix_aio
   /usr/lib/perf/cfg_perfstat load
   /usr/lib/perf/load_blockset_ext
   ```

3. When the **cfgmgr** command is run with the **-p 3** flag, the command gets all of the configuration rules with phase = 3 and runs them in the following order:

   ```
   /usr/lib/methods/defsys
   /usr/lib/methods/deflvm
   /etc/methods/startusb
   /usr/lib/methods/starttty
   /usr/lib/methods/ptynode
   /usr/lib/methods/startlft
   /etc/methods/vconnode
   /etc/methods/startrcm
   /etc/methods/startsgio
   ```

4. If the **cfgmgr** command is run without a flag, the command functions the same as when used with the -s flag. Thus, the phase 2 rules are run in the the following order:

   ```
   /usr/lib/methods/defsys
   /usr/lib/methods/deflvm
   /etc/methods/cfgvlan -2
   /usr/lib/methods/cfgrcnet
   /usr/lib/methods/ptynode
   /etc/methods/vconnode
   /usr/lib/methods/startlft
   /etc/methods/startrcm
   /usr/lib/methods/starttty
   /etc/methods/startsgio
   /usr/lib/methods/defaio
   /usr/lib/methods/def_posix_aio
   ```

```
/usr/lib/perf/cfg_perfstat load
/usr/lib/perf/load_blockset_ext
```

5. To configure detected devices attached to the `scsi0` adapter, type the following:

```
cfgmgr -l scsi0
```

6. To configure the child device of the `fscsi0` adapter that is attached to the connection specified by the **-c** flag, type the following:

```
cfgmgr -l fscsi0 -c "ww_name=0x5001738000330191,lun_id=0x1000000000000"
```

7. To install device software automatically during configuration with the software contained in the `/usr/sys/inst.images` directory, type the following:

```
cfgmgr -i /usr/sys/inst.images
```

Files

Item	Description
/usr/sbin/cfgmgr	Specifies the command file.
/usr/include/sys/cfgdb.h	Contains numeric representations for fields in the Configuration Rules object class.

cfgqos Method

Purpose

Loads, configures, and activates the Quality of Service (QoS) instance.

Syntax

cfgqos

Description

The **cfgqos** method enables Quality of Service (QoS) for the TCP/IP protocol suite on a host by performing the following steps:

1. Loads the QoS kernel exension
2. Initializes the QoS instance
3. Attaches to the TCP/IP instance

> **Note:** The **cfgqos** method is a programming tool and is not intended to be invoked from the command line.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

To configure QoS on a host, use the following format:

```
cfgqos
```

cfgvsd Command

Purpose

`cfgvsd` – Configures a virtual shared disk.

Syntax

`cfgvsd` {`-a` | *vsd_name* ...}

Description

Use this command to configure the already defined virtual shared disks and bring them to the stopped state. This command does not make the virtual shared disk available.

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

You can use the System Management Interface Tool (SMIT) to run the `cfgvsd` command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Configure a virtual shared disk** option.

Flags

-a
Specifies all virtual shared disks that have been defined.

Parameters

vsd_name
Specifies a defined virtual shared disk.

Security

You must have root authority to run this command.

Restrictions

Under normal circumstances, you should not issue this command. The RVSD subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Examples

To bring the virtual shared disk `vsd1vg1n1` from the defined state to the stopped state, enter:

```
cfgvsd vsd1vg1n1
```

Location

`/opt/rsct/vsd/bin/cfgvsd`

cflow Command

Purpose

Generates a C and C++ flow graph of external references.

Syntax

cflow [**-d** *Number*] [**-I** *Directory*] [**-i _**] [**-i p**] [**-i x**] [**-q***Option*] [**-r**] [**-MA**] [**-U** *Name*] [**-Nd***Number*] [**-Nl***Number*] [**-Nn***Number*] [**-Nt***Number*] [**-D** *Name*[**=***Definition*]] *File ...*

Description

The **cflow** command analyzes the C, C++, **yacc**, **lex,** assembler, and object files and writes a chart of their external references to standard output.

> **Note:** Processing of C++ language files by the **cflow** command requires the presence of the IBM C Set++ Compiler/6000 package.

The **cflow** command sends files with the **.y**, **.l**, and **.c** suffixes to the **yacc** command, **lex** command, and **cpp** command for processing. A modified first pass of the **lint** command then processes the **yacc**, **lex**, and **cpp** output, or any **.i** files. The **cflow** command sends files with a **.C** suffix to the C Set++ compiler.

The **cflow** command assembles files with the **.s** suffix, extracting information from the symbol table (as it does with **.o** files). From this output, the **cflow** command produces a graph of external references and writes it to standard output.

Each line of output provides the following information (in order from left to right):

- A line number followed by sufficient tabs to indicate the level of nesting
- The name of the global, a colon, and its definition.

The name is normally a function not defined as external and not beginning with an underline character (see the **-i_** and **-i** inclusion flags).

For information extracted from C and C++ source files, the definition consists of an abstract type declaration (for example, char *), the name of the source file surrounded by angle brackets, and the line number on which the definition was found. Definitions extracted from object files contain the file name and location counter under which the symbol appeared, such as **.text** or **.data**. The **cflow** command deletes leading underline characters in C-style external names.

Once the **cflow** command displays a name, later references to the name contain only the **cflow** line number where the definition can be found. For undefined references, **cflow** displays only < > (angled brackets).

If the nesting level becomes too deep to display in available space, pipe the output from the **cflow** command to the **pr** command, using the **-e** flag to compress the tab expansion to less than eight spaces per tab stop.

> **Note:** To ensure that the line numbers produced by the **cflow** command match your **lex** and **yacc** files, you must send the **.l** or **.y** file to the **cflow** command.

Flags

Item	Description
-d *Number*	Sets to a decimal integer the depth at which the flow graph is cut off. By default this is a large number. Do not set the cutoff depth to a nonpositive integer.

Item	Description
-i _	Includes names that begin with an underline character. The default excludes these functions (and corresponding data if the **-ix** flag is used).
-i p	Disables ANSI function prototypes. The default option is to fill in undefined function information with available prototype declarations.
-i x	Includes external and static data symbols. The default includes only functions.
-r	Produces an inverted listing that shows the callers of each function, sorted by called function.
-MA	Specifies ANSI mode. The **cflow** command expects ANSI C code in this mode. The default mode of operation is extended mode.
-Nd*Number*	Changes the dimension table size to the *Number* parameter. The default value of *Number* is 2000.
-Nl*Number*	Changes the number of type nodes to the *Number* parameter. The default value of *Number* is 8000.
-Nn*Number*	Changes the symbol table size to the *Number* parameter. The default value of *Number* is 1500.
-Nt*Number*	Changes the number of tree nodes to the *Number* parameter. The default value of *Number* is 1000.

In addition, the **cflow** command recognizes the following flags of the **cpp** command (macro preprocessor):

Item	Description
-D *Name*[**=***Definition*]	Defines the *Name* parameter, as if by the **#define** statement. The default *Definition* is 1.
-q*Option*	Passes the **-q***Option* to the preprocessor. For example, `-qmbcs` sets multibyte mode specified by the current locale and `-qidirfirst` modifies the search order for files included with the **#include file_name** directive.
-**I** *Directory*	Adds the specified *Directory* to the list of directories in which the **cflow** program searches for **#include** files.
-U *Name*	Removes any initial definition of the *Name* parameter, where *Name* is a reserved symbol that is predefined by the particular preprocessor.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To generate a default flow graph of these C files that compose a program, enter:

```
cflow timeout.c kill.c error.c
```

2. To produce a **cflow** graph with a single level of nesting of functions, enter:

```
cflow -d1 resam.c pptp.c ptpt.c rrr.c whn.c
```

3. To generate a **cflow** graph of a **lex** program, enter:

```
cflow scan.l
```

4. To generate a **cflow** graph of the **yacc** program, enter:

```
cflow yaccfile.y
```

5. To generate an inverted listing showing the callers of each of the functions in the C files used in example 2, enter:

```
cflow -r resam.c pptp.c ptpt.c rrr.c whn.c
```

Files

Item	Description
/usr/ccs/bin/cflow	Driver for the **cflow** command
/usr/ccs/lib/cflow1	Executable for the **cflow** command
/usr/ccs/lib/dag	Executable for the **cflow** command
/usr/ccs/lib/flip	Executable for the **cflow** command
/usr/ccs/lib/lpfx	Executable for the **cflow** command
/usr/ccs/lib/nmf	Executable for the **cflow** command
/var/tmp/cf.*	Temporary files created by the **cflow** command

cfsadmin Command

Purpose

Administers disk space used for caching file systems with the Cache File-System (CacheFS).

Syntax

cfsadmin -c [-o ***param=****n* [,***param=****n*]] *cache_directory*

cfsadmin -d *cacheID*|**all** *cache_directory*

cfsadmin -l *cache_directory*

cfsadmin -s mntpnt . . .|**all**

cfsadmin -u *cache_directory*

Description

The **cfsadmin** command provides the following functions:

- Cache creation
- Deletion of cached file systems
- Listing of cache contents and statistics
- Resource parameter adjustment when the file system is unmounted.

For each form of the command, unless the **-u** flag is specified, you must specify a cache directory, that is, the directory under which the cache is actually stored. A path name in the front file system identifies the cache directory. When the **-s** flag is used, you must specify a mount point.

You can specify a cache ID when you mount a file system with CacheFS, or you can let the system generate one for you. The **-l** flag includes the cache ID in its listing of information. You must know the cache ID to delete a cached file system.

Flags

Item	Description
-c *cache_directory*	Creates a cache under the directory specified by *cache_directory*. This directory must not exist prior to cache creation.
-d	Removes the file system whose cache ID you specify and release its resources, or remove all file systems in the cache by specifying *cache_directory*. After deleting a file system from the cache, you must run the command to correct the resource counts for the cache.
-l *cache_directory*	Lists file systems stored in the specified cache, as well as statistics about them. Each cached file system is listed by cache ID. The statistics document resource utilization and cache resource parameters.
-o [***param=****n*]*cache_directory*	Allows changing parameter values by using "CacheFS Resource Parameters" on page 400 as arguments.
-s *cache_directory*	Requests a consistency check on the specified file system (or all **cachefs** mounted file systems). The **-s** flag only works if the cache file system was mounted with **demandconst** enabled. Each file in the specified cache file system is checked for consistency with its corresponding file in the back file system. The consistency check is performed file by file as files are accessed. If no files are accessed, no checks are performed. Using this flag does not result in a sudden storm of consistency checks. The **-s** flag is not currently supported in this operating systems CacheFS.
-u *cache_directory*	Updates resource parameters of the specified cache directory. Parameter values can only be increased. To decrease the values, you must remove the cache and recreate it. All file systems in the cache directory must be unmounted when you use this flag. Changes will take effect the next time you mount any file system in the specified cache directory. **Note:** The **-u** flag with no **-o** flag sets all parameters to their default values.

CacheFS Resource Parameters

You can specify the following cacheFS resource parameters as arguments to the **-o** flag. Separate multiple parameters with commas.

Item	Description
maxblocks=*n*	Maximum amount of storage space that CacheFS can use, expressed as a percentage of the total number of blocks in the front file system. If CacheFS does not have exclusive use of the front file system, there is no guarantee that all the space the **maxblocks** parameter allows will be available. The default is 90.

Item	Description
minblocks=*n*	The minimum amount of storage space, expressed as a percentage of the total number of blocks in the front file system, that CacheFS is always allowed to use without limitation by its internal control mechanisms. If CacheFS does not have exclusive use of the front file system, there is no guarantee that all the space the **minblocks** parameter attempts to reserve will be available. The default is 0.
threshblocks=*n*	A percentage of the total blocks in the front file system beyond which CacheFS cannot claim resources once its block usage has reached the level specified by **minblocks**. The default is 85.
maxfiles=*n*	Maximum number of files that CacheFS can use, expressed as a percentage of the total number of inodes in the front file system. If CacheFS does not have exclusive use of the front file system, there is no guarantee that all the inodes the **maxfiles** parameter allows will be available. The default is 90.
minfiles=*n*	Minimum number of files, expressed as a percentage of the total number of inodes in the front file system, that CacheFS is always allowed to use without limitation by its internal control mechanisms. If CacheFS does not have exclusive use of the front file system, there is no guarantee that all the inodes the **minfiles** parameter attempts to reserve will be available. The default is 0.
threshfiles=*n*	A percentage of the total inodes in the front file system beyond which CacheFS cannot claim inodes once its usage has reached the level specified by **minfiles**. The default is 85.
maxfilesize=*n*	Largest file size, expressed in megabytes, that CacheFS is allowed to cache. The default is -1, which means there is no limit on the largest file size.

Note: You cannot decrease the block or inode allotment for a cache. To decrease the size of a cache, you must remove it and create it again with different parameters.

Examples

1. To create a cache directory named **cache**, enter:

```
cfsadmin -c /cache
```

2. To create a cache directory named **/cache1** that can claim a maximum of 60 percent of the blocks in the front file system, can use 40 percent of the front file system blocks without interference by CacheFS internal control mechanisms, and has a threshold value of 50 percent. The threshold value indicates that after CacheFS reaches its guaranteed minimum, it cannot claim more space if 50 percent of the blocks in the front file system are already used.

```
cfsadmin -c -o maxblocks=60,minblocks=40,threshblocks=50 /cache1
```

3. To change the **maxfilesize** parameter for the cache directory **/cache2** to 2 megabytes, enter:

```
cfsadmin -u -o maxfilesize=2 /cache2
```

4. To list the contents of a cache directory named **/cache3** and provides statistics about resource utilization, enter:

```
cfsadmin -l /cache3
```

5. To remove the cached file system with cache ID 23 from the cache directory **/cache3** and free its resources (the cache ID is part of the information returned), enter:

```
cfsadmin -d 23 /cache3
```

6. To remove all cached file systems from the **/cache3** directory, enter:

```
cfsadmin -d all /cache3
```

7. To check all filesystems mounted with **demandconst** enabled for consistency. No errors will be reported if no **demandconst** filesystems were found. Enter:

```
cfsadmin
```

chargefee Command

Purpose

Charges end users for the computer resources they use.

Syntax

/usr/sbin/acct/chargefee *User Number*

Description

The **chargefee** command is used by someone with administrative authority to charge the individual specified by the *User* parameter for the number of work units specified by the *Number* parameter. The *Number* value can be an integer or a decimal value.

The **chargefee** command writes a record to the **/var/adm/fee** file. This information is merged with other accounting records by the **acctmerg** command to create the daily report.

> **Note:** You should not share accounting files among nodes in a distributed environment. Each node should have its own copy of the various accounting files.

Security

Access Control: This command should grant execute (x) access only to members of the adm group.

Examples

To charge `smith` for `10` units of work on a financial report, enter:

```
/usr/sbin/acct/chargefee smith 10
```

A record is created in the **/var/adm/fee** file, which the **acctmerg** command will merge with records in other accounting files to produce the daily report.

Files

Item	Description
/usr/sbin/acct	The path to the accounting commands.
/var/adm/fee	Accumulates the fees charged to each login name.

chauth Command

Purpose

Changes user-defined authorization attributes.

Syntax

chauth [**-R** *load_module*] *Attribute* **=** *Value* ... *Name*

Description

The **chauth** command modifies attributes for the authorization that is identified by the *Name* parameter. The command only modifies existing user-defined authorizations in the authorization database. System-defined authorizations cannot be modified with the **chauth** command. To change an attribute of a user-defined authorization, specify the attribute name and the new value with the *Attribute* **=** *Value* parameter. If any specified attribute or attribute value is not valid, the **chauth** command does not modify the authorization.

Important: Modifying the ID of an authorization can affect the system security because the current value of the ID might be used by some processes, files, and so on. In general, use the **id** attribute to modify the ID of an authorization when you are sure that the authorization is not used. The **chauth** command only allows the ID to be set to an unused value greater than 10 000. IDs less than 10 000 are reserved for system-defined authorizations.

If the system is configured to use multiple domains for the authorization database, authorization modification is performed according to the order specified by the **secorder** attribute of the authorizations database stanza in the **/etc/nscontrol.conf** file. Only the first matching authorization is modified. Duplicate authorizations from the remaining domains are not modified. Use the **-R** flag to modify the authorization from a specific domain.

When the system is operating in enhanced Role Based Access Control (RBAC) mode, modifications made to the authorization database are not used for security considerations until the database is sent to the kernel security tables through the **setkst** command.

Flags

Item	Description
-R *load_module*	Specifies the loadable module to use for the authorization modification.

Attributes

Item	Description
id	Specifies a unique integer that is used to identify the authorization. The value is a decimal integer ranging from 10 001 through 32 768.
dfltmsg	Specifies the default description to use if message catalogs are not in use. The value is a string.

Item	Description
msgcat	Specifies the message catalog file name containing the description of the authorization. If the **msgcat** attribute is specified, the **msgset** and **msgnum** attributes must also be specified. The value is a string. If the specified string contains a leading forward slash (/), the value is assumed to be an absolute path name. Otherwise, the user environment defines the directory search path as specified by the **catopen** routine.
msgset	Specifies the message set number in the file name to retrieve the message number. The file name is specified by the **msgcat** attribute, and the message number is specified by the **msgnum** attribute. The value is a decimal integer.
msgnum	Specifies the message number for the description of the authorization in the file and the set. The authorization is specified by the **msgcat** attribute, and the set number is specified by the **msgset** attribute. The value is a decimal integer.

Parameters

Item	Description
Name	Specifies the authorization to modify.

Security

The **chauth** command is a privileged command. You must assume a role that has the following authorization to run the command successfully.

Item	Description
aix.security.auth.change	Required to run the command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files Accessed

Item	Description
File	Mode
/etc/security/authorizations	rw

Examples

1. To change the message catalog used to provide the authorization description for the `custom` authorization, use the following command:

```
chauth msgcat="custom_auths.cat" custom
```

2. To change the message set and number that designates the authorization description for the `custom.test` authorization, use the following command:

```
chauth msgset=5 msgnum=24 custom.test
```

3. To change the message catalog for the `custom.test` authorization in LDAP, use the following command:

```
chauth -R LDAP msgset=5 custom.test
```

chauthent Command

Purpose

Changes the configured authentication methods for the system.

Syntax

chauthent [**-k5**] [**-k4**] [**-std**]

Description

The **chauthent** command sets the desired configuration based on the flags the user sets. The authentication methods are set in the order in which the flags are given to the command. If none of the flags are set, then the **rcmds** will be disabled from functioning. If the **-std** flag is set, it must be the last flag set or the command will fail.

> **Note:** The complete order of authentication methods must be specified each time. The command does not modify the current order when replacing it with the new one.

The user must have root authority to execute the command.

The **chauthent** command takes the flags set and calls the **set_auth_method** routine in **libauthm.a** to cause the change.

The **chauthent** command writes an error message to **stderr** and returns a -1 if **set_auth_method** fails.

Flags

Item	Description
-k5	Sets the Kerberos 5 authentication method.
-k4	Sets the Kerberos 4 authentication method.
-std	Sets the Standard operating system authentication method.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. Set all of the methods in descending order:

```
chauthent  -k5   -k4  -std
```

2. Set all of the methods with Kerberos 4 attempted first:

```
chauthent -k4 -k5 -std
```

3. Clear all of the methods:

```
chauthent
```

chC2admin Command

Purpose

Changes the name of the administrative host for a system.

Syntax

chC2admin [**-a** *address*] *hostname*

Description

The **chC2admin** command maintains the name of the C2 System Administrative Host as well as the NFS mount points and hostname entries as defined in **/etc/filesystems**.

Changing the name of the Administrative Host will cause the NFS file systems listed in **/etc/filesystems** to be updated and the contents of **/etc/security/admin_host** to be replaced.

The given *hostname* must be defined when this command is executed. If *hostname* cannot be resolved, a warning will be given. The **-a** option may be used to specify the IP address of hostname. When the **-a** option is given, *hostname* and *address* will be added to the **/etc/hosts** file.

Flags

Item	Description
-a *address*	

Parameters

Item	Description
hostname	Specifies the hostname.

Exit Status

0
All updates have been made successfully.

1
Command has been executed on a non-C2 System.

2
Command failed while changing the administrative host.

Files

Item	Description
/usr/sbin/chC2admin	Contains the **chC2admin** commnad.

chCCadmin Command

Purpose

Changes the name of the Common Criteria enabled System Administrative Host for a system.

Syntax

chCCadmin [**-a** *address*] *hostname*

Description

The **chCCadmin** command maintains the name of the Common Criteria enabled System Administrative Host as well as the NFS mount points and hostname entries as defined in **/etc/filesystems**.

Changing the name of the Administrative Host will cause the NFS file systems listed in **/etc/filesystems** to be updated and the contents of **/etc/security/admin_host** to be replaced.

The given *hostname* must be defined when this command is executed. If *hostname* cannot be resolved, a warning will be given. The **-a** option may be used to specify the IP address of hostname. When the **-a** option is given, *hostname* and *address* will be added to the **/etc/hosts** file.

Flags

Item	Description
-a *address*	

Parameters

Item	Description
hostname	Specifies the hostname.

Exit Status

0
All updates have been made successfully.

1
Command has been executed on a non-Common Criteria enabled System.

2
Command failed while changing the administrative host.

Files

Item	Description
/usr/sbin/chCCadmin	Contains the **chCCadmin** commnad.

chcifscred Command

Purpose

Changes the password for a specific server/user entry stored in the `/etc/cifs_fs/cifscred` file.

Syntax

`chcifscred` `-h` *RemoteHost* `-u` *user* [`-p` *password*]

Description

The `chcifscred` command takes a server and user name as input. If this input has credentials listed in `/etc/cifs_fs/cifscred`, the command line prompts for a password to replace the existing password. The password is stored in an encrypted format.

Flags

Item	Description
-h *RemoteHost*	Specifies the name of the remote host (CIFS server). This can be provided as a host name, an IP address, or as a fully qualified domain name.
-p *password*	Specifies the new password for the specified user on the specified remote host.
-u *user*	Specifies the user name whose password is changing for access to the specified host.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Examples

1. To change the password stored for user1 to mount on server1, with server1 and user1 credentials already residing in /etc/cifs_fs/cifscred, enter:

```
chcifscred -h server1 -u user1
```

Location

/usr/sbin/chcifscred

Files

Item	Description
/etc/cifs_fs/cifscred	Stores the CIFS credentials.

chcifsmnt Command

Purpose

Changes the mount options, server name, share, or credentials for a CIFS mount.

Syntax

chcifsmnt -f *MountPoint* [-d *RemoteShare*] [-h *RemoteHost*] [-c *user*] [-p *password*] [-m *MountTypeName*] [-A|-a] [-I|-B|-N] [-t {rw|ro}] [-u *uid*] [-g *gid*] [-x *fmode*] [-w *wrkgrp*]

Description

The chcifsmnt command changes the mount options, server name, share name, or credentials for a CIFS mount defined in /etc/filesystems file. If the share is not mounted, it will be mounted after the changes to the /etc/filesystems file are made. If the share is not already defined in /etc/filesystems, an error is returned.

Flags

Item	Description
`-a`	Specifies that the `/etc/filesystems` entry for this file system should not be automatically mounted at system restart. This is the default.
`-A`	Specifies that the `/etc/filesystems` entry for this file system should be automatically mounted at system restart.
`-B`	Specifies that the `/etc/filesystems` entry should be modified and that it should be remounted with the options specified. This is the default.
`-c` *user*	Specifies user name used to gain access to the CIFS share.
`-d` *RemoteShare*	Specifies the share name on the CIFS server that should be mounted.
`-f` *MountPoint*	Specifies the path name over which the CIFS share should be mounted.
`-g` *gid*	Specifies the GID that is assigned to files in the mount. The default is 0.
`-h` *RemoteHost*	Specifies the name of the remote host (CIFS server). This can be provided as a host name, an IP address, or as a fully qualified domain name.
`-I`	Specifies that the `/etc/filesystems` entry should be modified, but should not be remounted.
`-m` *MountTypeName*	Defines the mount type that will be added to the `/etc/filesystems` file, which allows for mounting all file systems of a specific type using the `-t` option of the `mount` command. By default, no type value will be added to `/etc/filesystems`.
`-N`	Remounts the CIFS share with the options specified, but does not modify the `/etc/filesystems` file.
`-p` *password*	Specifies the password used to grant access to the specific user on the specific server. The specific credentials (server/user/password) are added to the `cifscred` file (the password will be encrypted). If the `-p` option is not specified, and the credentials do not already exist in the `cifscred` file, the command line prompts the user to provide the password, and the credentials will be added to the `cifscred` file. If the server/user credentials already exist in the `cifscred` file, this option is ignored, and the existing credentials are used for mounting.
`-t` {rw|ro}	Specifies whether file system should be mounted as read-only. The default is read-write (rw).

Item	Description
`-u` *uid*	Specifies the UID that is assigned to files in the mount. The default is 0.
`-x` *fmode*	Specifies the owner, group, and other permission bits assigned to files in the mount. The default is 755.
`-w` *wrkgrp*	Specifies the domain that should be used to authenticate the user during mount. If this option is not used, authentication is handled locally by the CIFS server.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Examples

1. To change the user name to `user1` for a CIFS mount defined on `/mnt`, enter:

```
chcifsmnt -f /mnt -c user1
```

Location

`/usr/sbin/chcifsmnt`

Files

Item	Description
`/etc/cifs_fs/cifscred`	Stores the CIFS credentials.
`/etc/filesystems`	Stores the CIFS entry.

chclass Command

Purpose

Change the attributes and resource entitlements of a Workload Management class.

Syntax

chclass -a *Attribute=Value* {[**-a** *Attribute=Value*]...} [**-c** | **-m** | **-b** | -v | **-C** | **-B** | **-P** | **-T** | **-L** | -V | **-A** *KeyWord=Value*] [**-d** *Config_Dir*] [**-S** *SuperClass*] *Name*

Description

The **chclass** command changes attributes for the class identified by the *Name* parameter. The class must already exist. To change an attribute, specify the attribute name and the new value with the *Attribute=Value* parameter. To change a limit or shares value, use option **-c** for cpu, **-m** for memory, and **-b** for disk I/O throughput, with the keyword value in **min**, **softmax**, **hardmax** or **shares**. To set the process total limits (the limits that apply to each process of the class), use one or more of the options **-C** (totalCPU), **-B** (totalDiskIO), **-A** (totalConnectTime), or -v (totalVirtualMemoryLimit), with the keyword

value of `hardmax`. To set the class total limits (the limits that apply to the whole class), use one or more of the options **-P** (totalProcesses), **-T** (totalThreads), **-L** (totalLogins), or `-V` (totalVirtualMemoryLimit), with the keyword value of `hardmax`. To reset any total limit, use `-` for *Value*. Process, class, or both total limits may be disabled when starting or updating the WLM (see **wlmcntrl** command).

> **Note:** Only the root user can change the attributes of a superclass. Only root or authorized users whose user ID or group ID matches the user name or group name specified in the attributes **adminuser** and **admingroup** of a superclass can change the attributes of a subclass of this superclass.

Normally, **chclass** updates the attributes of a class in the relevant WLM property files, and the modifications are applied to the in-core class definition (active classes) only after an update of WLM using the **wlmcntrl** command.

If an empty string is passed as the configuration name (*Config_dir*) with the **-d** flag, the change applies only to the in-core class attributes, and no property file is updated, making the changes temporary (the change is lost if WLM is stopped and restarted or the system is rebooted).

Note: This command cannot apply to a set of time-based configurations (do not specify a set with the **-d** flag). If the current configuration is a set, the **-d** flag must be given to indicate which regular configuration the command should apply to.

Attributes

The following attributes can be changed:

Class properties:

Item	Description
tier	Specifies the **tier** value. The **tier** value for a class is the position of the class in the hierarchy of resource limitation desirability for all classes. A class with a lower **tier** value is more favored. The **tier** value ranges from 0 through 9 (the default is 0).
inheritance	If the **inheritance** attribute is set to **yes**, the children of processes in this class remain in the class upon **exec** regardless of the automatic assignment rules in effect. If the **inheritance** attribute is set to **no**, the assignment rules apply normally. The default if not specified is **no**.
localshm	Indicates whether memory segments that are accessed by processes in different classes remain local to the class they were initially assigned to or if they go to the **Shared** class. You can specify a value of **Yes** or **No**. If not specified, the default is **No**.
authuser	Specifies the user name of the user who is allowed to assign processes to this class. The default when the attribute is not specified is **root**.
authgroup	Specifies the group name of the group of users that is allowed to assign processes to this class. There is no default value.
rset	Specifies the name of a resource set that the processes in the class have access to. By default, the class has access to all resources on the system.
vmenforce	Specifies whether all processes or only the offending processes in the class need to be terminated when the class hits the maximum VM limit. You can specify the value of `class` or `proc`. The default value is `proc`.
delshm	Specifies whether the shared segments will be deleted when the last process referencing them ends because virtual memory is exceeded. You can specify the value of `yes` or `no`. The default value is `no`.

Item	Description
adminuser	Specifies the user name of the user who is allowed to administer the subclasses of this superclass. This attribute is valid only for superclasses. The default, when the attribute is not specified, is a null string, and in this case, only **root** users can administer the subclasses. **Note:** If the **adminuser** or **admingroup** attribute is changed for a superclass that belongs to the running configuration (or to a configuration of the running set), a global WLM update should be performed to reflect these changes to the in-core configuration, elsewhere, updates that are restricted to superclass by such a user might fail due to lack of authority.
admingroup	Specifies the group name of the group of users that is allowed to administer the subclasses of this superclass. This attribute is valid only for superclasses. The default value, when the attribute is not specified, is a null string, meaning that no group can administer the subclasses. **Note:** If the **adminuser** or **admingroup** attribute is changed for a superclass that belongs to the running configuration (or to a configuration of the running set), a global WLM update should be performed to reflect these changes to the in-core configuration, elsewhere, updates that are restricted to superclass by such a user might fail due to lack of authority.
iopriority	Specifies the priority that is assigned to I/O requests. The I/O requests are issued by the threads that are classified to the class. The priority is used to prioritize I/O buffers at the device level. If the storage device does not support I/O priorities, the priority is ignored. Valid I/O priority values range from 0 through 15.

Class limits and shares for CPU, memory, or disk I/O resource:

Item	Description
min	Specifies the minimum percentage of the resource that must be made available when requested, expressed as a percentage of the total resource available in the system. Possible values range from 0 through 100 (the default is 0).
shares	Specifies the maximum ratio of the resource that can be made available if there is contention. This parameter is expressed in shares of the total resource available in the system. The actual ratio of the resource is dynamically computed, proportionally to the shares of all active classes. If a class has no running process, its shares are excluded from the computation. The shares are arbitrary numbers ranging from 1 through 65535. If shares is specified as a hyphen (-), the class is always considered on target and its utilization for this resource is not regulated by WLM, but the minimum and maximum limits if any still apply. This is the default if the shares for a resource are not specified.
softmax	Specifies the maximum percentage of the resource that can be made available, when there is contention. Possible values range from 1 through 100 (the default is 100). A class can exceed its soft maximum for a given resource if there is no contention on the resource.
hardmax	Specifies the maximum percentage of the resource that can be made available, even if there is no contention. Possible values range from 1 through 100 (the default is 100). Specifying a value different from the default value of 100 for memory can result in some memory pages remaining unused, while some processes in the class use more.
max	Specifies the maximum percentage of the resource that can be made available, even if there is no contention. Possible values range from 1 through 100 (the default is 100). Specifying a value different from the default value of 100 for memory can result in some memory pages remaining unused, while some processes in the class use more.

Note: The default values for a class can be read using the **lsclass -D** command and can be changed by manually editing the property files **classes**, **shares**, or **limits** to add a default stanza. For more information about these files, see the *Files Reference*.

Class description:

Item	Description
description	The class description text can be composed of any ASCII character, except colons (**:**) and commas (**,**).

Flags

Item	Description
-A hardmax=***Value***	Sets the maximum amount of time a login session in the class can stay active. Value is specified as an integer, possibly appending the unit (s for seconds, m for minutes, h for hours, d for days, and w for weeks, default is seconds). As a user approaches this connection time limit, WLM will send a warning message to the session terminal. When the limit is reached, the user will be notified and the session leader will be sent the **SIGTERM** signal, and after a short grace period, the session will be terminated (**SIGKILL**).
-b *KeyWord***=***Value*	Changes a limit or shares value for disk I/O throughput. Possible *KeyWord*s are **min**, **softmax**, **hardmax**, or **shares**.
-B hardmax=*Value*	Sets the total amount of disk I/Os allowed for each process in the class. Value is specified as an integer, possibly appending the unit (KB for kilobytes, MB for megabytes, TB for terabytes, PB for petabytes, and EB for exabytes, default is kilobytes). After a process has used this amount of disk I/Os, the process will be sent the **SIGTERM** signal, and after a grace period, it will be killed (**SIGKILL**).
-c *KeyWord***=***Value*	Changes a limit or shares value for a CPU. Possible *KeyWord*s are **min**, **softmax**, **hardmax**, or **shares**.
-C hardmax=*Value*	Sets the total amount of CPU time allowed for each process in the class. Value is specified as an integer, possibly appending the unit (s for seconds, m for minutes, h for hours, d for days, and w for weeks, default is seconds). After a process has used this amount of time, the process will be sent the **SIGTERM** signal, and after a grace period, it will be killed (**SIGKILL**).
-d *Config_Dir*	Uses the **/etc/wlm/***Config_Dir* directory as alternate directory for the properties files. If this flag is not present, the current configuration files in the directory pointed to by **/etc/wlm/current** are used. If an empty string is passed as the configuration name (`-d ""`) the modifications only affect the in-core class definition and no configuration file is modified.
-L hardmax=*Value*	Sets the total number of login sessions simultaneously available in the class. If a user tries to log onto the system and the login shell would end up in a class that has reached the total logins limit, the login operation will fail.
-m *KeyWord***=***Value*	Changes a limit or shares value for memory. Possible *KeyWord*s are **min**, **softmax**, **hardmax**, or **shares**.
-P hardmax=*Value*	Sets the maximum number of processes allowed in the class. If an operation would result in a new process entering the class when the class has this many processes in it, the operation will fail.

Item	Description
-S *SuperClass*	Specifies the name of the superclass when changing the attributes of a subclass. There are two ways of specifying that the change is to be applied to the subclass Sub of superclass Super: 1. Specify the full name of the subclass as Super.Sub and not use -S. 2. Uses the -S flag to give the superclass name and use the short name for the subclass: `chclass options -S Super  Sub`
-T hardmax=*Value*	Sets the maximum number of threads allowed in the class. If an operation would result in a new thread entering the class when the class has this many processes in it, the operation will fail. The total thread limit must be at least as large as the total process limit for a class. If a class has a total thread limit but no total process limit specified, the total process limit will be set to the total thread limit.
`-v hardmax=`*Value*	Specifies the virtual memory limit allowed per process in the specified class. The maximum amount of virtual memory allowed per process is (2^31)-1 for 32-bit kernels and (2^63)-1 for 64-bit kernels.
`-V hardmax=`*Value*	Specifies the virtual memory allowed for the specified class. The maximum amount of virtual memory allowed per process is (2^31)-1 for 32-bit kernels and (2^63)-1 for 64-bit kernels.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
classes	Contains the names and definitions of the classes.
limits	Contains the resource limits enforced on the classes.
shares	Contains the resource shares attributes for each class.

chcluster Command

Purpose

Changes the cluster configuration.

Syntax

To change the local site cluster topology through addition or deletion of node entities, shared disk entities, or backup repository entities, use the following syntax:

```
chcluster [ -m [+|-]nodename{[cle_ip=addr1[,cle_ip=addr2][,...],cle_uuid=UUID,cle_globid=id}
[,...] ]
                [ -d [+|-]shareddisk[,...] ] [ -b [+|-]backupdisk[,...] ] [-n cluster_name ][-
p [+|-]comdisk] [-v]
```

To extend the cluster topology to include a remote site, use the following syntax:

```
chcluster -S sitename -r remote_reposdisk -m nodename [-s multi_cast_addr] [-n cluster_name ]
[-v]
```

To change the remote site cluster topology through addition or deletion of node entities, use the following syntax:

```
chcluster -S sitename -m [+|-]nodename [-n cluster_name] [-v]
```

To change the site cluster topology by adding or removing backup repository entities, use the following syntax:

```
chcluster  -b [+|-]backupdisk[,...]  [-@ contact_node -S sitename] [-n cluster_name] [-v]
```

To change the attribute on a site entity, use the following syntax:

```
chcluster -S sitename{[cle_name=new_name,cle_prio=prio]}
```

To change the attribute of a node entity, use the following syntax:

```
chcluster -m nodename{[[+|-]cle_ip=addr1[,cle_ip=addr2][,...] | cle_hostname=name]} [ -S
sitename ] [-v]
```

Description

The **chcluster** command changes the cluster configuration.

The **chcluster** command adds and removes the storage area network (SAN) shared disks and nodes to or from the cluster configuration, or extends the existing cluster to span multiple sites. When you create another site, specify only one remote node, along with the remote site name, remote repository disk name, and the remote site multicast address (optional). Additional remote nodes can be added after the remote site is created.

Flags

Item	Description
-@ *at_node*	Specifies the node where the disk universal unique identifier (UUID) is located. The node must be reachable and within the same site that the disk is located. It is optional for local site changes but required for remote site changes. This flag applies only to the disks that are specified by the **-b** flag.
-b [+|-] *backupdisk* [,...]	Specifies a comma-separated list of SAN shared storage device such as hdisk5, hdisk6. These disks are used as the backup for the central repository of the cluster. When the central repository is inaccessible, the disk from the list is used as a replacement. These devices must be accessible from all nodes in the site.
-d [+|-] *shareddisk* [,....]	Specifies a comma-separated list of shared storage device names to be added to or removed from the cluster configuration. The shared disks must not be open when the **chcluster** command is run. The disk must be in the local site of the node that is running the command.

Item	Description
-m [+|-]*node*[,...]	Specifies a comma-separated list of node names to be added to or removed from the cluster configuration. The following node information can be specified only when a node is added to the cluster: **cle_uuid** Specifies the node UUID that is used if the note is unique across the cluster. If the node UUID is not specified, it is automatically generated. **cle_globid** Specifies the short ID of the node that must be a unique unsigned number. The value must be greater than zero. If the short ID is not specified, it is automatically generated. The following node attributes can be specified with any arguments: **cle_ip** Specifies the gateway address of the node (in case the cluster spans across multiple sites). Typically, this attribute is the address through which the node can be reached from an external node. This attribute can be specified in either an IPv4 or IPv6 format. If a new node is added to the cluster by specifying the **+** sign and additional values, the node is added to the cluster with the specified values. If an existing node is specified with the **+** sign and additional attributes, the new attributes are added to the node. If an existing node is specified with the **-** sign and additional attributes, the specified attributes are deleted from the node. **cle_hostname** Specifies the new host name of the node.
-n *name*	Specifies the name of the cluster that needs to be changed. If this flag is omitted, the default cluster is used.
-p [+|-] *comdisk*	Specifies a SAN shared storage device such as `hdisk5` and `hdisk6`. These disks are used by the shared storage pool cluster for inter-node communication when the network is down. If you specify the plus sign (**+**) or if you do not specify any sign, the storage device is added to the shared storage pool cluster. If you specify the minus sign (**-**), the storage device is removed from the shared storage pool cluster.
-v	Specifies verbose mode.
-r *+remote_reposdisk*	Specifies the name of the remote disk that is used as the repository of the remote site, as seen on the first remote node. This flag is used only for remote site creation.
-s *+multi_cast_addr*	Specifies the multicast address that is used for the remote site. If this flag is omitted, a default multicast address is generated.

Item	Description
-S *+sitename*	Specifies the name of the site that is associated with the specified entities. Currently, a cluster supports only 2 sites. If this flag is omitted, the site of the running node is used. The following site information can be specified only during site creation: **cle_uuid** Specifies the site UUID that is used if the node is unique across the cluster. If the site UUID is not specified, it is automatically generated. **cle_globid** The short ID of the site that must be a unique unsigned number. The value must be greater than zero. If short ID is not specified, it is automatically generated. The following site attribute can be specified during site creation: **cle_prio** Specifies the priority of a site. A lower value indicates a higher priority. The priority is mainly used in the context of synchronizing the repository metadata. If two sites split and the repository data becomes out-of-sync, the data from the site with higher priority is copied over to the site with lower priority. If a site already exists, the following attributes can be changed: **cle_name** Specifies the new name of the site. **cle_prio** Specifies the new priority of the site. The other values cannot be changed.

Examples

1. To add shared disks to the cluster configuration:

   ```
   chcluster -n mycluster -d +hdisk20,+hdisk21
   ```

2. To remove shared disks from the cluster configuration:

   ```
    chcluster -n mycluster -d -hdisk20,-hdisk21
   ```

3. To add nodes to the cluster configuration:

   ```
   chcluster -n mycluster -m +nodeD,+nodeE
   ```

4. To remove nodes from the cluster configuration:

   ```
   chcluster -n mycluster -m -nodeD,-nodeE
   ```

5. To add a site to the cluster configuration:

   ```
   chcluster -n mycluster -S +remotesite -m +nodeZ -r +hdisk5
   ```

   where *hdisk5* is the name of the disk as seen by *nodeZ* node.

6. To change the name of an existing site:

   ```
   chcluster -n mycluster -S remotesite{cle_name=myremotesite}
   ```

7. To change the name of an existing node in the cluster:

```
chcluster -n dynamicCluster –m rosy{cle_hostname=pinky}
```

8. To add the backup disks hdisk1 and hdisk2 to the local site, enter the following command:

```
chcluster -S Local -b +hdisk1,+hdisk2
```

9. To remove the backup disks hdisk1 and hdisk2 from the local site, enter the following command:

```
chcluster -S Local -b -hdisk1,-hdisk2
```

10. To add the backup disks hdisk3 and hdisk4 to the remote site, enter the following command:

```
chcluster -S Remote -b +hdisk3,+hdisk4 -@ remote_node
```

11. To remove the backup disks hdisk3 and hdisk4 from the remote site, enter the following command:

```
chcluster -S Remote -b -hdisk3,-hdisk4 -@ remote_node
```

chcod Command

Purpose

Manages Capacity Upgrade on Demand.

Syntax

chcod [**-r** *ResourceType* **-n** *NbrResources*] [**-c** *CustomerInfo*] [**-m** *MailAddr*] [**-h**]

Description

The **chcod** command manages Capacity Upgrade on Demand, or CUoD. CUoD enables the authorization of more *ResourceTypes*, such as processors, on the system than were initially authorized. The additional resources may be enabled if they are available, and if the system supports CUoD for that *ResourceType*. Only one *ResourceType* may be managed at a time. The change in the number of *ResourceTypes* takes effect after the next system boot. CUoD management also includes displaying the current number of *ResourceType*(s) that have CUoD support, monitoring the number of *ResourceType*(s) on the system, and notifying appropriately. Notification occurs on a monthly basis and also whenever *NbrResources* changes.

Notification takes the form of error logging and, optionally, sending e-mail. An entry is made in the system error log whenever the specified *ResourceType* changes and also on a monthly basis. The *CustomerInfo* text is included in the error log. If you specify an e-mail address with *MailAddr*, notification also occurs through an e-mail message sent to *MailAddr*. The *CustomerInfo* text is included in the text of the message. You can have notification by both error logging and e-mail if you specify both *CustomerInfo* and *MailAddr*.

With no flags specified, **chcod** displays the current value of *CustomerInfo*, *MailAddr*, the system's model name and serial number, and the current value(s) of *NbrResources* for any *ResourceType* that has CUoD support.

Note: Beginning with the IBM p650 and later models (all POWER4 Systems), CUoD is managed at the Hardware Management Console (HMC).

Flags

Item	Description
-c *CustomerInfo*	Specifies the text string to include in the error log. This string is also included in the body of any e-mail message sent. *CustomerInfo* may not be more than 255 characters. Blanks may not be included in the string. After*CustomerInfo* has been specified, subsequent **chcod** uses do not have to specify the **-c** flag, but you do have the option of changing it. *CustomerInfo* may consist of alphanumeric characters and any of **.** (period), **,** (comma), **-** (hyphen), **(** (open parenthesis), or **)** (close parenthesis).
-h	Displays the usage message.
-m *MailAddr*	Specifies the e-mail address to which e-mail should be sent. *MailAddr* may not be more than 255 characters. If *MailAddr* is reset by specifying "" (a blank string), then only error logging will monitor the resources that have CUoD support. You must have e-mail configured on your system if you want to send notification to this e-mail address.
-n *NbrResources*	Specifies the number of *ResourceTypes* to be authorized on the system. It must be zero or greater. If it is 0, CUoD is disabled for the specified *ResourceType*. If **-n** is specified, then **-r** must also be specified.
-r *ResourceType*	Specifies the *ResourceType*, for example, `proc` for processors, to be enabled and monitored on the system. The system must support CUoD for *ResourceType*. If **-r** is specified, then **-n** must also be specified.

Examples

1. To initiate CUoD for processors, type:

   ```
   chcod -r proc -n 10 -m"someone@ibm.location.com" -c"Jane_Doe-
   Customer_Number_999999-(111)111-1111"
   ```

2. To change the *CustomerInfo*, type:

   ```
   chcod -c"Jane_Doe-Customer_Number_999999-(222)222-2222"
   ```

3. To stop the e-mail form of notification, type:

   ```
   chcod -m""
   ```

4. To see the current values of the resources with CUoD support, type:

   ```
   chcod
   ```

   A message similar to the following will be displayed:

   ```
   Current CustomerInfo = Jane_Doe-Customer_Number_999999-(222)222-2222
   Current MailAddr = someone@ibm.location.com
   Current model and serial number = IBM,7043-150 000974934C00
   Current number of authorized processors = 10 of 12 installed on system
   ```

chcomg Command

Purpose

Changes a previously-defined communication group for a peer domain.

Syntax

To change an attribute of a communication group:

chcomg [-s *sensitivity*] [-p *period*] [**-g** *grace*] [-t *priority*] [-b] [-r] [-x b | r | br] [-e *NIM_path*] [-m *NIM_parameters*] [**-N** *UseForNodeMembership*] [-h] [-TV] *communication_group*

To change a reference in a heartbeat interface resource to a different communication group:

chcomg [**-i h:***heartbeat_interface1*[**:***node1*] [**,***heartbeat_interface2*[**:***node2*]...] | **-S h:"***heartbeat_interface_selection_string***"**] [**-h**] [**-TV**] *communication_group*

To change a reference in a network interface resource to a different communication group:

chcomg [-i n:*network_interface1*[:*node1*][,*network_interface2*[:*node2*]...] | -S n:"*network_interface_selection_string*"] [**-6**] [-h] [-TV] *communication_group*

Description

The chcomg command changes an existing communication group definition with the name specified by the *communication_group* parameter for the online peer domain. The communication group is used to define heartbeat rings for use by topology services and to define the tunables for each heartbeat ring. The communication group determines which devices are used for heartbeating in the peer domain.

The chcomg command must be run on a node that is currently online in the peer domain where the communication group is defined. One or more attributes can be changed with one chcomg command, but at least one change is required.

The -e and -m flags are used to set the network interface module (NIM) path and parameters. The NIM path is the path to the NIM that supports the adapter types used in the communication group. The NIM parameters are passed to NIM when it is started.

The chcomg command can also be used to assign a communication group to an interface resource. Use the -i flag to assign the communication group to a specific interface resource name. The interface resource can be limited to one on a particular node. An interface resource can also be specified using the -S flag and a selection string. This is used when specifying the interface resource name is not sufficient. Before a communication group can be removed, any interface resources that refer to it must be reassigned.

More than half of the nodes must be online to change a communication group in the domain.

Flags

-s *sensitivity*
Specifies the heartbeat sensitivity. This is the number of missed heartbeats that constitute a failure. The sensitivity is an integer greater than or equal to 4.

-p *period*
Specifies the period, which is the number of seconds between heartbeats. The value of *period* can be an integer or a floating-point number that is greater than or equal to 1.

-g *grace*
Specifies the grace period that is used when heartbeats are no longer received. When a heartbeat is missed, an Internet Control Message Protocol (ICMP) echo packet is sent to the failed node. If the echo is returned, the grace period is initiated.

The grace period is specified in seconds and is significant to milliseconds. It can be specified as an integer, a floating-point number, or one of these values:

0
Specifies that the grace period is disabled.

-1 | d
Specifies that the topology services subsystem controls the grace period. This is the default value.

-t *priority*
Specifies the priority. The priority indicates the importance of this communication group with respect to others. It is used to order the heartbeat rings. The lower the number, the higher the priority. The highest priority is `1`.

-b
Specifies that broadcast will be used if the underlying media support it. The `-b` flag cannot be used when specifying `-x b`.

-r
Specifies that source routing will be used if the underlying media support it. The `-r` flag cannot be used when specifying `-x r`.

-x b | r | br
Excludes control for the heartbeat mechanism. This indicates that one or more controls for heartbeat mechanisms should not be used even if the underlying media support it. The following can be excluded:

b
Specifies that broadcast should not be used even if the underlying media support it.

r
Specifies that source routing should not be used even if the underlying media support it.

Excluding more than one control is specified by listing the feature option letters consecutively (`-x br`).

-i h | n:*network_interface1*[:*node1*] [,*network_interface2*[:*node2*]...
Assigns this communication group to the network interface resource defined by the network interface resource name and optionally the node name where it can be found. Specify **-i h** for heartbeat interface resources or **-i n** for network interface resources. By default, the **-i n** flag adds network interface resources that have IPv4 addresses to *communication_group*. If the **-6** flag is specified, the **-i n** flag adds network interface resources that have IPv6 addresses to *communication_group*.

If `-i` is specified, `-S` cannot be specified.

-S h | n: "*network_interface_selection_string*"
Assigns this communication group to the interface specified by the network interface selection string. Specify **-S h** for heartbeat interfaces or **-S n** for network interfaces. By default, the **-S n** flag adds network interface resources that have IPv4 addresses to *communication_group*. If the **-6** flag is specified, the **-S n** flag adds network interface resources that have IPv6 addresses to *communication_group*.

If `-S` is specified, `-i` cannot be specified.

-e *NIM_path*
Specifies the network interface module (NIM) path name. This character string specifies the path name to the NIM that supports the adapter types in the communication group.

-m *NIM_parameters*
Specifies the NIM start parameters. This is a character string that is passed to the NIM when starting it.

-N *UseForNodeMembership*
Specifies whether group services use the communication group in calculating node membership. Sets the **UseForNodeMembership** persistent resource attribute for the communication group resource. Valid values are:

0
Indicates that, regardless of the results of liveness checks run on **NetworkInterface** resources that are members of this communication group, group services do not use those results in calculating whether the node owning the interfaces is online.

1
Indicates that group services use the results of liveness checks run on the **NetworkInterface** resources in calculating the online state of their owning nodes.

-6
Specifies that IPv6 addresses represented as resources on each interface have their communication group changed to the one specified. IPv4 addresses represented as resources on the interfaces are unaffected.

By default (without the **-6** flag specified), the inverse is true. Only IPv4 addresses represented as resources on the interface have their communication group changed.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

communication_group
Specifies the name of an existing communication group to be changed in the peer domain.

Security

The user of the `chcomg` command needs write permission for the `IBM.CommunicationGroup` resource class. Write permission for the `IBM.NetworkInterface` resource class is required to set the communication group for a network interface resource. By default, `root` on any node in the peer domain has read and write access to these resource classes through the configuration resource manager.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on

the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is defined and online to the peer domain where the communication group is to be changed.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the `-f "-"` or `-F "-"` flag is specified, this command reads one or more node names from standard input.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

In these examples, node `nodeA` is defined and online to peer domain `ApplDomain`.

1. To change the communication group `ComGrp1` for `ApplDomain` to a sensitivity of 4 and period of 3, run this command on `nodeA`:

```
chcomg -s 4 -p 3 ComGrp1
```

2. To change the communication group `ComGrp1` for `ApplDomain` to use broadcast, run this command on `nodeA`:

```
chcomg -b ComGrp1
```

3. To change the communication group `ComGrp1` for `ApplDomain` to no longer use source routing, run this command on `nodeA`:

```
chcomg -x r ComGrp1
```

4. To change the communication group `ComGrp1` for `ApplDomain`, to use a NIM path of `/opt/rsct/bin/hats_nim`, and to use NIM parameters `-l 5` to set the logging level, run this command on `nodeA`:

```
chcomg -e /opt/rsct/bin/hats_nim -m "-l 5" ComGrp1
```

5. To assign the communication group `ComGrp1` for `ApplDomain` to the heartbeat interface resource named `hbi0` on `nodeC`, run this command on `nodeA`:

```
chcomg -i h:hbi0:nodeC ComGrp1
```

6. To assign the communication group ComGrp1 for ApplDomain to the heartbeat interface resource named eth0 on nodeB, run this command on nodeA:

```
chcomg -i n:eth0:nodeC ComGrp1
```

7. To assign the communication group ComGrp1 for ApplDomain to the heartbeat interface resource that uses the subnet 9.345.67.812, run this command on nodeA:

```
chcomg -S h:"Subnet == '9.345.67.812'" ComGrp1
```

8. To assign the communication group ComGrp1 for ApplDomain to the network interface resource that uses the subnet 9.123.45.678, run this command on nodeA:

```
chcomg -S n:"Subnet == '9.123.45.678'" ComGrp1
```

9. To change the communication group **ComGrp1** for **ApplDomain** to a period of 500 milliseconds, run this command on **nodeA**:

```
chcomg -p 0.5 ComGrp1
```

Location

/opt/rsct/bin/chcomg

chcondition Command

Purpose

Changes any of the attributes of a defined condition.

Syntax

To change the attributes of a condition:

chcondition [-r *resource_class*] [-e "*event_expression*"] [-E "*rearm_expression*"] [-d "*event_description*"] [-D "*rearm_description*"] [**-b** *interval*[**,***max_events*][**,***retention_period*] [**,***max_totalsize*]] [-m l | m | p] [-n *node_name1*[,*node_name2*...]] [--qnotoggle | --qtoggle] [-s "*selection_string*"] [-S c | w | i] [**-g 0** | **1** | **2**] [-h] [-TV] *condition*[:*node_name*]

To rename a condition:

chcondition -c *new_condition* [-h] [-TV] *condition*[:*node_name*]

To lock or unlock a condition:

chcondition { -L | -U } [-h] [-TV] *condition*[:*node_name*]

Description

The chcondition command changes the attributes of a defined condition to the values supplied. If the name of the condition is changed using the -c flag, any condition/response associations remain intact.

If a particular condition is needed for system software to work properly, it may be locked. A locked condition cannot be modified or removed until it is unlocked. If the condition you specify on the chcondition command is locked, it will not be modified; instead an error will be generated informing you that the condition is locked. To unlock a condition, you can use the -U flag. However, since a condition is typically locked because it is essential for system software to work properly, you should exercise caution before unlocking it. To lock a condition so it cannot be modified, use the -L flag.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM

node groups and using the CSM `nodegrp` command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference.*

Flags

-b *interval*[,*max_events*][,*retention_period*][,*max_totalsize*]
Changes one or more batching-related attributes. Use commas to separate the attribute values. Do not insert any spaces between the values or the commas.

interval specifies that the events are to be batched together for the indicated interval. Batching continues until no events are generated for an interval. Use an interval of 0 to turn batching off.

max_events specifies that the events are to be batched together until the *max_events* number of events are generated. The interval restarts if the *max_events* number of events is reached before the interval expires.

retention_period specifies the retention period in hours. The batched event file is saved for the time specified as the retention period. Once this time is reached, the file is automatically deleted.

max_totalsize specifies the total size for the batched event file in megabytes (MB). The batched event file is saved until this size is reached. Once the size is reached, the file is automatically deleted.

max_events, *retention_period*, and *max_totalsize* cannot be specified unless interval is greater than 0. When *interval* is greater than 0 and *max_events* is 0, no maximum number of events is used.

If *retention_period* and *max_totalsize* are both specified, the batched event file is saved until the specified time or size is reached, whichever occurs first.

If you want to change one, two, or three attribute values, you must specify a valid value or an empty field for any attributes that precede the value you want to change. You do not have to specify any values for attributes that follow the value you want to change. For example, if you only need to change the retention period, you need to specify values for *interval* and *max_events* as well. You can provide an empty field if an attribute does not need to be changed. For example, to change the retention period to 36 hours without changing the values of *interval* and *max_events*, enter:

```
chcondition -b ,,36
```

-c *new_condition*
Assigns a new name to the condition. *new_condition*, which replaces the current name, is a character string that identifies the condition. If *new_condition* contains one or more spaces, it must be enclosed in quotation marks. A name cannot be null, consist of all spaces, or contain embedded double quotation marks.

-e "*event_expression*"
Specifies an *event expression,* which determines when an event occurs. An event expression consists of a dynamic attribute or a persistent attribute of *resource_class*, a mathematical comparison symbol (or <, for example), and a constant. When this expression evaluates to TRUE, an event is generated.

-E "*rearm_expression*"
Specifies a *rearm expression.* After *event_expression* has evaluated to TRUE and an event is generated, the rearm expression determines when monitoring for the *event_expression* will begin again. Typically,the rearm expression prevents multiple events from being generated for the same event evaluation. The rearm expression consists of a dynamic attribute of *resource_class*, a mathematical comparison symbol (>, for example), and a constant.

-d "*event_description*"
Describes the event expression.

-D "*rearm_description*"
Describes the rearm expression.

--g 0 | 1 | 2
Specifies granularity levels that control audit logging for the condition. The levels of granularity are:

0
Enables audit logging. ERRM writes all activities to the audit log. This is the default value.

1
Enables error logging only. ERRM writes only in case of errors to the audit log.

2
Disables audit logging. ERRM does not write any records to the audit log.

-L
Locks a condition so it cannot be modified or removed. When locking a condition using the **-L** flag, no other operation can be performed by this command.

-m l | **m** | **p**
Specifies the management scope to which the condition applies. The management scope determines how the condition is registered and how the selection string is evaluated. The scope can be different from the current configuration, but monitoring cannot be started until an appropriate scope is selected. The valid values are:

l
Specifies *local* scope. The condition applies only to the local node (the node where the condition is defined). Only the local node is used in evaluating the selection string.

-L
Locks a condition so it cannot be modified or removed. When locking a condition using the `-L` flag, no other operation can be performed by this command.

m
Specifies *management domain* scope. The condition applies to the management domain in which the node where the condition is defined belongs. All nodes in the management domain are used in evaluating the selection string. The node where the condition is defined must be the management server in order to use management domain scope.

p
Specifies *peer domain* scope. The condition applies to the peer domain in which the node where the condition is defined belongs. All nodes in the peer domain are used in evaluating the selection string.

-n *node_name1*[,*node_name2*...]
Specifies the host name for a node (or a list of host names separated by commas for multiple nodes) where this condition will be monitored. Node group names can also be specified, which are expanded into a list of node names.

You must specify the `-m` flag with a value of `m` or p if you want to use the `-n` flag. This way, you can monitor conditions on specific nodes instead of the entire domain.

The host name does not have to be online in the current configuration, but once the condition is monitored, the condition will be in error if the node does not exist. The condition will remain in error until the node is valid.

--qnotoggle
Specifies that monitoring does not toggle between the event expression and the rearm expression, but instead the event expression is always evaluated.

--qtoggle
Specifies that monitoring toggles between the event expression and the rearm expression.

-r *resource_class*
Specifies which resource class this condition will monitor. The `lsrsrcdef` command can be used to list the resource class names.

-s "*selection_string*"
Specifies a selection string that is applied to all of the *resource_class* attributes to determine which resources *event_expression* should monitor. The default is to monitor all resources within *resource_class*. The resources used to evaluate the selection string is determined by the management scope (the `-m` flag). The selection string must be enclosed within double or single quotation marks. For information on how to specify selection strings, see the *RSCT: Administration Guide* .

-S c | **w** | **i**
Specifies the severity of the event:

c
Critical

w
Warning

i
Informational (the default)

-U
Unlocks a condition so it can be modified or removed. If a condition is locked, this is typically because it is essential for system software to work properly. For this reason, you should exercise caution before unlocking it. When unlocking a condition using the `-U` flag, no other operation can be performed by this command.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

condition
Specifies the name of an existing condition that is defined on *node_name*.

node_name
Specifies the node in a domain where the condition is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

Security

The user of the `chcondition` command needs write permission to the `IBM.Condition` resource class on the node where the condition is defined. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To change the condition name from "FileSystem space used" to "Watch FileSystem space", run this command:

```
chcondition -c "Watch FileSystem space"  "FileSystem space used"
```

2. To change a rearm expression and rearm description for a condition with the name "tmp space used", run this command:

```
chcondition -E "PercentTotUsed < 80" \
-D "Start monitoring tmp again after it is less than 80 percent full" \
"tmp space used"
```

3. To disable the recording of audit log information for the condition called "File System space used", run this command:

```
chcondition -g 2 "File System space used"
```

4. To change the maximum size of the batched event file for the condition called "File System space used" to 100 MB, run this command:

```
chcondition -b ,,,100 "File System space used"
```

5. To disable batching for the condition called "File System space used", run this command:

```
chcondition -b 0 "File System space used"
```

   This command resets *max_event*, *retention_period*, and *max_totalsize*, if these values were previously specified. You must specify values for these attributes when you re-enable batching, if needed.

In the following examples, which apply to management domains, the node where the command is run is on the management server.

1. To change the condition with the name "FileSystem space used" on the management server to check for space usage that is greater than 95%, run this command:

```
chcondition -e "PercentTotUsed > 95" "FileSystem space used"
```

2. To change the condition with the name "NodeB FileSystem space used" on NodeB to check for space usage that is greater than 95%, run this command:

```
chcondition -e "PercentTotUsed > 95" \
"NodeB FileSystem space used":NodeB
```

This example applies to a peer domain:

1. To change the condition defined on NodeA with the name "FileSystem space used" to check for space usage that is greater than 95%, run this command:

```
chcondition -e "PercentTotUsed > 95" \
"FileSystem space used":NodeA
```

Location

/opt/rsct/bin/chcondition

chcons Command

Note: The console log can only be present under **/** ,**/usr** ,**/var** or **/tmp** directory alone.

Purpose

Redirects the system console to a specified device or file to be effective on the next startup of the system.

Syntax

chcons [**-a login** { **=disable** | **=enable**}] [**-a console_logname=***file*] [**-a console_logsize=***size*] [**-a console_logverb=***number*] [**-a console_tagverb=***number*] *PathName*

Description

The **chcons** command changes the system console effective on the next system startup. The current operation of the system console is not affected.

The *PathName* parameter must be a fully qualified path name to a device or file that is to become the system console.

If the *PathName* parameter specifies a file that does not exist, the **chcons** command creates the file at the next system startup. If the file does exist, the **chcons** command sends any console message output to the file. For a regular file, the system does not start the login program.

If the console path name is a character device, the system starts the login program on the device. Login is enabled on the console at all run levels. If no login is desired, use the **-a login=disable** flag.

> **CAUTION:** If the console is the only login terminal on the system, you cannot log in at the next start of the system using the **-a login=disable** flag.

Additional Information

The **chcons** command saves the specified information into the database to be used on the next start-up of the system with the console configuration method. This method checks the specified device path name to determine if it is a character special file. If it is not, or does not exist, the device path name is assumed to be a file, and the console is set accordingly. If the device path name is a character special file, the console configuration method uses the base name as a logical name and attempts to look up the device name in the device database. If the device is found and available, the console is set to the device.

If the device is not found or is found but not available, a console finder routine is run that displays a prompt requesting that a new system console device be selected. By default, the tty on the S1 port and all graphics displays will display the prompt. The **/etc/consdef** file must be modified to display the prompt on S2 or other ports.

For a device, an entry in the **inittab** file with the console identifier is set to the respawn action to allow a login on the console if the console login was specified as the **enable** parameter. This causes a login to be available at all run levels. If the console login was specified with the **disable** parameter or if a file is designated as the console, the console entry in the **inittab** file is set to the OFF action, and login is disabled on the console for all run levels.

Flags

Item	Description
-a login= [**disable** | **enable**]	Enables or disables the login on the console for all run levels at the next start-up of the system.
-a console_logname=*file*	Specifies the full path name to use for the console output log file.
-a console_logsize=*size*	Specifies the size, in bytes, of the console output log file.
-a console_logverb=*number*	Specifies the verbosity level for console output logging. Zero disables logging; 1 through 9 enable logging.
-a console_tagverb=*number*	Specifies the verbosity level for console output tagging. Zero disables tagging, 1 through 9 enable tagging.

Examples

1. To change the system console to a file called **console.out** in the **/tmp** directory, enter:

   ```
   chcons /tmp/console.out
   ```

2. To change the system console to a terminal with the `tty3` logical name, enter:

   ```
   chcons /dev/tty3
   ```

3. To change the system console to the terminal associated with the `/dev/tty3` device and ensure a login at the console, enter:

```
chcons -a login=enable /dev/tty3
```

4. To change the system console to a terminal with the `tty0` logical name and disable login at the console, enter:

```
chcons -a login=disable /dev/tty0
```

5. To change the console to the default physical LFT display, enter:

```
chcons /dev/lft0
```

Files

Item	Description
/dev/console	Specifies the special file for system console access.
/etc/consdef	Enables non-default terminal to be selected as the console device.
/usr/sbin/chcons	Specifies the command file.

chcore Command

Purpose

Changes the corefile settings.

Syntax

`chcore` [`-R` *registry*] [`-c` {on|`off`|`default`}] [`-p` {on|`off`|`default`}] [`-l` {*path*| default] [`-n` {on|`off`| `default`}] [*username* | `-d`]

Description

The `chcore` command is the user interface to change the core settings. It has the following usage:

```
chcore [-R registry] options [username|-d]
```

where,

options is at least one (and possibly more) of the following:

```
-c {on|off|default}
```

setting for core compression

```
-p {on|off|default}
```

setting for core location

```
-l path
```

specify directory to use

```
-n {on|off|default}
```

setting for core naming

If `-d` is specified, `chcore` will change the default setting for the system. The `-d` option is mutually exclusive with a specified *username* and with any specification of a *registry*. If neither `-d` nor a *username* is supplied, `chcore` will change the setting for the current user. Both the `-d` option and the ability to change settings for another user (other than the current user) are privileged operations, and may only be

run by root or another user with system authority. Any changes made will not take effect until the next login session.

To change attributes an alternate Identification and Authentication (I&A) mechanism, the **-R** flag can be used to specify the I&A load module. If the **-R** flag is not specified, the `chcore` command uses the default attributes. Load modules are defined in the **/usr/lib/security/methods.cfg** file.

Note: The core settings changed by the `chcore` command are persistent across reboots of the system.

Flags

Item	Description
`-c {on|off|default}`	Setting for core compression.
`-d`	Changes the default setting for the system.
`-l` *path*	Directory path for stored corefiles.
`-n {on|off|default}`	Setting for core naming.
`-p {on|off|default}`	Setting for core location.
`-R` *registry*	Specifies the loadable I&A module.

Security

The command can only be run by root or another user with system authority.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To make any process run by root dump compressed core files and restore the location of the core files to the system default, type:

   ```
   chcore -c on -p default root
   ```

   Note: If no default is specified, cores will dump in the current directory.
2. To enable a default core path for the system, type:

   ```
   chcore -p on -l /corefiles -d
   ```

   Note: All users who do not explicitly disable the core path with `chcore -p off` or override the core path with `chcore -l` will dump core files into the directory `/corefiles`. If a user does not have write permission to that directory, or the directory does not exist, no corefile will be generated.

Files

Item	Description
`/usr/lib/security/methods.cfg`	Contains load module definitions.
`/etc/security/user`	Contains extended user attributes.

chcosi Command

Purpose

Manages a Common Operating System Image (COSI).

Syntax

To install software:

chcosi **-i** **-s** *Source* [**-f***Fileset* | **-b** *installp_bundle* | **-F** *Fixes* | **-B** *fix_bundle*] [**-c**] [**-R**] [**-v**] *COSI*

To update software:

chcosi **-u** **-s** *Source* [**-f***Fileset* | **-b** *installp_bundle* | **-F** *Fixes* | **-B** *fix_bundle*] [**-c**] [**-R**] [**-v**] *COSI*

To reject software:

chcosi **-j** [**-f***Fileset* | **-b** *installp_bundle* | **-F** *Fixes* | **-B** *fix_bundle*] [**-R**] [**-v**] *COSI*

To remove software:

chcosi **-r** {**-f***Fileset* | **-b** *installp_bundle* | **-F** *Fixes* | **-B** *fix_bundle*} [**-R**] [**-v**] *COSI*

To remove software:

chcosi **-u** [**-f***Fileset* | **-b** *installp_bundle* | **-F** *Fixes* | **-B** *fix_bundle*] [**-R**] [**-v**] *COSI*

Description

The `chcosi` command manages a Common Operating System Image (COSI) created from the `mkcosi` command. Management tasks include installing, updating, rejecting, removing, and committing the software on the common image.

For installing and updating software on a common image, the required *Source* parameter specifies where the command gets installable images. The particular installable images are taken from the `-f`, `-b`, `-F`, `-B` flag and parameters. For the install, update, reject, and commit operations, if the `-f`, `-b`, `-F`, `-B` flags and parameters are not specified, the operation uses an assume-all value. So if the operation is an install or an update, all images from the source are used in the operation. If the operation is a reject or a commit, all software is committed or rejected from the common image. If the `-c` flag is specified with the install or update operation, the software is committed instead of applied. If a common image to be managed is being used by thin servers, a clone is created from the common image and the manage operation is performed on the clone image. The naming convention for the clone is the original common image name with the suffix `_X{`*`count`*`}`, where *count* is a number that is incremented every time a common image is cloned.

The `chcosi` command depends on the `bos.sysmgt.nim.master` fileset being present on the system. This command fails to execute if the `mkcosi` command is not run first to create a common image for managing.

Flags

Item	Description
`-b` *installp_bundle*	Specifies an `installp_bundle` NIM resource to be performed against the common image.
`-B` *fix_bundle*	Specifies a `fix_bundle` NIM resource to be performed against the common image.
`-c`	Specifies that the software to be installed or updated on the common image is put in the COMMIT state.

Item	Description
`-f` *Fileset*	Specifies a list of filesets to be performed against the common image.
`-F` *Fixes*	Specifies a list of fixes to be performed against the common image.
`-i`	Specifies the software to be installed.
`-j`	Specifies the software to be rejected.
`-r`	Specifies the software to be removed.
`-R`	Specifies the operation that is applied to requisite software.
`-s` *Source*	Specifies the source for common image management. The source can be an `lpp_source`, a device with installable media, a directory to installable images, or a remote location to installable images.
`-u`	Specifies the software to be updated or committed.
`-v`	Enables verbose debug output when the `chcosi` command runs.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

Access Control: You must have root authority to run the `chcosi` command.

Examples

1. To install `csm.core` software from a CD-ROM onto a common image named `cosi1`, enter:

   ```
   chcosi -i -s cd0 -f csm.core cosi1
   ```

   The `csm.core` fileset is installed on the `cosi1` common image, and the fileset is placed in an APPLIED state.

Location

`/usr/sbin/chcosi`

Files

Item	Description
`/etc/niminfo`	Contains variables used by NIM.

chdef Command

Purpose

Changes the default value of the predefined attribute.

Syntax

chdef [**-a** *Attribute = Value* **-c** *Class* **-s** *Subclass* **-t** *Type*]

chdef [**-H**]

chdef [**-h**]

Description

The **chdef** command modifies the default value of a predefined attribute of the specified device type. The modified default value must be within a specified list or range of values for the specified attribute, and only attributes, that have an explicit list or range of values can be modified. For devices that are of the same class, subclass, and type that are currently configured using the default value of the attribute, modifying the default value does not take effect for the device until you reboot or a subsequent unconfiguration and configuration operation takes place. This is similar to running the **chdev** command operation with the **-P** option except that the **chdef** command modifies every device of the same class, subclass, and type.

Note: It is recommended but not necessary to run the **bosboot** command after an execution of the **chdef** command.

Flags

Item	Description
-a *Attribute* **=** *Value*	Specifies the device attribute-value pair that can be used for setting the new default value. The *Attribute*=*Value* variable can be used to specify one attribute=value pair.
-c *Class*	Specifies the device class.
-h	Displays the command usage message.
-H	Displays headers above the column output.
-s *Subclass*	Specifies the subclass which is of the device.
-t *Type*	Specifies the device type from the predefined devices object class.

Security

Access Control

Privilege Control: Only the root user has the execute (x) access to this command.

Auditing Events

Event	Information
DEV_DEFAULT	The command line on the device.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To change the default value for the `hcheck_interval` attribute for an scsd disk from 0 to 3, enter:

```
chdef -a hcheck_interval=3 -c disk -s scsi -t scsd
```

2. To change the default value for the `hcheck_interval` attribute for an scsd disk back to the default of 0, enter:

```
chdef -a hcheck_interval=0 -c disk -s scsi -t scsd
```

3. To list all attributes that have modified default values with a header, enter:

```
chdef -H
```

Files

Item	Description
/usr/sbin/chdef	Contains the **chdef** command.

chdev Command

Purpose

Changes the characteristics of a device.

Syntax

chdev **-l** *Name* [**-a** *Attribute*=*Value ...*] [**-f** *File*] [**-h**] [**-p** *ParentName*] [**-P** | **-T**] [**-U**][**-q**] [**-w** *ConnectionLocation*] [**-g**]

Description

The **chdev** command changes the characteristics of the specified device with the given device logical name that is specified with the **-l** *Name* flag. The device can be in the Defined, Stopped, or Available state. Some changes may not be allowed when the device is in the Available state. When changing the device characteristics, you can supply the flags either on the command line or in the specified **-f** *File* flag.

When the **-P**, **-U**, and **-T** flags are not specified, the **chdev** command applies the changes to the device and updates the database to reflect the changes. If the **-P** flag is specified, only the database is updated to reflect the changes, and the device is left unchanged. This is useful in cases where a device cannot be changed because it is in use. In cases where the device is in use, the changes can be made to the database with the **-P** flag, and the changes will be applied to the device when the system is restarted.

If the **–U** flag is specified, the database is updated to reflect the changes, and the device is changed while the device remains in the Available state. This option is applicable only to attributes that can be updated while the device is in the Available state. When the **–U** flag is specified the database is updated with the attributes that are provided with the **–U** flag and the device is changed to the current values of all attributes that can be updated while the device is in the Available state. See the **lsattr** command to determine whether the device supports this attribute type.

The **-T** flag is used to make a temporary change in the device without the change being reflected in the database.The device temporary reverts to the characteristics that are described in the database when the system is restarted. All devices do not support the **-P**, **-U**, and **-T** flags. If a device is in the Defined state, changes are applied only to the database.

Attention: To protect the Configuration database, the **chdev** command is not interruptible. Stopping this command before it is complete could result in a corrupted database.

You can use the System Management Interface Tool (SMIT) **smit chdev** fast path to change device characteristics.

Flags

Item	Description
-a *Attribute*=*Value*	Specifies the device attribute-value pairs used for changing specific attribute values. The *Attribute*=*Value* parameter can use one attribute value pair or multiple attribute value pairs for one **-a** flag. If you use an **-a** flag with multiple attribute value pairs, the list of pairs must be enclosed in quotes with spaces between the pairs. For example, entering `-a Attribute=Value` lists one attribute value pair per flag, while entering `-a 'Attribute1=Value1 Attribute2=Value2'` lists more than one attribute value pair.
-f *File*	Reads the necessary flags from the named *File* parameter.
-g	Forces the change operation to take place on a locked device.
-h	Displays the command usage message.
-l *Name*	Specifies the device logical name in the Customized Devices object class whose characteristics are to be changed.
-P	Changes the device's characteristics permanently in the Customized Devices object class without actually changing the device. This is useful for devices that cannot be made unavailable and cannot be changed while in the available state. The change is made to the database, and the changes are applied to the device when the system is rebooted. This flag cannot be used with the **-T** flag. Not all devices support the **-P** flag.
-p *ParentName*	Specifies the new device logical name of the parent device in the Customized Devices object class. Use this flag only when changing the parent of the device. Not all devices support the **-p** flag.
-q	Suppresses the command output messages from standard output and standard error.
-T	Changes the characteristics of the device temporarily without changing the Customized Devices object class for the current start of the system. This flag cannot be used with the **-P** flag. Not all devices support the **-T** flag.
-U	Changes the characteristics of the device while allowing the device to remain in the Available state. This flag cannot be used with the **-P** or **-T** flag. Not all devices and attributes support the **-U** flag.
-w *ConnectionLocation*	Specifies the new connection location of the device on the parent. Use this flag only when changing the connection location of the device. Not all devices support the **-w** flag.

Security

Access Control

Only the root user and members of the security group should have execute (x) access to this command.

Auditing Events

Auditing Event	Information
DEV_Change	Parameters to the method the **cfgmgr** command calls.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To change the retension instructions of the `rmt0` 4mm SCSI tape drive so that the drive does not move the tape to the beginning, then to the end, and then back to the beginning each time a tape is inserted or the drive is powered on, enter the following:

   ```
   chdev -l rmt0 -a ret=no
   ```

   The system displays a message similar to the following:

   ```
   rmt0 changed
   ```

2. To change one or more attributes of the `tok0` token-ring adapter to preset values as described in the `changattr` file, enter the following:

   ```
   chdev -l tok0 -f changattr
   ```

   The system displays a message similar to the following:

   ```
   tok0 changed
   ```

3. To change the SCSI ID of the available `scsi0` SCSI adapter that cannot be changed made unavailable due to available disk drives connected to it, enter the following:

   ```
   chdev  -l scsi0 -a id=6 -P
   ```

   The system displays a message similar to the following:

   ```
   scsi0 changed
   ```

   To apply the change to the adapter, shut down and restart the system.

4. To move the defined `tty11` tty device to port 0 on the `sa5` serial adapter, enter the following:

   ```
   chdev  -l tty11  -p sa5  -w 0
   ```

   The system displays a message similar to the following:

   ```
   tty11 changed
   ```

5. To change the maximum number of processes allowed per user to 100, enter the following:

   ```
   chdev -l sys0 -a maxuproc=100
   ```

   The system displays a message similar to the following:

   ```
   sys0 changed
   ```

6. To delete the `alias4=10.3.4.3` Object Data Manager (ODM) entry from the `en2` standard Ethernet network interface, enter the following:

   ```
   chdev -l en2 -a delalias4=10.3.4.3
   ```

   The system displays a message similar to the following:

   ```
   en2 changed
   ```

7. To delete the `alias6=fe80::20b4:40ff:fe00:f016/64` ODM entry from the `en3` standard Ethernet network interface, enter the following:

```
chdev -l en3 -a delalias6=fe80::20b4:40ff:fe00:f016/64
```

   The system displays a message similar to the following:

```
en3 changed
```

8. To enable dynamic tracking for a FC adapter:

```
chdev -l fscsix -a dyntrk=yes
```

9. To enable `fast_fail` for a FC adapter:

```
chdev -l fscsix -a fc_err_recov=fast_fail
```

Files

Item	Description
/usr/sbin/chdev	Specifies the command file.

chdisp Command

Purpose

The **chdisp** command changes the default display being used by the Low Function Terminal Subsystem.

Syntax

chdisp { **-d** *DeviceName* | **-p** *DeviceName* }

Description

The **chdisp** command changes the display used by the low function terminal (LFT) subsystem.

To generate a list of available displays and their respective display identifiers and descriptions, use the **lsdisp** command. For an example of the listing displayed, see the **lsdisp** command example listing.

> **Note:** The **chdisp** command can be used only on an LFT.

You could also use the System Management Interface Tool (SMIT) **smit chdisp** fast path to run this command for certain devices.

Flags

Item	Description
-d *DeviceName*	Changes the default display currently being used by the LFT. This change is temporary resulting in the default display reverting back to the original display when the system is rebooted.
-p *DeviceName*	Changes the default display to the specified display at the next reboot. This stays in effect until the user changes the default display again. The user must have superuser access to use this option.

Examples

1. To temporarily change the default display to a display with a device name `ppr0`, enter:

```
chdisp  -d ppr0
```

2. To permanently change the default display beginning with the next reboot to a display with the device name gda1, enter:

```
chdisp  -p gda1
```

Files

Item	Description
/bin/chdisp	Contains the **chdisp** command.

chdom Command

Purpose

Changes the domain attributes.

Syntax

chdom *Attribute* **=** *Value* ... *Name*

Description

The **chdom** command modifies attributes of the domain that the *Name* parameter identifies. This command only modifies attributes of existing domains in the domain database. To change an attribute of a domain, specify the attribute name and the new value with the *Attribute*=*Value* parameter. If the specified attribute or attribute value is invalid, the **chdom** command does not modify the domain.

Although modification of the *ID* attribute of a domain is allowed, it can affect the security aspects of the system because processes and files might be using the current value of the ID. In general, only modify the ID of a domain if that the domain has not been used. When the system is operating in enhanced role-based access control (RBAC) mode, modifications made to the domain database are not used for security considerations until the database has been sent to the kernel security tables (KST) through the **setkst** command.

Attributes

Item	Description
ID	Specifies a unique integer that is used to identify the domain.

Parameters

Item	Description
Name	Specifies the domain to be modified.

Security

The **chdom** command is a privileged command. Invokers of the command must have activated a role that has the following authorization to run the command successfully.

Item	Description
aix.security.dom.change	Required to execute the command.

Files Accessed

Mode	File
rw	/etc/security/domains

Examples

1. To change the ID of the domain `hrdom`, enter:

```
chdom id=99 hrdom
```

checkeq, checkmm Command

Purpose

Checks documents formatted with memorandum macros.

Syntax

{ **checkeq** | **checkmm** } [*File*...]

Description

The **checkeq** command is used to check for syntax errors in the specified files (*File*) that have been prepared for the **neqn** or **eqn** command. The **checkeq** command reports missing or unbalanced delimiters and the **.EQ** and **.EN** macro pair.

The **checkeq** command is functionally equivalent to the **checkmm** command.

The **checkmm** (check memorandum macros) command is used to check for syntax errors in files that have been prepared for the **mm** command or **mmt** command. For example, the **checkmm** command checks that you have a **.DE** (display end) macro corresponding to every **.DS** (display start) macro. *File* specifies files to be checked by the **checkeq** or **checkmm** command.

The output for the **checkmm** command is the number of lines checked and a list of macros that are unfinished because of missing macros.

checknr Command

Purpose

Checks **nroff** and **troff** files.

Syntax

checknr [**-a**.*Macro1*.*Macro2* ...] [**-c**.*Command1*.*Command2* ...] [**-f**] [**-s**] [*File* ...]

Description

The **checknr** command checks a list of **nroff** or **troff** input files for certain kinds of errors involving mismatched opening and closing delimiters and unknown commands. If no files are specified, the **checknr** command checks standard input.

Delimiters checked are:

- Font changes using the **\f***Newfont* ... **\f**P.
- Size changes using the **\s***Newsize* ... **\s**0.

- Macros that come in open and close forms (such as the **.TS** and **.TE** macros) that must always come in pairs.

The **checknr** command can handle both the **ms** and **me** macro packages.

The **checknr** command is intended to be used on documents that are prepared with the **checknr** command in mind, much the same as the **lint** command. The **checknr** command requires a certain document writing style for the **\f** and **\s** commands, in that each **\f***Newfont* must be terminated with **\fP** and each **\s***Newsize* must be terminated with **\s0**. While it works to go directly into the next font or to explicitly specify the original font or point size, such a practice produces error messages from the **checknr** command.

File specifies **nroff** or **troff** input files for errors involving mismatched opening and closing delimiters and unknown commands. The default is standard input.

Flags

Item	Description
-a.*Macro1***.***Macro2*	Adds pairs of macros to the list. This flag must be followed by groups of six characters, each group defining a pair of macros. The six characters are a period, *Macro1*, another period, and *Macro2*. For example, to define the pair, **.BS** and **.ES**, use **-a.BS.ES**. **Note:** There is no way to define a 1-character macro name using the **-a** flag.
-c.*Command1***.***Command2*	Defines otherwise undefined commands that would get error messages from the **checknr** command.
-f	Causes the **checknr** command to ignore **\f** font changes.
-s	Causes the **checknr** command to ignore **\s** size changes.

Note: The **checknr** command does not correctly recognize certain reasonable constructs, such as conditionals.

cw, checkcw Command

Purpose

Prepares constant-width text for the **troff** command.

Syntax

cw [**+t** | **t**] [**-d**] [**-f** *font*] [**-l** *Delimiter*][**-r** *Delimiter*] [File...]

checkcw [**-l** *Delimiter*] [**-r** *Delimiter*] [File...]

Description

The **cw** command preprocesses any specified **troff** files containing English-language text to be typeset in the constant-width (CW*)* font. The **cw** command reads standard input if you do not specify a file or if you specify a **-** (minus sign) as one of the input file names. The **cw** command writes to standard output.

Because output resulting from this command resembles the output of line printers and workstations, use this command to typeset examples of programs and computer output for user manuals and programming text. The **cw** command produces distinctive output when used with the Times Roman font.

The CW font contains a nonstandard set of characters. Any text typeset with this font requires different character and interword spacing from that used for standard fonts. Therefore, you must use the **cw** command to preprocess documents that use the CW font.

The CW font contains the following 94 ASCII printing characters:

```
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
0123456789
!$%&()`'*+@.,/:;=?[]|-_^~"<>{}#\
```

This font also contains 11 non-ASCII characters represented by 4-character **troff** strings (in some cases attaching these strings to nonstandard graphics).

The **cw** command recognizes five request lines as well as user-defined delimiters. The request lines look like **troff** macro requests. The **cw** command copies them in their entirety onto the output. Thus, you can define the requests as **troff** macros; in fact, the **.CW** and **.CN** macros should be so defined. The five requests are:

Flags

Item	Description
.CW	Marks the start of text to be set in the CW font. This request causes a break. It can take the same flags (in the same format) as those available on the **cw** command line.
.CN	Marks the end of text to be set in the CW font. This request causes a break. It can take the same flags (in the same format) as those available on the **cw** command line.
.CD	Changes the delimiters and settings of other flags. It can take the same flags (in the same format) as those available on the **cw** command line. The purpose of this request is to allow the changing of flags other than at the beginning of a document.
.CP *Option-list*	Concatenates all the options (delimited like **troff** macro options), with the odd-numbered options set in the CW font and the even-numbered options set in the prevailing font.
.PC *Option-list*	Acts the same as the **.CP** macro, except the even-numbered options are set in CW font and the odd-numbered options are set in the prevailing font.

The **.CW** and **.CN** requests should bracket text that is to be typeset as is, using the CW font. Normally, the **cw** command operates in the transparent mode. In that mode, every character between **.CW** and **.CN** request lines represents itself, except for the **.CD** request and the special 4-character names listed previously. In particular, the **cw** command causes all **.** (periods) and **'** (apostrophes) at the beginning of lines, and all **** (backslashes) and ligatures (such as fi and ff), to be hidden from the **troff** command. The transparent mode can be turned off by using the **-t** flag, in which case normal **troff** rules apply. In either case, the **cw** command hides from the user the effect of the font changes generated by the **.CW** and **.CN** requests.

You can also use the **-l** and **-r** flags to define delimiters with the same function as the **.CW** and **.CN** requests. These requests are meant to enclose words or phrases that are set in CW font in the running text. The **cw** command treats text between delimiters as it does text bracketed by **.CW**/**.CN** pairs, with one exception. Spaces within **.CW**/**.CN** pairs, have the same width as other CW characters, while spaces within delimited text are half as wide, so they have the same width as spaces in the prevailing text. Delimiters have no special meaning inside **.CW**/**.CN** pairs.

The **checkcw** command checks that left and right delimiters as well as the **.CW**/**.CN** pairs are properly balanced. It prints out all lines in the selection with the unmatched delimiters.

Notes:

1. The **.** (period) or **** (backslash)delimiter characters should not be used.

2. Certain CW characters do not combine well with certain Times Roman characters; for example, the spacing between a CW **&** (ampersand) followed by a Times Roman **,** (comma). In such cases, using **troff** half- and quarter-space requests can help.
3. The **troff** code produced by the **cw** command is difficult to read.
4. The **mm** macro package and **mv** macro package contain definitions of **.CW** and **.CN** macros that are adequate for most users. If you define your own macros, make sure that the **.CW** macro starts the **troff** no-fill (**.nf**) mode, and the **.CN** macro restores the fill mode (**.fi**), if appropriate.
5. When set in running text, the CW font is meant to be set in the same point size as the rest of the text. In displayed matter, on the other hand, it can often be profitably set 1 point smaller than the prevailing point size. The CW font is sized so that, when it is set in 9-point, there are 12 characters per column inch.
6. Documents that contain CW text can also contain tables and equations. In this case, the order of preprocessing must be the **cw** command, **tbl** command, and **eqn** command. Usually, the tables do not contain CW text, although it is possible to have elements in the table set in the CW font. Ensure that the **cw** command does not modify the **tbl** command format information. Attempts to set equations in the CW font are usually unsuccessful.
7. In the CW font, overstriking is most easily accomplished with backspaces. Because spaces (and therefore backspaces) are half as wide between delimiters as inside **.CW**/**.CN** pairs, two backspaces are required for each overstrike between delimiters.
8. Some devices such as the IBM 3816 Pageprinter do not have a CW font. You receive a `troff can't open /usr/lib/font/devNAME/CW.out` message for these devices. The **troff** command uses the font in font position 3 as the CW font.

Item	Description
+t	Turns the transparent mode on (this is the default).
t	Turns the transparent mode off.
d	Displays the current flag settings on the standard error output in the form of **troff** comment lines. This flag is meant for debugging.
f *Font*	Replaces the value of the *Font* variable with the **cw** command font (the default equals 3, which replaces the bold font). The **-f5** flag is commonly used for matters that allow more than four simultaneous fonts.

Note: This flag is useful only on the command line.

Item	Description
-l *Delimiter*	Sets the left delimiter as the 1- or 2-character string specified by the *Delimiter* variable. The left delimiter is undefined by default.
-r *Delimiter*	Sets the right delimiter to that specified by the *Delimiter* variable. The right delimiter is undefined by default. The left and right delimiters can (but need not) be different.

Parameters

Item	Description
File	Specifies **troff** English-language text files to be preprocessed by the **cw** command to produce constant-width characters in the output file.
File	Specifies **troff** English-language text files to be preprocessed by the **checkcw** command to check right and left delimiters as well as **.CW** and **.CN** pair balance.

chedition Command

Purpose

Allows query or change of the current signature file on the system.

Syntax

To list the current edition on the system:

```
chedition -l
```

To change to the `standard` edition:

```
chedition -s  [-d Device  [ -p ]]
```

To change to the `enterprise` edition:

```
chedition -e  [-d Device  [ -p ]]
```

To change to the `enterprise_cloud` edition:

```
chedition -c  [-d Device  [ -p ]]
```

Description

The **chedition** command can be used to query the current edition of the system. The edition of the system, either express, standard, or enterprise will be displayed. The edition may also be changed by specifying the new edition the customer wishes to change to. If a bundle file exists for the new edition in `/usr/sys/inst.data/sys_bundles`, it will be installed if the device or directory containing the images to install is specified. Changing the edition modifies the signature file that is located in the `/usr/lib/bos/swidtag` directory. Depending on the level of the AIX operating system installed, previous locations were `/usr/lpp/bos/iso-swid`, `/usr/lpp/bos/properties/version`, and `/usr/lpp/bos`.

If you have upgraded from a recent Service Pack or Technology Level, then the **chedition** command might not work . If there are changes in the edition signature file names, you can change the edition on the system. You can choose one of the following editions: `standard`, `enterprise`, or `enterprise_cloud`. New signature files ship with the `bos.rte` update. The newest signature files are available after all the software are at the new Service Pack level or Technology level.

If you have upgraded only `bos.rte.install`, from a more recent Service Pack or Technology Level, then the **chedition** command might not work if you attempt to change the edition on the system. This can happen if there are changes in the edition signature file names. New signature files ship with the `bos.rte` update. Once all the software is at the new Service Pack or Technology level, the newest signature files will be available.

Flags

Item	Description
-d *Device or Directory*	Specifies the device or directory containing the images to install.
-e	Used when changing to the `enterprise` edition.
-l	List the current edition of the system. The edition of the system, either express, standard, or enterprise will be displayed.
-p	Performs a preview of the bundle file installation by running all per-installation checks. The edition of the system will not be updated.

Item	Description
-s	Used when changing to the `standard` edition.
-c	Used when changing to the `enterprise_cloud` edition.

Examples

1. To list the current edition on the system, type:

   ```
   chedition -l
   ```

   One of the following outputs will be returned:

   ```
   standard | enterprise | enterprise_cloud
   ```

2. To change to the standard edition, type:

   ```
   chedition -s
   ```

3. To change to the `enterprise` edition and perform a preview install of the contents of the enterprise edition bundle file, should it exist, type:

   ```
   chedition -e -d /usr/sys/inst.images -p
   ```

Files

Item	Description
/usr/sbin/chedition	Contains the **chedition** command.
/usr/sys/inst.data/sys_bundles	Contains system bundle files.

chfilt Command

Purpose

Changes a filter rule.

Syntax

chfilt **-v** **4|6** **-n** *fid* [**-a** **D**|**P**|I|L|E|H|S] [**-s** *s_addr*] [**-m** *s_mask*] [**-d** *d_addr*] [**-M** *d_mask*] [**-g** **Y**|**N**] [**-c** *protocol*] [**-o** *s_opr*] [**-p** *s_port*] [**-O** *d_opr*] [**-P** *d_port*] [**-r** **R**|**L**|**B**] [**-w** **I**|**O**|**B**] [**-l** **Y**|**N**] [**-f** **Y|N|O|H**] [**-t** *tid*] [**-i** *interface*][**-D** *description*] [-e *expiration_time*] [-x *quoted_pattern* | -X *pattern_filename* | -C *antivirus_filename*]

Description

Use the **chfilt** command to change the definition of a filter rule in the filter rule table. Auto-generated filter rules and manual filter rules can be changed by this command. If an auto-generated filter rule is modified by the **chfilt** command it will then become a manual filter rule. IPsec filter rules for this command can be configured using the `genfilt` command or IPsec smit (IP version 4 or IP version 6).

Flags

Item	Description
-a *Action*	The following *Action* values are allowed: • D (Deny) blocks traffic. • P (Permit) allows traffic. • I makes this an IF filter rule. • L makes this an ELSE filter rule. • E makes this an ENDIF filter rule. • H makes this a SHUN_HOST filter rule. • S makes this a SHUN_PORT filter rule.
-C *anitvirus_filename*	Specifies the antivirus file name. The -C flag understands some versions of ClamAV Virus Database (http://www.clamav.net).
-c *protocol*	Protocol. The valid values are: **udp**, **icmp**, **icmpv6**, **tcp**, **tcp/ack**, **ospf**, **ipip**, **esp**, **ah**, and **all**. Value **all** indicates that the filter rule will apply to all the protocols. The protocol can also be specified numerically (between 1 and 252).
-d *d_addr*	Destination address. It can be an IP address or a host name. If a host name is specified, the first IP address returned by the name server for that host will be used. This value along with the destination subnet mask will be compared against the destination address of the IP packets.
-D	Filter description. A short description text for the filter rule.
-e *expiration_time*	Specifies the amount of time the rule should remain active in minutes. The *expiration_time* does not remove the filter rule from the database. The *expiration_time* relates to the amount of time the filter rule is active while processing network traffic. If no *expiration_time* is specified, the live time of the filter rule is infinite. If the *expiration_time* is specified in conjunction with a SHUN_PORT (-a S) or SHUN_HOST (-a H) filter rule, then this is the amount of time the remote port or remote host is denied or shunned once the filter rule parameters are met. If this *expiration_time* is specified independent of a shun rule, this is the amount of time the filter rule will remain active after the filter rules are loaded into the kernel and start processing network traffic.
-f	Fragmentation control. This flag specifies that this rule will apply to either all packets (**Y**), fragment headers and unfragmented packets only (**H**), fragments and fragment headers only (**O**), or unfragmented packets only (**N**).
-g	Apply to source routing? Must be specified as **Y** (yes) or **N** (No). If **Y** is specified, this filter rule can apply to IP packets that use source routing.
-i *interface*	The name of IP interface(s) to which the filter rule applies. Examples are: **all**, **tr0**, **en0**, **lo0**, and **pp0**.
-l	Log control. Must be specified as **Y** (yes) or **N** (No). If specified as **Y**, packets that match this filter rule will be included in the filter log.
-M *d_mask*	Destination subnet mask. This will be applied to the Destination address(**-d** flag) when compared with the destination address of the IP packets.
-m *s_mask*	Source subnet mask. This will be applied to the Source address (**-s** flag) when compared with the source address of the IP packet.
-n *fid*	The ID of the filter rule you want to change. It must exist in the filter rule table and for IP version 4, it cannot be 1 (rule 1 is a system reserved rule and is unchangeable).

Item	Description
-O *d_opr*	Destination port or ICMP code operation. This is the operation that will be used in the comparison between the destination port/ICMP code of the packet with the destination port or ICMP code (**-P** flag). The valid values are: **lt**, **le**, **g**t, **ge**, **eq**, **neq**, and **any**. This value must be **any** when the **-c** flag is **ospf**.
-o *s_opr*	Source port or ICMP type operation. This is the operation that will be used in the comparison of the source port/ICMP type of the packet with the source port or ICMP type (**-p** flag) specified in this filter rule. The valid values are: **lt**, **le**, **gt**, **ge**, **eq**, **neq**, and **any**. The value must be **any** when the **-c** flag is **ospf**.
-P *d_port*	Destination port/ICMP code. This is the value/code that will be compared to the destination port (or ICMP code) of the IP packet.
-p *s_port*	Source port or ICMP type. This is the value/type that will be compared to the source port (or ICMP type) of the IP packet.
-r	Specifies whether the rule will apply to forwarded packets (**R**), packets destined or originated from the local host (**L**), or both (**B**).
-s *s_addr*	Specifies the source address. It can be an IP address or a host name. If a host name is specified, the first IP address returned by the name server for that host will be used. This value along with the source subnet mask will be compared against the source address of the IP packets.
-t *tid*	Specifies the ID of the tunnel related to this filter rule. All the packets that match this filter rule must go through the specified tunnel.
-v	Specifies the IP version of the target filter rule.
-w	Specifies whether the rule will apply to incoming packets (**I**), outgoing packets (**O**), or both (**B**).
-X *pattern_filename*	Specifies the pattern file name. If more than one patterns are associated with this filter rule, then a pattern file name must be used. The pattern file name must be in the format of one pattern per line. A pattern is an unquoted character string. This file is read once when the filter rules are activated. For more information, see the `mkfilt` command.
-x *quoted_pattern*	Specifies the quoted character string or pattern. The -x *pattern* flag is compared against network traffic.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

chfn Command

Purpose

Changes a user's gecos information.

Syntax

chfn [**-R** *load_module*] [*Name*]

Description

The **chfn** command changes a user's gecos information. Gecos information is general information stored in the **/etc/passwd** file. This information is not used by the system. The type of information you store in this field is up to you. Some system administrators store information such as the user's full name, phone number, and office number.

The **chfn** command is interactive. After you enter the command, the system displays the current gecos information and prompts you to change it. To exit the **chfn** command without changing any information, press Enter.

You can use any printable characters in the gecos information string except a **:** (colon), which is an attribute delimiter.

By default, the **chfn** command changes the gecos information of the user who runs the command. You can also use this command to change the gecos information of other users. However, you must have execute permission for the **chuser** command to change the gecos information for another user.

For users that were created using an alternate Identification and Authentication mechanism (I&A) , the **-R** flag can be used to specify the I&A load module used to create the user. Load modules are defined in the **/usr/lib/security/methods.cfg** file.

Flag

Item	Description
-R	Specifies the loadable I&A module used to change the user's gecos information

Security

Access Control

All users should have execute (x) access to this command since the program enforces its own access policy. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the security group with the **setgid** (SGID) bit set.

Files Accessed

Mode	File
x	**/usr/bin/chuser**
rw	**/etc/passwd**

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Limitations

Changing a user's gecos information may not be supported by all loadable I&A modules. If the loadable I&A module does not change a user's gecos information, an error is reported.

Examples

1. If you are John Smith and want to change your gecos information, type:

   ```
   chfn
   ```

   The current gecos string appears, followed by a prompt that asks if a change should be made:

```
current gecos:
     "John Smith;555-1746;room 74"
change (y/n)? >
```

To change the room number from 74 to 36, type y to request a change and type the revised information when the `to? >` prompt appears:

```
current gecos:
     "John Smith;555-1746;room 74"
change (y/n)? > y
to? > John Smith;555-1746;room 36
```

2. If you are John Smith and want to view your gecos information but not change it, type:

```
chfn
```

The current gecos string appears, followed by a prompt that asks if a change should be made:

```
current gecos:
     "John Smith;555-1746;room 74"
change (y/n)? >
```

If you decide not to change the information, type n after the `change (y/n)?` prompt or press the Enter key:

```
current gecos:
     "John Smith;555-1746;room 74"
change (y/n)? > n
```

This is your opportunity to indicate that the information should remain unchanged. If you enter y, you are committed to enter an information string or use the Enter key to set the string to null. Note that the function of the Enter key differs before and after a y character is entered.

3. If you have execute (x) permission for the **chuser** command and want to change the gecos information for the `johns` user, type:

```
chfn johns
```

The current gecos string and prompts appear as in Example 1.

4. To change the gecos for an LDAP I&A load module defined user `davis`, type:

```
chfn -R LDAP davis
```

Files

Item	Description
/usr/bin/chfn	Specifies the path to the **chfn** command.
/usr/bin/chuser	Changes user information.
/etc/passwd	Contains basic user attributes.

chfont Command

Purpose

Changes the default font selected at boot time.

Syntax

chfont [*FontID*]

Description

The chfont command changes the font used by a display at system restart.

To see a list of available fonts with their respective font ids, font names, the glyph size and the font encoding, see the **lsfont** command. For an example of the listing displayed, see the **lsfont** command example listing.

You must have root authority to run this command.

Note: This command can be used only on an LFT (Low Function Terminal).

You could also use the System Management Interface Tool (SMIT) **smit chfont** fast path to run this command.

Parameter

Item	Description
FontID	The font id of the new font.

Examples

To change the font used by this display to the third font in the font palette, enter:

```
chfont 2
```

Files

Item	Description
/bin/chfont	Contains the **chfont** command.
/usr/lpp/fonts	Contains the font directory.

chfs Command

Purpose

Changes attributes of a file system.

Syntax

chfs [**-n** *NodeName*] [**-m** *NewMountPoint*] [**-u** *MountGroup*] [**-A** { **yes** | **no** }] [**-p** { **ro** | **rw** }] [**-t** { **yes** | **no** }] [**-a** *Attribute*=*Value*] [**-d** *Attribute*] *FileSystem*

Description

The **chfs** command changes the attributes of a file system. The new mount point, automatic mounts, permissions, and file system size can be set or changed. The *FileSystem* parameter specifies the name of the file system, expressed as a mount point.

Some file system attributes are set at the time the file system is created and cannot be changed. For the Journaled File System (JFS), such attributes include the fragment size, block size, number of bytes per i-node, compression, and the minimum file system size. For the Enhanced Journaled File System (JFS2), the block size cannot be changed.

The **chfs** command also accepts attributes that have no meaning to the file system. The attributes are saved in the `/etc/filesystems` file, but the file system does not act on the attributes. Additional attributes must be limited. The total size of a stanza in the `/etc/filesystems` file cannot exceed 512 bytes. If the size exceeds the limit, the stanza is no longer recognized.

The **chfs** command ignores any *Attribute=Value* pair that the command does not understand but adds them to an appropriate stanza in the **/etc/filesystems** file.

Example:

```
chfs -a abcd=1G /
```

This will set the new **abcd** attribute to the value of **1G** in the root stanza in **/etc/filesystems** file.

Flags

Item	Description
-a *Attribute=Value*	Specifies the *Attribute=Value* pairs dependent on virtual file system type. To specify more than one Attribute=Value pair, provide multiple -a *Attribute=Value* parameters. The following attribute or value pairs are specific to the Journaled File System (JFS): **-a copy=Copy#** Specifies which mirror copy to split off when used in conjunction with the *splitcopy* attribute. The default copy is the second copy. Valid values are 1, 2, or 3. **-a log=LVName** Specifies the full path name of the filesystem logging logical volume name of the existing log to be used. The log device for this filesystem must reside on the same volume group as the filesystem. **-a size=NewSize** Specifies the size of the Journaled File System. The size can be specified in units of 512-byte blocks, megabytes or gigabytes. If Value has the M suffix, it is interpreted to be in megabytes. If Value has a G suffix, it is interpreted to be in gigabytes. If Value begins with a +, it is interpreted as a request to increase the file system size by the specified amount. If the specified size is not evenly divisible by the physical partition size, it is rounded up to the closest number that is evenly divisible. The volume group in which the file system resides defines a maximum logical volume size and also limits the file system size. The maximum size of a JFS file system is a function of its fragment size and the nbpi value. These values yield the following size restrictions:

NBPI	Minimum AG Size	Fragment Size	Maximum Size (GB)
512	8	512, 1024, 2048, 4096	8
1024	8	512, 1024, 2048, 4096	16
2048	8	512, 1024, 2048, 4096	32
4096	8	512, 1024, 2048, 4096	64
8192	8	512, 1024, 2048, 4096	128
16384	8	1024, 2048, 4096	256
32768	16	2048, 4096	512
65536	32	4096	1024
131072	64	4096	1024

-a splitcopy=NewMountPointName
Splits off a mirrored copy of the file system and mounts it read-only at the new mount point. This provides a copy of the file system with consistent JFS meta-data that can be used for backup purposes. User data integrity is not guaranteed, so it is recommended that file system activity be minimal while this action is taking place. Only one copy may be designated as an online split mirror copy.

Item	Description
-a *Attribute=Value*	The following attribute or value pairs are specific to the Enhanced Journaled File System (JFS2): **-a ea=v2** Converts the JFS2 file system extended attribute (ea) format. A JFS2 file system using the v1 format can be converted to one using v2 format. After it is converted the file system cannot be converted back to v1. The conversion is done in an on-demand manner such that any extended attribute or ACL writes cause the conversion for that file object to occur. The v2 format provides support for scalable named extended attributes as well as support for NFS4 ACLs. The v1 format is compatible with prior releases of AIXoperating system. **-a efs=yes** Converts a file system to an Encrypted File System (EFS). The **chfs** command changes an existing file system into an EFS file system. When the file system is EFS enabled, the **ea** attribute is automatically converted to store scalable extended attributes (**v2**). This command fails if you have not run the **efsenable** command on the system. **Restriction**: The **chfs** commands prevents conversion of the following file systems (mount points) to EFS because the security infrastructures (kernel extensions, libraries and so on) are not available during boot: • **/** • **/usr** • **/var** • **/opt** **-a freeze = { timeout | 0 | off }** Specifies that the file system must be frozen or thawed, depending on the value of `timeout`. The act of freezing a file system produces a nearly consistent on-disk image of the file system, and writes all dirty file system metadata and user data to the disk. In its frozen state, the file system is read-only, and anything that attempts to modify the file system or its contents must wait for the freeze to end. The value of `timeout` must be either `0`, `off`, or a positive number. If a positive number is specified, the file system is frozen for a maximum of `timeout` seconds. If `timeout` is `0` or `off`, the file system will be thawed, and modifications can proceed. **Attention**: Freezing base file systems (**/**, **/usr**, **/var**, **/tmp**) can result in unexpected behavior.

Item	Description
-a [log | logname]=LVName	Specifies the full path name of the filesystem logging logical volume name of the existing log to be used. The log device for this filesystem must reside on the same volume group as the filesystem. Keyword INLINE can be used to specify that the log is in the logical volume with the JFS2 file system. The file system must have been created with an INLINE log to use this option. This option updates the **/etc/filesystems** file so that if the name of the logical volume containing the file system changes the log will be recognized. **Note:** For a file system using OUTLINE log, this option can be used to change the outline log from one logical volume to another logical volume as long as the logical volume is properly formatted and the type of the logical volume is jfs2log. If a file systems is mounted at the time `chfs` is called to change the outline log, the `/etc/filesystems` file will show the change, but the actual log will not be changed until the next mount for the file system (which follows a `umount` operation or a system crash and recovery). For a file system using INLINE log, this option does not support switching logs between INLINE and OUTLINE log. Currently, to switch from inlinelog to outlinelog (or vise versa), the file system has to be removed and recreated. In release AIX 5L and AIX 5.1, if the file system is using inlinelog, the log entry is the same as the file system in **/etc/filesystems** file: `/j2.1:` `dev = /dev/fslv00` `vfs = jfs2` `log = /dev/fslv00` `mount = false` `account = false` But, from AIX 5.2 and later releases, if the file system is using inlinelog, the log entry is the keyword INLINE in **/etc/filesystems** file: `/j2.23:` `dev = /dev/fslv04` `vfs = jfs2` `log = INLINE` `mount = false` `options = rw` `account = false` If the file system was created at AIX 5L or AIX 5.1, and later upgraded to AIX 5.2 or later releases, then `chfs` can be used to alter the inlinelog name in **/etc/filesystems** file.
-a logsize=LogSize	Specifies the size for an INLINE log in MBytes. The input size must be a positive value. If the inline log size is greater than or equal to 1, the input size must be an integer. If the input is floating point value of less than 1 and greater than or equal to 0, the input size is ignored and the default inline log size is taken. If value begins with a + (plus sign), it is interpreted as a request to increase the INLINE log size by the specified amount. If value begins with a - (minus sign), it is interpreted as a request to reduce the INLINE log size by the specified amount. The input is ignored if an INLINE log not being used. The INLINE log size cannot be greater than 10% of the size of the file system and it cannot be greater than 2047 MB.

Item	Description
	-a managed={yes | no} Enables Data Management Application Programming Interface (DMAPI) on a JFS2 file system.
	-a maxext=Value Specifies the maximum size of a file extent in file system blocks. A zero value implies that the JFS2 default maximum should be used. Values less than 0 or exceeding maximum supported extent size of 16777215 are invalid. Note that existing file extents are not affected by this change.
	-a mountguard={yes | no} Guards the file system against the unsupported concurrent mounts in a PowerHA® SystemMirror® or other clustering environment. If the mountguard is enabled, the file system cannot be mounted if it appears to be mounted on another node or system. To temporarily override the mountguard setting, see the **noguard** option of the **mount** command.
	-a options = mountOptions Specifies which `mount` option is passed into the `chfs` command. For a list of the valid options, refer to the **mount** command.
	-a reclaim={normal | fast} If the `normal` option is chosen, the **reclaim** command packs the filesystem as much as possible. The **reclaim** command looks for the biggest contiguous chunk of free space and then reclaims as much of it as it can. This makes the reclaimed free space available for reuse elsewhere in the system. However, when you use the `normal` option for the **reclaim** command, the file system becomes frozen. Therefore, if large amount of data is packed, the freeze time can be significant. If the `fast` option is chosen, the **reclaim** command looks for the biggest contiguous chunk of free space and then reclaims as much of it as it can. This makes the reclaimed free space available for reuse elsewhere in the system. It is not possible to determine exactly how much free space is recovered by the **reclaim** command. In order to get a rough estimate of the space reclaimed before running the **chfs** command, which will actually reclaim the space, you must first run, `lvmstat –v <volume group> –e`, and then after the **chfs** command finishes, run `lvmstat –v <volume group> –r`. The first **lvmstat** command enables statistic collection for that volume group, and the second prints out the recorded statistics. All of the disks in the file system must support the reclaim operation. The reclaim operation does not alter the actual file system size. The reclaim option cannot be specified if there are snapshots in the file system and cannot be run while live update is running, or if the file system is read-only. The reclaim option cannot be used along with any file system resize operation. Live update will not start if this command is in progress.
	-a refreeze={timeout} Specifies that the timeout for a frozen file system be reset. The `timeout` is reset to the value specified. The file system must still be frozen (using the `-a freeze` option or the `fscntl` interface).

Item	Description

-a size=NewSize

Specifies the size of the Enhanced Journaled File System in 512-byte blocks, megabytes or gigabytes. If Value has the M suffix, it is interpreted to be in megabytes. If Value has a G suffix, it is interpreted to be in gigabytes. If Value begins with a +, it is interpreted as a request to increase the file system size by the specified amount. If Value begins with a -, it is interpreted as a request to reduce the file system size by the specified amount.

If the specified size does not begin with a + or -, but it is greater or smaller than the file system current size, it is also a request to increase or reduce the file system size.

If the file system has an `inlinelog`, the `inlinelog` size remains unchanged if the new size of this file system is the same as the current file system size. If the specified size is not evenly divisible by the physical partition size, it is rounded up to the closest number that is evenly divisible. If the file system is on a striped logical volume, the size of the new file system is rounded to the nearest multiple of the striping width multiplied by the physical partition size. The striping width is the number of hard disks that form the striped logical volume.

This attribute is required when creating a JFS2 file system unless the **-d** flag has been specified. The volume group in which the file system resides defines a maximum logical volume size and limits the file system size. The maximum size is determined by the file system block size:

```
fs block size (byte)        MAX fssize (TB)
========================================
512                         4
1024                        8
2048                        16
4096                        32
```

When a request to reduce the file system size is successful, the logical volume should be equal to or smaller than the original LV size depending on the requested filesystem size.

Both *size* and *logsize* attributes can be specified in one `chfs` request to resize the filesystem and its `inlinelog` sizes.

Note: The file system might be frozen for a significant time during the shrink operations. To minimize the impact on applications, you must shrink the file system in small amounts and during low workloads.

-a vix={yes|no}

Specifies whether the file system can allocate inode extents smaller than the default of 16 KB if there are no contiguous 16 KB extents free in the file system. After a file system is enabled for small free extents, it cannot be accessed on earlier versions of AIX and the marking cannot be removed.

yes
File system can allocate variable length inode extents.

no
File system must use default size of 16 KB for inode extents. This has no effect if the file system already contains variable length inode extents.

Item	Description
	Note: 1. JFS2 does not have nbpi or fragment size values to affect the resulting size of the file system. 2. You cannot shrink a file system if the requested size is less that a physical partition size. At least one physical partition size is asked to be reduced. 3. Shrinking a file system that has snapshots is not allowed. 4. During the shrink operation of the filesystem, the write operations to the file system might be restricted intermittently. 5. The file system is not accessible when the extend operation is running. Large file systems with inline logs might not be usable for several minutes. The inline log must be reformatted. 6. When the new file system size is specified, but its `inlinelog` size is NOT specified, the new *logsize* will be adjusted (extended/shrunk) proportionally, based on the specified extended/shrunk file system size. The log size increase or reduction should not be more than 40% of the file system size increase or reduction. 7. When a new file system size is not specified and there is an `inlinelog`, if a new *logsize* is specified, the file system size might be changed to include the new log size. 8. The freed space reported by the **df** command is not necessary the space that can be truncated by a `shrink`FS request due to filesystem fragmentation. A fragmented filesystem may not be shrunk if it does not have enough free space for an object to be moved out of the region to be truncated, and `shrink`FS does not perform filesystem defragmentation. In this case, the **chfs** command should fail with the returned code 28 (ENOSPC) 9. The **maxext** attribute is ignored in older releases even if the filesystem was created with it on a later release. 10. In AIX 7.2 Technology Level 1, or later, after the partition is freed by running the **chfs** command, the space reclamation process is started on the freed partition.
`-A`	Specifies the attributes for auto-mount. **yes** File system is automatically mounted at system restart. **no** File system is not mounted at system restart.
`-d` *Attribute*	Deletes the specified attribute from the `/etc/filesystems` file for the specified file system.
`-m` *NewMountPoint*	Specifies a new mount point for the specified file system.
`-n` *NodeName*	Specifies a node name for the specified file system. The node name attribute in the `/etc/filesystems` file is updated with the new name. The node name attribute is specific to certain remote virtual file system types, such as the NFS (Network File System) virtual file system type.

Item	Description
-p	Sets the permissions for the file system. **ro** Specifies read-only permissions. **rw** Specifies read-write permissions.
-t	Sets the accounting attribute for the specified file system. **yes** File system accounting is to be processed by the accounting subsystem. **no** File system accounting is not to be processed by the accounting subsystem; this is the default.
-u *MountGroup*	Specifies the mount group. Mount groups are used to group related mounts, so that they can be mounted as one instead of mounting each individually. For example, when performing certain tests, if several scratch file systems always need to be mounted together, they can each be placed in the `test` mount group. They can then all be mounted with a single command, such as the `mount -t test` command.

Security

Access Control

Only the root user or a member of the **system** group can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To change the file system size of the `/test` Journaled File System, enter:

   ```
   chfs  -a size=24576 /test
   ```

   This command changes the size of the `/test` Journaled File System to 24576 512-byte blocks, or 12MB (provided it was previously no larger than this).
2. To increase the size of the `/test` Journaled File System, enter:

   ```
   chfs  -a size=+8192 /test
   ```

   This command increases the size of the `/test` Journaled File System by 8192 512-byte blocks, or 4 MB.
3. To convert a JFS2 file system to a version which can support NFS4 ACLs, type:

   ```
   chfs -a ea=v2 /test
   ```
4. To change the mount point of a file system, enter:

   ```
   chfs  -m /test2 /test
   ```

   This command changes the mount point of a file system from `/test` to `/test2`.
5. To delete the accounting attribute from a file system, enter:

```
chfs -d account /home
```

This command removes the accounting attribute from the **/home** file system. The accounting attribute is deleted from the `/home:` stanza of the **/etc/filesystems** file.

6. To split off a copy of a mirrored file system and mount it read-only for use as an online backup, enter:

```
chfs -a splitcopy=/backup -a copy=2 /testfs
```

This mount a read-only copy of `/testfs` at `/backup`.

7. To change the file system size of the `/test` Journaled File System, enter:

```
chfs -a size=64M /test
```

This command changes the size of the `/test` Journaled File System to 64MB (provided it was previously no larger than this).

8. To reduce the size of the `/test` JFS2 file system, enter:

```
chfs  -a size=-16M /test
```

This command reduces the size of the `/test` JFS2 file system by 16MB.

9. To freeze a file system, enter:

```
chfs  -a freeze=60 /adl
```

This command freezes the `/adl` file system for a maximum of 60 seconds.

10. To thaw a file system, enter:

```
chfs  -a freeze=off /zml
```

This command thaws the `/zml` file system.

File

Item	Description
/etc/filesystems	Lists the known file systems and defines their characteristics.

chgif Method

Purpose

Reconfigures an instance of a network interface.

Syntax

chgif [**-d** | **-T**] **-l** *InterfaceInstance* **-a "***Attribute***=***Value ...***"**

Description

The **chgif** method first modifies the database and then reconfigures the specified network interface instance (*InterfaceInstance*) by issuing a call to the **ifconfig** command. Only one interface can be changed per command invocation, and at least one attribute must be specified. This method is not normally used on the command line. Rather, it is called by high-level commands.

Note: The **chgif** method is a programming tool and it must not be run from the command line. The **chdev** command must be used to change the network interface, which invokes the **chgif** method internally.

Flags

Item	Description
-a"*Attribute***=***Value ...***"**	Specifies pairs of attributes and values that configure the Interface instance. The *AttributeValue* pairs must be surrounded by quotes. Valid attribute values are as follows: **netaddr** Specifies the Internet address of the network interface. **netaddr6** Specifies the IPv6 Internet address of the network interface. **prefixlen** Specifies the prefix length of the IPv6 Internet address of the network interface. **alias4** Specifies the IPv4 Internet address alias of the network interface **alias6** Specifies the IPv6 Internet address alias of the network interface. **delalias4** Deletes the IPv4 Internet address alias of the network interface. **delalias6** Deletes the IPv6 Internet address alias of the network interface. **state (up/down)** Marks the interface as up or down. **trailers (on/off)** Turns the trailer link-level encapsulation on or off. **arp (on/off)** Enables or disables the use of the Address Resolution Protocol. **allcast (on/off)** Specifies whether to broadcast packets to all token-ring networks or just the local token-ring network. This attribute applies only to token-ring networks. **hwloop (on/off)** Enables or disables hardware loopback mode. **netmask** Specifies the network mask in dotted-decimal format. **security *SecurityLevelKeyword*** (**inet** only) Specifies the security level associated with the interface. The value of the *SecurityLevelKeyword* variable can be one of the following: • **none** • **unclassified** • **confidential** • **secret** • **top_secret** When the level of security is defined as **none** or **unclassified**, no IP Option header is added to the IP header.

Item	Description
	authority *AuthorityLevelKeyword* (**inet** only) Specifies the security authority level associated with the interface. The value of the *AuthorityLevelKeyword* variable can be one or more of the following: **genser** Defense Communications Agency **siop** Department of Defense Organization of the Joint Chiefs of Staff **dsccs-spintcom** Defense Intelligence Agency **dsccs-criticom** National Security Agency When more than one level of authority is specified, the values are separated by commas without embedded spaces. **mtu** Maximum IP packet size for this system. **broadcast** Specifies the address to use for representing broadcasts to networks. **dest** Specifies the destination address on a point-to-point link.
-d	Specifies that changes are made only in the configuration database. Changes take effect at the next system restart.
-l *InterfaceInstance*	Specifies the instance of the network interface to be reconfigured.
-T	Makes a temporary change in the device without the change being reflected in the database. It is temporary in that the device reverts to the characteristics described in the database when the system is restarted.

Examples

1. To add the netaddr=10.3.4.2 Object Data Manager (ODM) entry to the en2 standard Ethernet network interface with netmask=255.255.255.0, enter the following command:

   ```
   chdev -l en2 -a netaddr=10.3.4.2 -a netmask=255.255.255.0
   ```

   A message that is similar to the following example is displayed:

   ```
   en2 changed
   ```

2. To add the alias4=10.3.4.3 ODM entry to the en2 standard Ethernet network interface, enter the following command:

   ```
   chdev -l en2 -a alias4=10.3.4.3,255.255.255.0
   ```

   A message that is similar to the following example is displayed:

   ```
   en2 changed
   ```

3. To delete the alias4=10.3.4.3 ODMentry from the en2 standard Ethernet network interface, enter the following command:

   ```
   chdev -l en2 -a delalias4=10.3.4.3
   ```

A message that is similar to the following example is displayed:

```
en2 changed
```

4. To add the netaddr6=fe80::20b4:40ff:fe00:f012 ODM entry to the en2 standard Ethernet network interface with prefixlen=64, enter the following command:

   ```
   chdev -l en2 -a netaddr6=fe80::20b4:40ff:fe00:f012 -a prefixlen=64
   ```

   A message that is similar to the following example is displayed:

   ```
   en2 changed
   ```

5. To add the alias6=fe80::20b4:40ff:fe00:f016/64 ODM entry to the en3 standard Ethernet network interface, enter the following command:

   ```
   chdev -l en3 -a alias6=fe80::20b4:40ff:fe00:f016/64
   ```

   A message that is similar to the following example is displayed:

   ```
   en3 changed
   ```

6. To delete the alias6=fe80::20b4:40ff:fe00:f016/64 ODM entry from the en3 standard Ethernet network interface, enter the following command:

   ```
   chdev -l en3 -a delalias6=fe80::20b4:40ff:fe00:f016/64
   ```

   A message that is similar to the following example is displayed:

   ```
   en3 changed
   ```

chginet Method

Purpose

Reconfigures the Internet instance.

Syntax

chginet [**-d**] [**-a"***Attribute***=***Value*...**"**]

Description

The **chginet** method reconfigures the Internet instance, and can also change the *HostName* variable and any static routes that are defined. The **chginet** method calls the **hostname** command to change the host name. The **chginet** method also calls the **route** command to change any static routes. The **chdev** command calls method.

Note: The **chginet** method is a programming tool and should not be entered from the command line.

Flags

Item	Description
-a"*Attribute***=***Value* ..."	Specifies the customized attributes of the Internet instance. The following are valid attributes: **hostname** Specifies the name of the host. **gateway** Specifies the default gateway. **route** Specifies the route. The format of the *Value* variable of the **route** attribute is: route = *type*, [*args,*], *destination*, *gateway*, [*metric*] . The value of the *type* parameter can be **net** or **host**. **delroute** Specifies the route to delete. The format of the *Value* variable of the **delroute** attribute is: delroute = *type*, [*args*,], *destination*, *gateway*, [*metric*] . The value of the *type* parameter can be **net** or **host**. **rout6** Specifies the IPv6 route. The format of the *Value* variable of the **rout6** attribute is: rout6 = *type*, [*args*,], *destination*, *gateway*, [*metric*] The value of the *type* parameter can be **net** or **host**. **delrout6** Specifies the IPv6 route to delete. The format of the *Value* variable of the **delrout6** attribute is: delrout6 = *type*, [*args*,], *destination*, *gateway*, [*metric*] The value of the *type* parameter can be **net** or **host**.
-d	Specifies that changes are made only in the configuration database. Changes take effect with the next IPL.

Examples

1. To change an Internet instance and specify a route, enter a method in the following format:

   ```
   chginet -a"route=192.9.200.0,bcroom"
   ```

   This example specifies a new route. The new route is being set to network `192.9.200.0`, the `bcroom` gateway.
2. This example specifies a new route. The new route is being set to host `192.9.200.5` with `hopcount` 2, interface `en0`, and the *bcroom* gateway.

   ```
   chginet -a"route=host,-hopcount,2,-if,en0,192.9.200.5,bcroom"
   ```
3. This example deletes the route added in the previous example.

   ```
   chginet -a"delroute=host,-hopcount,2,-if,en0,192.9.200.5,bcroom"
   ```
4. This example specifies a new IPv6 route. The new route is being set to host `2001::1` with `hopcount` 2, interface `en0`, and the `fe80::20b4:40ff:fe00:f016` gateway.

   ```
   chginet -a"rout6=host,-hopcount,2,-if,en0,2001::1,fe80::20b4:40ff:fe00:f016"
   ```
5. This example deletes the IPv6 route added in the previous example.

   ```
   chginet -a"delrout6=host,-hopcount,2,-
   if,en0,2001::1,fe80::20b4:40ff:fe00:f016"
   ```

chgroup Command

Purpose

Changes attributes for groups.

Syntax

chgroup [**-R** *load_module*] *Attribute*=*Value ... Group*

Description

Attention: Do not use the **chgroup** command if you have a Network Information Service (NIS) database installed on your system, as this could cause serious system database inconsistencies.

The **chgroup** command changes attributes for the group specified by the *Group* parameter. The group name must already exist. To change an attribute, specify the attribute name and the value you want to change it to in the *Attribute*=*Value* parameter.

To change the attributes for a group that was created with an alternate Identification and Authentication (I&A) mechanism, the **-R** flag can be used to specify the I&A loadable module. Load modules are defined in the **/usr/lib/security/methods.cfg** file.

You could also use the System Management Interface Tool (SMIT) **smit chgroup** fast path to run this command.

Changing the ID for an account can compromise system security and as a result one should not do so. However, when the ID is changed using the **chgroup** command, ID collision checking is also controlled by the **dist_uniqid** attribute in the usw stanza of the `/etc/secvars.cfg` file. The behavior of ID collision control is the same as that described for the **mkgroup** command.

Restrictions on Changing Groups

To ensure the security of group information, there are restrictions on using the **chgroup** command. Only the root user or users with UserAdmin or aix.security.group.change authorization can use the **chgroup** command to change any group. These changes include:

- Make a group an administrative group by setting the **admin** attribute to true.
- Change any attributes of an administrative group.
- Add users to an administrative group's administrators list.

An administrative group is a group with the **admin** attribute set to true. Members of the **security** group can change the attributes of nonadministrative groups including adding users to the list of administrators.

Flag

Item	Description
-R	Specifies the loadable I&A module used to change user's attributes.

Attributes

You change attributes by specifying an *Attribute*=*Value* parameter. If you have the proper authority you can set the following group attributes:

Item	Description
adms	Defines the users who can perform administrative tasks for the group, such as setting the members and administrators of the group. This attribute is ignored if **admin = true**, since only the root user can alter a group defined as administrative. The *Value* parameter is a list of comma-separated user login names. If you do not specify a *Value* parameter, all the administrators are removed.

Item	Description
admin	Defines the administrative status of the group. You can specify the following values: **true** Defines the group as administrative. Only the root user can change the attributes of groups defined as administrative. **false** Defines a standard group. The attributes of these groups can be changed by the root user or a member of the security group. This is the default value.
id	The group ID. The *Value* parameter is a unique integer string. Changing this attribute compromises system security and, for this reason, you should not change this attribute.
projects	Defines the list of projects to which the user's processes can be assigned. The value is a list of comma-separated project names and is evaluated from left to right. The project name should be a valid project name as defined in the system. If an invalid project name is found on the list, it will be reported as an error.
users	Specifies a list of one or more users in the form: *User1*, *User2*,...**,** *Usern*. The group member names are separated by commas. Each user must be defined in the database configuration files. You cannot remove users from their primary group. If the **domainlessgroups** attribute is set in the **secvars.cfg** file, users from the Lightweight Directory Access Protocol (LDAP) group can be assigned to the local group and vice versa.
efs_initialks_mode	Specifies the initial mode of the group keystore. You can specify the following values: **admin** Root or other security privileged system users can open the group keystore using the admin key. **guard** Root users cannot open the group keystore using the admin key. The default value is **admin**. The attribute specifies the initial mode of the group keystore. You can use the attribute with the **mkgroup** command. After the keystore has been created, changing the attribute value with the **chuser**, **chgroup**, or **chsec** command, or manual editing does not change the mode of the keystore unless the keystore is deleted and a new one is created. To change the keystore mode, use the **efskeymgr** command. **Restriction**: The attribute is valid only when the system is EFS-enabled.

Item	Description
efs_keystore_algo	Specifies the algorithm that is used to generate the private key of the group during the keystore creation. You can specify the following values: • **RSA_1024** • **RSA_2048** • **RSA_4096** The default value is **RSA_1024**. You can use the attribute with the **mkgroup** command. After the keystore has been created, changing the value of this attribute with the **chuser**, **chgroup**, or **chsec** command, or manual editing does not regenerate the private key unless the keystore is deleted and a new one is created. To change the algorithm for the keys, use the **efskeymgr** command. **Restriction**: The attribute is valid only when the system is EFS-enabled.
efs_keystore_access	Specifies the database type of the group keystore. You can specify the following values: **file** Creates the **/var/efs/groups/*grpname*/keystore** keystore file associated with the group. **none** The keystore is not created. All other keystore attributes have no effect. The default value is **file**. **Restriction**: The attribute is valid only when the system is EFS-enabled.

The **adms** and **admin** attributes are set in the **/etc/security/group** file. The remaining attributes are set in the **/etc/group** file. If any of the attributes you specify with the **chgroup** command are invalid, the command makes no changes at all.

Exit Status

This command returns the following exit values:

Item	Description
0	The command runs successfully and all requested changes are made.
>0	An error occurred. The printed error message gives further details about the type of failure.

Security

Access Control

This command should grant execute (x) access only to the root user and the security group. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the root user with the **setuid** (SUID) bit set.

Auditing Events

Event	Information
GROUP_Change	group, attributes

Files Accessed

Mode	File
rw	/etc/group
rw	/etc/security/group
r	/etc/passwd

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Limitations

Changing a group's attributes may not be supported by all loadable I&A modules. If the loadable I&A module does not support changing a group's attributes, an error is reported.

Examples

1. To add `sam` and `carol` to the `finance` group, which currently only has `frank` as a member, type:

   ```
   chgroup users=sam,carol,frank  finance
   ```

2. To remove `frank` from the `finance` group, but retain `sam` and `carol`, and to remove the administrators of the `finance` group, type:

   ```
   chgroup users=sam,carol adms= finance
   ```

   In this example, two attribute values were changed. The name `frank` was omitted from the list of members, and the value for the `adms` attribute was left blank.

3. To change the LDAP I&A loadable module group user's attribute, type:

   ```
   chgroup -R LDAP users=sam,frank monsters
   ```

Files

Item	Description
/usr/bin/chgroup	Specifies the path to the **chgroup** command.
/etc/group	Contains the basic attributes of groups.
/etc/security/group	Contains the extended attributes of groups.
/etc/passwd	Contains the basic attributes of users.

chgrp Command

Purpose

Changes the group ownership of a file or directory.

Syntax

chgrp [**-f**] [**-h**] [**-R**] *Group* { *File ...* | *Directory ...* }

chgrp -R [**-f**] [**-H** | **-L** | **-P**] *Group* { *File...* | *Directory...* }

Description

The **chgrp** command changes the group of the file or directory specified by the *File* or *Directory* parameter to the group specified by the *Group* parameter. The value of the *Group* parameter can be a group name from the group database or a numeric group ID. When a symbolic link is encountered and you have not specified the **-h** or **-P** flags, the **chgrp** command changes the group ownership of the file or directory pointed to by the link and not the group ownership of the link itself.

Although the **-H**, **-L** and **-P** flags are mutually exclusive, specifying more than one is not considered an error. The last flag specified determines the behavior that the command will exhibit.

If you specify the **-h** flag, the **chgrp** command has the opposite effect and changes the group ownership of the link itself and not that of the file or directory pointed to by the link.

If you specify both the **-h** flag and the **-R** flag, the **chgrp** command descends the specified directories recursively, and when a symbolic link is encountered, the group ownership of the link itself is changed and not that of the file or directory pointed to by the link.

Flags

Item	Description
-f	Suppresses all error messages except usage messages.
-h	Changes the group ownership of an encountered symbolic link and not that of the file or directory pointed to by the symbolic link.
-H	If the **-R** option is specified and a symbolic link referencing a file of type directory is specified on the command line, **chgrp** shall change the group of the directory referenced by the symbolic link and all files in the file hierarchy below it.
-L	If the **-R** option is specified and a symbolic link referencing a file of type directory is specified on the command line or encountered during the traversal of a file hierarchy, **chgrp** shall change the group of the directory referenced by the symbolic link and all files in the file hierarchy below it.
-P	If the **-R** option is specified and a symbolic link is specified on the command line or encountered during the traversal of a file hierarchy, **chgrp** shall change the group ID of the symbolic link if the system supports this operation. The **chgrp** utility shall not follow the symbolic link to any other part of the file hierarchy.
-R	Descends directories recursively, setting the specified group ID for each file. When a symbolic link is encountered and the link points to a directory, the group ownership of that directory is changed but the directory is not further traversed. If the **-h**, **-H**, **-L** or **-P** flags are not also specified, when a symbolic link is encountered and the link points to a directory, the group ownership of that directory is changed but the directory is not traversed further.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To change the group ownership of the file or directory named `proposals to staff`:

   ```
   chgrp staff proposals
   ```

   The group access permissions for `proposals` now apply to the `staff` group.
2. To change the group ownership of the directory named `proposals,` and of all the files and subdirectories under it, to `staff`:

   ```
   chgrp -R staff proposals
   ```

   The group access permissions for `proposals` and for all the files and subdirectories under it now apply to the `staff` group.

Files

Item	Description
/usr/bin/chgrp	The **chgrp** command
/etc/group	File that identifies all known groups

chgrpmem Command

Purpose

Changes the administrators or members of a group.

Syntax

chgrpmem [-R *load_module*] [{ **-a** | **-m** } { **+** | **-** | **=** } *User* ...] *Group*

Description

The **chgrpmem** command changes the administrators or members of the group specified by the *Group* parameter. Use this command to add, delete, or set a group's members or administrators list. You cannot remove users from their primary group. A user's primary group is maintained in the **/etc/passwd** file. If you specify only a group with the **chgrpmem** command, the command lists the group's members and administrators.

To change the administrators or members of a group that were created with an alternate Identification and Authentication (I&A) mechanism, the `-R` flag can be used to specify the I&A loadable module. Load modules are defined in the `/usr/lib/security/methods.cfg` file.

To add, delete, or set a user as a group administrator, specify the **-a** flag. Otherwise, to add, delete, or set a user as a group member, specify the **-m** flag. You must specify one of these flags and an operator to change a user's group membership. The operators do the following:

Item	Description
+	Adds the specified user.
-	Deletes the specified user.
=	Sets the list of administrators or members to the specified user.

You can specify more than one *User* parameter at a time. To do this, specify a comma-separated list of user names.

See the **chgroup** command for a list of restrictions that apply to changing group information.

Flags

Item	Description
-a	Changes a group's administrators list.
-m	Changes the group's members list.
-R	Specifies the loadable I&A module used to change the administrators or members of a group.

Exit Status

This command returns the following exit values:

Item	Description
0	The command runs successfully and all requested changes are made.
>0	An error occurred. The printed error message gives further details about the type of failure.

Security

Access Control

All users should have execute (x) access to this command because the command itself enforces the access rights. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the **security** group with the **setgid** (SGID) bit set.

Files Accessed

Item	Description
Mode	**File**
x	**/usr/bin/chgroup**
r	**/etc/passwd**
r	**/etc/group**
rw	**/etc/security/group**

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To remove `jones` as an administrator of the `f612` group, enter:

   ```
   chgrpmem -a - jones f612
   ```

2. To add members `davis` and `edwards` to group `f612`, enter:

   ```
   chgrpmem -m + davis,edwards f612
   ```

3. To list members and administrators of group `staff`, enter:

   ```
   chgrpmem staff
   ```

4. To list members of the LDAP I&A loadable module group `monsters`, enter:

```
chgrpmem -R LDAP monsters
```

Files

Item	Description
/usr/bin/chgrpmem	Specifies the path to the **chgrpmem** command.
/etc/passwd	Contains the basic attributes of users.
/etc/group	Contains the basic attributes of groups.
/etc/security/group	Contains the extended attributes of groups.

chhwkbd Command

Purpose

Changes keyboard attributes stored in the Object Data Manager (ODM) database.

Syntax

chhwkbd [**-d** *Delay*] [**-r** *Repetition*] [**-c** *ClickerVolume*] [**-a** *AlarmVolume*] [**-m** ["KR" | "JP" | "TW"]] [**-t** ["nonum"]]

Description

The **chhwkbd** command changes the following keyboard attributes stored in the ODM database:

- Repetition delay
- Repetition rate
- Clicker volume
- Alarm volume
- Korean, Japanese, and Chinese keyboard identification
- Numeric pad emulation enable/disable

Changes to the keyboard attributes take effect after system restart.

You could also use the System Management Interface Tool (SMIT) **smit chgkbd** fast path to run this command.

Flags

Item	Description
-a *AlarmVolume*	Sets the alarm volume to the specified value. Values for the *AlarmVolume* variable are defined below: **0** off **1** low **2** medium **3** high
-c *ClickerVolume*	Sets the clicker volume to the specified value. Values for the *ClickerVolume* variable are defined below: **0** off **1** low **2** medium **3** high
-d *Delay*	Sets the keyboard repetition delay to the specified value. The *Delay* variable can be 250, 500, 750, or 1000 msec. The default value is 500 msec.
-m ["KR" | "JP" | "TW"]	Provides extended keyboard identification for the following keyboards: **"KR"** Korean keyboard **"JP"** Japanese keyboard **"TW"** Chinese keyboard Use the **-m** flag without specifying a value to remove extended keyboard identification. **Note:** This flag is valid only when an IBM RS/6000® 106-key keyboard or an IBM PS/2 keyboard or equivalent keyboard is attached to the workstation. The **-m** flag is set automatically when the locale is selected using SMIT.
-r *Repetition*	Sets the rate of repetition to the specified value. The *Repetition* variable can be an integer from 2 to 30 inclusive. The default value is 11 characters per second.

Item	Description
-t ["nonum"]	Enables or disables numeric pad emulation. To enable numeric pad emultaion, specify the "nonum" parameter. Use the **-t** flag without specifying a value to disable numeric pad emulation.

Notes:

1. This flag is valid only when an IBM PS/2 keyboard or equivalent keyboard is attached to the workstation.
2. "nonum" means no numeric keypad.

Examples

1. To change the keyboard repetition delay rate to 250 msec, enter:

   ```
   chhwkbd  -d 250
   ```

2. To change the keyboard repetition rate to 30 characters per second, enter:

   ```
   chhwkbd  -r 30
   ```

File

Item	Description
/usr/bin/chhwkbd	Contains the **chhwkbd** command.

chiscsi Command

Purpose

Changes iSCSI target data.

Syntax

`chiscsi -l` *AdapterName* `-g static -t` *TargetName* [`-n` *PortNumber* `-i` *IPaddress*] [`-p` *password*] [`-u` *username*] [`-T` *NewTargetName*] [`-N` *NewPortNumber*] [`-I` *NewIPaddress*]

`chiscsi -l` *AdapterName* `-g auto -t` *TargetName* [`-p` *password*] [`-u` *username*] [`-T` *NewTargetName*]

Description

The `chiscsi` command changes iSCSI target data in ODM. There are two categories of data stored in ODM. The first is for statically configured iSCSI targets, which require that all the relevant iSCSI target information (such as target name, IP address, and port number) are specified in order for AIX to discover them. The 2nd category of iSCSI target data is for iSCSI target devices that can be configured automatically, but require authentication from the host (such as passwords). These two categories of iSCSI target data are associated with the `static` and `auto` groups, respectively, specified by the `-g` flag.

Flags

Item	Description
-g *group*	Specifies which group this iSCSI target is associated with. There two valid groups are `static` and `auto`. The `static` group is for iSCSI targets that cannot be automatically discovered from this host; all relevant iSCSI target information for them (such as target name, IP address, and port number) must be specified. The `auto` group is for iSCSI targets that are automatically discovered, but require authentication information such as passwords.
-I *NewIPaddress*	Specifies the new IP address of the iSCSI target when it is being changed.
-i *IPaddress*	Specifies the IP address of the iSCSI target.
-l *AdapterName*	Specifies the adapter name for the iSCSI TCP/IP Offload Engine (TOE) adapter that is attached to this iSCSI target. It can also specify the iSCSI protocol device for the iSCSI software solution device.
-N *NewPortNumber*	Specifies the new port number of the iSCSI target when it is being changed.
-n *NewPortNumber*	Specifies the port number on which the iSCSI target is accessed. The default port number is `3260`.
-p *password*	Specifies the new password for this iSCSI target.
-T *NewTargetName*	Specifies the new iSCSI target name when it is being changed.
-t *TargetName*	Specifies the iSCSI target name (for example, `iqn.sn9216.iscsi-hw1`).
-u *username*	Specifies the new Challenge Handshake Authentication Protocol (CHAP) user name that can be used for CHAP authentication.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

The `chiscsi` command is executable only by root.

Examples

1. To change the password of a statically configured iSCSI target to `my password`, enter:

```
chiscsi -l ics0 -g static -t qn.mds9216.iscsi_hw -n 3260 -i 10.1.2.116 -p "my password"
```

2. To change the IP address of a statically configured iSCSI target to 10.1.3.141, enter:

```
chiscsi -l ics0 -g static -t qn.mds9216.iscsi_hw -n 3260 -i 10.1.2.116 -I 10.1.3.141
```

Location

/usr/sbin/chiscsi

Files

Item	Description
src/bos/usr/sbin/iscsia	Contains the common source files from which the iSCSI commands are built.

chitab Command

Purpose

Changes records in the **/etc/inittab** file.

Syntax

chitab {*Identifier* **:** *RunLevel* **:** *Action* **:** *Command* }

Description

The **chitab** command changes a record in the **/etc/inittab** file. The *Identifier*:*Run Level*:*Action*:*Command* parameter string is the new entry to the **/etc/inittab** file. You can search for a specific record by using fields in the *Identifier* portion of the parameter string. The command finds the specified *Identifier* and changes that record.

Note: The **chitab** command can not comment out an entry in the **/etc/inittab** file.

Parameters

The *Identifier*:*Run Level*:*Action*:*Command* parameter string specifies a record in the **/etc/inittab** file where the following parameters apply:

Item	Description
Action	A 20-character parameter that informs the **init** command how to process the *Command* parameter you specify. The **init** command recognizes the following actions: **boot** Read this record only when the system boots and reads the **/etc/inittab** file. The **init** command starts the process. Do not wait for the process to stop, and when it does stop, do not restart the process. The run level for this process should be the default, or it must match the run level specified by the **init** command at startup time. **bootwait** Read this record only when the system boots and reads the **/etc/inittab** file. The **init** command starts the process. Wait for it to stop, and when it does stop, do not restart the process. **hold** When the process identified in this record is terminated, do not start a new one. The **hold** action can only be activated by the **phold** command.

Item	Description
	initdefault Start the process identified in this record only when the **init** command is originally invoked. The **init** command uses this line to determine which run level to originally enter. It does this by taking the highest run level specified in the *RunLevel* field and using that as its initial state. If the *RunLevel* parameter is empty, this is interpreted as 0123456789, and the **init** command enters a run level of **9**. If the **init** command does not find an **initdefault** line in the **/etc/inittab** file, it requests an initial run level from the operator at initial program load (IPL) time. **off** If the process identified in this record is currently running, send the warning signal **SIGTERM** and wait 20 seconds before sending the **SIGKILL** kill signal. If the process is nonexistent, ignore this line. **once** When the **init** command enters the run level specified for this record, start the process, do not wait for it to stop, and when it does stop, do not restart the process. If the system enters a new run level while the process is running, the process is not restarted. **ondemand** Functionally identical to **respawn**. If the process identified in this record does not exist, start the process. If the process currently exists, do nothing and continue scanning the **/etc/inittab** file. Specify this action to perform the **respawn** action when using **a**, **b**, or **c** run levels. **powerfail** Start the process identified in this record only when the **init** command receives a **SIGPWR** power fail signal. **powerwait** Start the process identified in this record only when the **init** command receives a **SIGPWR** power fail signal, and wait until it stops before continuing to process the **/etc/inittab** file. **respawn** If the process identified in this record does not exist, start the process. If the process currently exists, do nothing and continue scanning the **/etc/inittab** file. **sysinit** Start the process identified in this record before the **init** command tries to access the console. For example, you might use this to initialize devices. **wait** When the **init** command enters the run level specified for this record, start the process and wait for it to stop. While the **init** command is in the same run level, all subsequent reads of the **/etc/inittab** file ignore this object. If you are operating in a diskless environment, specifying the **wait** action causes your system to boot more quickly.
Command	A 1024-character field specifying the shell command.
Identifier	A 14-character parameter that uniquely identifies an object. The *Identifier* must be unique. If the *Identifier* is not unique, the command is unsuccessful. The *Identifier* cannot be changed; if you try to change it, the command is unsuccessful.
RunLevel	A 20-character parameter defining the run levels in which the *Identifier* can be processed. Each process started by the **init** command can be assigned one or more run levels in which it can be started.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To change the run level of a record for `tty2`, enter:

```
chitab "tty002:23:respawn:/usr/sbin/getty /dev/tty"
```

The quotes are required when the record being added has spaces or tabs.

Files

Item	Description
/etc/inittab	Indicates which processes the **init** command starts.

chkbd Command

Purpose

Changes the software keyboard map to be loaded into the system at the next IPL (Initial Program Load).

Syntax

chkbd *KeyMapPathName*

Description

The **chkbd** command changes the default software keyboard map loaded at system IPL. The *KeyMapPathname* parameter provides the location of the software keymap file. This pathname can be absolute or simply the filename. If only the filename is specified then the command will look for it in the default directory **/usr/lib/nls/loc**.

Note: This command can be used only on an LFT display.

For a list of all available keyboard maps, use the **lskbd** command.

You could also use the System Management Interface Tool (SMIT) **smit chkbd** fast path to run this command.

Parameter

Item	Description
KeyMapPathName	Provides the location of the software keymap file.

Files

Item	Description
/bin/chkbd	Contains the **chkbd** command.
/usr/lib/nls/loc	Contains the keyboard directory.

chkey Command

Purpose

Changes your encrypting key.

Syntax

/usr/bin/chkey

Description

The **chkey** command prompts you for a password and uses it to encrypt a new encryption key. Once the key is encrypted, the **ypupdated** daemon updates the **/etc/publickey** file.

chlang Command

Purpose

Changes the language settings for system or user.

Syntax

To Modify the Environment or Profile File Changing the Default Language Setting:

chlang [**-u** *UID* | *Uname*] [**-m** *MsgTransLst* | **-M**] *Language*

To Modifiy the Environment or Profile File without Changing the Default Language Setting:

chlang [**-u** *UID* | *Uname*] **-m** *MsgTransLst* | **-M**

To Remove the NLSPATH Setting from the Environment or Profile File:

chlang -d [**-u** *UID* | *UName*]

Description

The **chlang** command is a high-level shell command that changes the language settings for either the entire system or an individual user. If the effective id of the invoker is root and the **-u** option was not used, the language settings will be changed for the entire system in the **/etc/environment** file. If the effective id of the invoker is not root, or if the **-u** option was used, the language settings will be changed for an individual user in the user's **.profile** file.

When **chlang** is run with a language and no options, the **LANG** environment variable will be set to the language specified.

When chlang is run with the **-m** option, the **LANG** and **NLSPATH** environment variables will be set. In addition, the **LC_MESSAGES** variable will be set to the first value specified in the *MsgTransLst* of the **-m** flag if it is different from the **Language** parameter and the **Language** parameter has a system supplied translation available.

When **chlang** is run with the **-d** option, the **NLSPATH** environment variable will be removed.

Notes:

1. Changes made to the NLS environment by **chlang** are not immediate when either **/etc/environment** or the user's **.profile** are modified. Changes to **/etc/environment** requires rebooting the sytsem. Changes to a user's **.profile** requires logging in again or running the **.profile** file.
2. When modifying a user's configuration file, if the user uses the C shell (**/usr/bin/csh**) their **.cshrc** file will be modified rather than the **.profile** file.

Flags

Item	Description
-d	Used to remove the **NLSPATH** environment variable. This option will remove **NLSPATH** from either **/etc/environment** or the user's **.profile**. If **NLSPATH** was not currently in the file being modified, a warning message will be displayed.
-m *MsgTransLst*	Used to make modifications to the **NSLPATH** environment variable. *MsgTransLst* is a colon-separated list of message translations (locale names) that indicates the message translation hierarchy required for the system or user. If the first language in the list is different from the **Language** parameter and **Language** parameter has system supplied translation, then the **LC_MESSAGES** environment variable will be set to that first value. If the first language-territory in the list is the same as the language being set, the **LC_MESSAGES** environment variable will be removed. All entries in the list become hard coded directories in the **NLSPATH** environment.
-M	Used to reset the **LC_MESSAGES** environment variable and set the **NLSPATH** environment variable to the default translation hierarchy, which is: **/usr/lib/nls/msg/%L/%N:** **/usr/lib/nls/msg/%L/%N.cat:**
-u *UID* or *UName*	Used to make modification to an individual user. The user can be specified by either user id number or user login name. If the effective id of **chlang** is root, the **-u** parameter must be used to change the language environment for any specific user ID, including root itself (no **-u** parameter in this case will update the **/etc/environment** file rather than root's **.profile**). If the effective id is not root, the **-u** parameter is not needed. If it is specified, it must be equal to the effective id of the invoker.
Language	This is the language-territory (locale name) that will become the locale setting for the LANG environment variable.

Exit Status

Item	Description
0	Indicates successful completion.
>0	Indicates an error occurred.

Examples

1. Assume the preferred locale is Norwegian, and the language translations in order of preference are Norwegian, Swedish, and English. The command to achieve this for user *amcleod* is as follows:

   ```
   chlang -u amcleod -m no_NO:sv_SE:en_US no_NO
   ```

   The following settings would be made in the **.profile** for user *amcleod*. Because the first language in the message translation list is Norwegian, as is the **Language** parameter, **LC_MESSAGES** would not be set by **chlang**. If **LC_MESSAGES** had been set, it would be removed:

   ```
   LANG=no_NO

   NLSPATH=/usr/lib/nls/msg/%L/%N:
           /usr/lib/nls/msg/no_NO/%N:
           /usr/lib/nls/msg/sv_SE/%N:
           /usr/lib/nls/msg/en_US/%N:
           /usr/lib/nls/msg/%L/%N.cat:
   ```

```
        /usr/lib/nls/msg/no_NO/%N.cat:
        /usr/lib/nls/msg/sv_SE/%N.cat:
        /usr/lib/nls/msg/en_US/%N.cat
```

2. Assume the preferred locale is French, and the language translations in order of preference are French Canadian and English. To achieve this for a non-root user enter:

```
chlang -m fr_CA:en_US fr_FR
```

   The following settings would be made in the **.profile** file for the user invoking **chlang**. Because the first language in the message translation list is different from the cultural convention (locale), **LC_MESSAGES** is set by **chlang**.

```
LANG=fr_FR

LC_MESSAGES=fr_CA

NLSPATH=/usr/lib/nls/msg/%L/%N:
        /usr/lib/nls/msg/fr_CA/%N:
        /usr/lib/nls/msg/en_US/%N:
        /usr/lib/nls/msg/%L/%N.cat:
        /usr/lib/nls/msg/fr_CA/%N.cat:
        /usr/lib/nls/msg/en_US/%N.cat
```

3. Assume that a system administrator (root authority) in Spain is configuring a system from another country, and needs to change the default language environment so the machine operates properly in its new location. To change the default in the **/etc/environment** file, enter:

```
chlang -m es_ES es_ES
```

   The following settings would be made in the **/etc/environment** file.

```
LANG=es_ES

NLSPATH=/usr/lib/nls/msg/%L/%N:
        /usr/lib/nls/msg/es_ES/%N:
        /usr/lib/nls/msg/%L/%N.cat:
        /usr/lib/nls/msg/es_ES/%N.cat
```

Files

Item	Description
/usr/bin/chlang	Change language command
/etc/environment	Specifies basic environment for all processes
$HOME/.profile	Specifies environment for specific user needs

chlicense Command

Purpose

Changes the number of fixed licenses and the status of the floating licensing of the system.

Syntax

chlicense [[**-D** | **-I**] **-u** *FixedUsers*]] [[**-v**] **-f** *FloatingStatus*]

Note: At least one flag must be specified with the **chlicense** command.

Description

There are two types of user licensing: fixed and floating. Fixed licensing is always enabled and the number of licenses can be changed using **-u** flag of the **chlicense** command. Floating licensing is enabled or disabled using the **-f** flag.

Flags

Note: At least one flag must be specified with the **chlicense** command.

Item	Description
-D	The **-D** flag causes the new fixed-license value to be updated in the **login.cfg** file only. This is the option if the **-I** flag is not issued. You must restart the system before the new number takes effect.
-f *FloatingStatus*	Changes the status of the floating licensing of the system. The status must be either **on** or **off**. The status of **on** enables the floating licensing and **off** disables the floating licensing. The **-f** flag is optional.
-I	The **-I** flag causes the chlicense command to modify the current value of the fixed-license counting semaphore, in addition to modifying the value in the **login.cfg** file.
-u *FixedUser*	Changes the number of fixed licenses on a system. The value of *FixedUser* must be a number greater than 0. The **-u** flag is optional.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To enable the floating licensing for the system, enter:

   ```
   chlicense -f on
   ```

2. To disable the floating licensing for the system, enter:

   ```
   chlicense -f off
   ```

3. To change the number of fixed licenses to 125 and to enable floating licensing on the system, enter:

   ```
   chlicense -u 125 -f on
   ```

4. To immediatly increase the number of fixed licenses to 5, enter:

   ```
   chlicense -I -u 5
   ```

chlpclacl Command

Purpose

Changes the access controls for the least-privilege (LP) resource class (`IBM.LPCommands`).

Syntax

To add one or more accesses to the IBM.LPCommands Class ACL or to overwrite the IBM.LPCommands Class ACL with one or more accesses:

chlpclacl [-a | -n *host1*[,*host2*,...]] [-o] [-h] [-TV] *ID_1 perm1* [*ID_2 perm2*] ...

To add one or more accesses to the IBM.LPCommands Class ACL or to overwrite the IBM.LPCommands Class ACL with one or more accesses all using the same permissions:

chlpclacl [-a | -n *host1*[,*host2*,...]] -l [-o] [-h] [-TV] *ID_1* [*ID_2*...] *perm*

To delete one or more accesses from the IBM.LPCommands Class ACL:

chlpclacl [-a | -n *host1*[,*host2*,...]] -d [-h] [-TV] *ID_1* [*ID_2*...]

To add accesses to (or remove accesses from) the IBM.LPCommands Class ACL or to overwrite the IBM.LPCommands Class ACL, with the accesses specified in a file:

chlpclacl [-a | -n *host1*[,*host2*,...]] [-o | -d] -f *file_name* [-h] [-TV]

To set the IBM.LPCommands Class ACL to deny all accesses:

chlpclacl [-a | -n *host1*[,*host2*,...]] -x [-h] [-TV]

Description

The chlpclacl command changes the access control list (ACL) that is associated with the least-privilege (LP) resource class (IBM.LPCommands). This command allows an access to be added to or removed from the IBM.LPCommands Class ACL. This ACL controls access to such class operations as creating LP resources and deleting LP resources. One Class ACL exists on each node for the IBM.LPCommands class.

To add accesses to the IBM.LPCommands Class ACL, specify the ID and the permission the ID is to have. More than one ID and permission pair can be specified. If you want to add multiple IDs and they will all have the same permission, use the -l flag to indicate that the format of the command is a list of IDs followed by a single permission that applies to all of the IDs. If you use the -o flag, the IDs and permissions specified with the command will overwrite the existing accesses. The previously-defined accesses in the Class ACL are deleted.

To delete accesses from the IBM.LPCommands Class ACL, use the -d flag and specify the IDs to be deleted.

Use the -f flag to indicate that the accesses are specified in a file. Each line of the file will be an ID and permission for that ID. If the -d flag is used with the -f flag, only the ID is needed on each line. Everything after the first space is ignored.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the -a flag. If you want this command to run on a subset of nodes in a domain, use the -n flag. Otherwise, this command runs on the local node.

Flags

-a

Changes IBM.LPCommands Class ACLs on all nodes in the domain. The CT_MANAGEMENT_SCOPE environment variable's setting determines the cluster scope. If CT_MANAGEMENT_SCOPE is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The chlpclacl command runs once for the first valid scope that the LP resource manager finds. For example, suppose a management domain and a peer domain exist and the CT_MANAGEMENT_SCOPE environment variable is not set. In this case, chlpclacl –a runs in the management domain. To run chlpclacl –a in the peer domain, you must set CT_MANAGEMENT_SCOPE to 2.

-d
Removes the ACL entry for the specified ID from the IBM.LPCommands Class ACL.

-f *file_name*
Indicates that the accesses are specified in *file_name*. Each line of this file consists of an ID and the permission for that ID. If the -d flag is used with the -f flag, only the ID is needed on each line. Everything after the first space is ignored.

-l
Indicates that there is a list of IDs followed by a single permission that is used for all of the IDs.

-n *host1*[,*host2*,...]
Specifies the nodes in the domain on which the IBM.LPCommands Class ACL should be changed. By default, the IBM.LPCommands Class ACL is changed on the local node. This flag is valid only in a management domain or a peer domain. If CT_MANAGEMENT_SCOPE is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found.

-o
Indicates that the specified accesses overwrite any existing ACL entries for the IBM.LPCommands Class ACL. Any ACL entries in the IBM.LPCommands Class ACL are deleted.

-x
Sets the IBM.LPCommands Class ACL to deny all accesses to the IBM.LPCommands class attributes and class operations. Any ACL entries in the IBM.LPCommands Class ACL are deleted.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error.

-V
Writes the command's verbose messages to standard output.

Parameters

ID
Specifies the network identity of the user. If the same *ID* is listed more than once, the last permission specified is used. For a description of how to specify the network identity, see the User identities section of the lpacl information file.

perm
Specifies the permission allowed for *ID*. *perm* is specified as a string of one or more characters, where each character represents a particular permission. The valid values for *perm* are:

r
Read permission (consists of the q, l, e, and v permissions)

w
Write permission (consists of the d, c, s, and o permissions)

a
Administrator permission

x
Execute permission

q
Query permission

l
Enumerate permission

e
Event permission

v
Validate permission

d
Define and undefine permission

c
Refresh permission

s
Set permission

o
Online, offline, and reset permission

0
No permission

See the `User permissions` section of the `lpacl` information file for descriptions of these permissions.

Security

To run the `chlpclacl` command, you need read and administrator permission in the Class ACL of the `IBM.LPCommands` resource class. Permissions are specified in the LP ACLs on the contacted system. See the `lpacl` information file for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
The resource was not found.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When `CT_CONTACT` is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If `CT_CONTACT` is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the **CT_IP_AUTHENT** environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the **CT_CONTACT** environment variable is set. **CT_IP_AUTHENT** only has meaning if **CT_CONTACT** is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used, unless the `-a` flag or the `-n` flag is specified.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. When the `-V` flag is specified, this command's verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To give user `joe` on nodeA write permission to the `IBM.LPCommands` class so that he can create LP resources on nodeA, run one of these commands on nodeA:

```
chlpclacl joe@NODEID w

chlpclacl joe@LOCALHOST w
```

2. nodeA and nodeB are in a peer domain. To give user `joe` on nodeB write permission to the `IBM.LPCommands` class so that he can create LP resources on nodeB, run this command on nodeA:

```
chlpclacl -n nodeB joe@LOCALHOST  w
```

   In this example, specifying `joe@NODEID` instead of `joe@LOCALHOST` gives `joe` on nodeA write permission to the `IBM.LPCommands` class on nodeB.

3. To give user `joe` on nodeA write permission to the `IBM.LPCommands` class and `bill` on nodeA administrator permission and write permission to the `IBM.LPCommands` class on nodeA, run this command on nodeA:

```
chlpclacl joe@LOCALHOST w bill@LOCALHOST wa
```

4. To give user `joe` on nodeA administrator permission to the `IBM.LPCommands` class on nodeA, overwriting the current `IBM.LPCommands` Class ACL so that this is the only access allowed, run this command on nodeA:

```
chlpclacl -o joe@LOCALHOST a
```

5. To give users joe, bill, and jane on nodeA read and write permissions to the IBM.LPCommands class on nodeA, run this command on nodeA:

```
chlpclacl -l joe@LOCALHOST  bill@LOCALHOST  jane@LOCALHOST  rw
```

6. To delete access for joe on nodeA from the IBM.LPCommands class on nodeA, run this command on nodeA:

```
chlpclacl -d  joe@LOCALHOST
```

7. To add a list of accesses that are in a file named /mysecure/aclfile on nodeA to the IBM.LPCommands class on nodeA, run this command on nodeA:

```
chlpclacl -f /mysecure/aclfile
```

   The contents of /mysecure/aclfile on nodeA could be:

```
joe@LOCALHOST              w
bill@LOCALHOST             wa
jane@LOCALHOST             rw
```

8. To deny all accesses to the IBM.LPCommands class on nodeA, run this command on nodeA:

```
chlpclacl -x
```

Location

/opt/rsct/bin/chlpclacl
Contains the chlpclacl command

chlpcmd Command

Purpose

Changes the attribute values of a least-privilege (LP) resource.

Syntax

To change the attribute values of an LP resource:

- On the local node:

  chlpcmd [-l 0 | 1] [-c 0 | 1 | 2 | 3] [-h] [-TV] *resource_name attr1=value1* [*attr2=value2*...]

  chlpcmd -r [-h] [-TV] *resource_name*

- On all nodes in a domain:

  chlpcmd -a [-l 0 | 1] [-c 0 | 1 | 2 | 3] [-h] [-TV] *resource_name attr1=value1* [*attr2=value2*...]

  chlpcmd -a -r [-h] [-TV] *resource_name*

- On a subset of nodes in a domain:

  chlpcmd -n *host1* [,*host2*,...] [-l 0 | 1] [-c 0 | 1 | 2 | 3] [-h] [-TV] *resource_name attr1=value1* [*attr2=value2*...]

  chlpcmd -n *host1* [,*host2*,...] -r [-h] [-TV] *resource_name*

Description

Use the chlpcmd command to change any of the read/write attribute values of an LP resource. An *LP resource* is a root command or script to which users are granted access based on permissions in the LP access control lists (ACLs). Use the -r flag to recalculate and assign the CheckSum attribute. Use the -c flag to change the ControlFlags attribute. Use the -l flag to change the Lock attribute. Use

attr=*value* parameters to modify these attributes: Name, CommandPath, RunCmdName, FilterScript, FilterArg, and Description.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the -a flag. If you want this command to run on a subset of nodes in a domain, use the -n flag. Otherwise, this command runs on the local node.

Flags

-a

Changes attribute values for *resource_name* on all nodes in the domain. The CT_MANAGEMENT_SCOPE environment variable's setting determines the cluster scope. If CT_MANAGEMENT_SCOPE is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The chlpcmd command runs once for the first valid scope that the LP resource manager finds. For example, suppose a management domain and a peer domain exist and the CT_MANAGEMENT_SCOPE environment variable is not set. In this case, chlpcmd –a runs in the management domain. To run chlpcmd –a in the peer domain, you must set CT_MANAGEMENT_SCOPE to 2.

-n *host1*[,*host2*,...]

Specifies one or more nodes in the domain on which the LP resource is to be changed. By default, the LP resource is changed on the local node. This flag is valid only in a management domain or a peer domain. If the CT_MANAGEMENT_SCOPE environment variable is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The chlpcmd command runs once for the first valid scope that the LP resource manager finds.

–r

Recalculates and assigns the CheckSum attribute value for this LP resource. Use the -r flag when:

- You have modified the command or script that this LP resource represents.
- You want to change the CheckSum value from 0 to the correct value after the command or script becomes available on the system.

-l 0 | **1**

Locks or unlocks the resource. You can use this flag to protect the resource from being deleted by accident. The default value is 0, which means no lock is set. To lock the resource, use chlpcmd -l 1.

-c 0 | **1** | **2** | **3**

Sets the ControlFlags attribute, which is used to specify the control features for an LP command. If ControlFlags is not specified, it is set to 1 by default. Use this flag to specify one of these values:

0

Does not validate the CheckSum value.

1

Does not validate the CheckSum value. This is the default.

2

Validates the CheckSum value.

3

Validates the CheckSum value.

When an attempt is made to run the LP resource using the `runlpcmd` command, the value of the `ControlFlags` attribute determines which checks are performed before running the command represented by the resource.

In this release of RSCT, the `ControlFlags` attribute value specifies whether the `CheckSum` value is to be validated.

In previous releases of RSCT, the `ControlFlags` attribute value also specified whether the presence of certain characters in the input arguments to `runlpcmd` were to be disallowed. Checking for these characters is no longer necessary.

To maintain compatibility with LP resources that were defined in previous releases of RSCT, the `ControlFlags` attribute values, with respect to validating the `CheckSum` value, have remained the same. Consequently, values `0` and `1` indicate that the `CheckSum` value is not to be validated, and values 2 and 3 indicate that the `CheckSum` value is to be validated.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error.

-V
Writes the command's verbose messages to standard output.

Parameters

resource_name
Specifies the name of the LP resource to change.

***attr1=value1* [*attr2=value2*...]**
Specifies one or more read/write attributes and their new values.

Security

To run the `chlpcmd` command, you need:

- read permission in the Class ACL of the `IBM.LPCommands` resource class.
- write permission in the Resource ACL.

  As an alternative, the Resource ACL can direct the use of the Resource Shared ACL if this permission exists in the Resource Shared ACL.

Permissions are specified in the LP ACLs on the contacted system. See the `lpacl` file for general information about LP ACLs and the *RSCT Administration Guide* for information about modifying them.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
The resource was not found.

Environment Variables

CT_CONTACT
Determines the system that is used for the session with the RMC daemon. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the LP resources that are processed.

CT_IP_AUTHENT
When the **CT_IP_AUTHENT** environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the **CT_CONTACT** environment variable is set. **CT_IP_AUTHENT** only has meaning if **CT_CONTACT** is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to process the LP resources. The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If CT_MANAGEMENT_SCOPE is not set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. When the -V flag is specified, this command's verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To change the Lock attribute of LP resource lpcommand1 before deleting a resource on a local node, enter:

```
chlpcmd -l 0 lpcommand1
```

2. Suppose nodeA is in a management domain and CT_MANAGEMENT_SCOPE is set to 3. To recalculate the CheckSum attribute value of LP resource lpcommand2 on nodeA, enter:

```
chlpcmd -r -n nodeA lpcommand2
```

Location

/opt/rsct/bin/chlpcmd
Contains the `chlpcmd` command

chlpracl Command

Purpose

Changes the access controls for a least-privilege (LP) resource.

Syntax

To add one or more accesses to a Resource ACL or to overwrite a Resource ACL with one or more accesses:

`chlpracl` [`-a` | `-n` *host1*[,*host2*,...]] [`-o`] [`-r`] [`-h`] [`-TV`] *resource ID_1 perm1* [*ID_2 perm2*] ...

To add one or more accesses to a Resource ACL or to overwrite an Resource ACL with one or more accesses all using the same permissions:

`chlpracl` [`-a` | `-n` *host1*[,*host2*,...]] `-l` [`-o`] [`-r`] [`-h`] [`-TV`] *resource ID_1* [*ID_2*...] *perm*

To delete one or more accesses from a Resource ACL:

`chlpracl` [`-a` | `-n` *host1*[,*host2*,...]] `-d` [`-r`] [`-h`] [`-TV`] *resource ID_1* [*ID_2*...]

To add accesses to (or remove accesses from) a Resource ACL or to overwrite a Resource ACL, with the accesses specified in a file:

`chlpracl` [`-a` | `-n` *host1*[,*host2*,...]] [`-o` | `-d`] `-f` *file_name* [`-r`] [`-h`] [`-TV`] *resource*

To set a Resource ACL so that no permissions are allowed, or to use the Resource Shared ACL:

`chlpracl` [`-a` | `-n` *host1*[,*host2*,...]] { `-b` | `-x` } [`-r`] [`-h`] [`-TV`] *resource*

To set all of the Resource ACLs so that no permissions are allowed, or to use the Resource Shared ACL:

`chlpracl` [`-a` | `-n` *host1*[,*host2*,...]] { `-B` | `-X` } [`-h`] [`-TV`]

Description

The `chlpracl` command changes the access control list (ACL) that is associated with a least-privilege (LP) resource. This command allows an access to be added to or removed from the Resource ACL. This ACL controls access to such resource operations as listing attribute values and running LP commands. One Resource ACL exists for each LP resource.

For controlling access to the LP resource, three different types of Resource ACLs exist:

1. Resource ACL
2. Resource Initial ACL
3. Resource Shared ACL

The `chlpracl` command allows the Resource ACL to indicate that the Resource Shared ACL should be used in its stead to control access. For descriptions of these ACLs, see the `lpacl` information file.

To add an access to the Resource ACL, specify the name of the LP resource, the ID, and the permission the ID is to have. More than one ID and permission pair can be specified. If you want to add multiple IDs and they will all have the same permission, use the `-l` flag to indicate that the format of the command is a list of IDs followed by a single permission that applies to all of the IDs. If you use the `-o` flag, the IDs and permissions specified with the command will overwrite the existing accesses. The previously-defined accesses in the ACL are deleted.

To delete accesses from the Resource ACL, use the `-d` flag and specify the name of the LP resource and the IDs to be deleted.

Use the `-f` flag to indicate that the accesses are specified in a file. Each line of the file will be an ID and permission for that ID. If the `-d` flag is used with the `-f` flag, only the ID is needed on each line. Everything after the first space is ignored.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the `-a` flag. If you want this command to run on a subset of nodes in a domain, use the `-n` flag. Otherwise, this command runs on the local node.

Flags

-a
Changes the Resource ACLs for *resource* on all nodes in the domain. The `CT_MANAGEMENT_SCOPE` environment variable's setting determines the cluster scope. If `CT_MANAGEMENT_SCOPE` is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The `chlpracl` command runs once for the first valid scope that the LP resource manager finds. For example, suppose a management domain and a peer domain exist and the `CT_MANAGEMENT_SCOPE` environment variable is not set. In this case, `chlpracl –a` runs in the management domain. To run `chlpracl –a` in the peer domain, you must set `CT_MANAGEMENT_SCOPE` to 2.

-b
Bypasses the ACL for the specified LP resource. The Resource Shared ACL is used for access control for this LP resource. Any ACL entries in the Resource ACL are deleted.

-B
Bypasses the ACLs for all LP resources. The Resource Shared ACL is used for access control for all LP resources. Any ACL entries in the Resource ACLs are deleted. One Resource Shared ACL exists for each `IBM.LPCommands` class (or node).

-d
Removes the ACL entry for the specified ID from the specified Resource ACL.

-f *file_name*
Indicates that the accesses are specified in *file_name*. Each line of this file consists of an ID and the permission for that ID. If the `-d` flag is used with the `-f` flag, only the ID is needed on each line. Everything after the first space is ignored.

-l
Indicates that there is a list of IDs followed by a single permission that is used for all of the IDs.

-n *host1*[,*host2*,...]
Specifies the nodes in the domain on which the Resource ACL should be changed. By default, the Resource ACL is changed on the local node. This flag is valid only in a management domain or a peer domain. If `CT_MANAGEMENT_SCOPE` is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found.

-o
Indicates that the specified ACL accesses overwrite any existing ACL entries for the specified Resource ACL. Any ACL entries in the Resource ACL are deleted.

-r
Indicates that *resource* is a "typical" RSCT resource handle. The resource handle must be enclosed in quotation marks. The Resource ACL of the resource handle is modified.

-x
Sets the Resource ACL for the specified LP resource to deny all accesses to the LP resource. Any ACL entries in the Resource ACL are deleted.

-X
Sets the Resource ACL of all LP resources to deny all accesses to the LP resource. Any ACL entries in the Resource ACLs are deleted.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error.

-V
Writes the command's verbose messages to standard output.

Parameters

resource
Specifies the name of the LP resource for which the Resource ACL is changed.

ID
Specifies the network identity of the user. If the same *ID* is listed more than once, the last permission specified is used. For a description of how to specify the network identity, see the `lpacl` information file.

perm
Specifies the permission allowed for *ID*. *perm* is specified as a string of one or more characters, where each character represents a particular permission. The valid values for *perm* are:

r
Read permission (consists of the q, l, e, and v permissions)

w
Write permission (consists of the d, c, s, and o permissions)

a
Administrator permission

x
Execute permission

q
Query permission

l
Enumerate permission

e
Event permission

v
Validate permission

d
Define and undefine permission

c
Refresh permission

s
Set permission

o
Online, offline, and reset permission

0
No permission

See the `lpacl` information file for a description of each permission and how it applies.

Security

To run the `chlpracl` command, you need:

- read permission in the Class ACL of the `IBM.LPCommands` resource class.
- read and administrator permission in the Resource ACL.

  As an alternative, the Resource ACL can direct the use of the Resource Shared ACL if these permissions exist in the Resource Shared ACL.

Permissions are specified in the LP ACLs on the contacted system. See the `lpacl` information file for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
The resource was not found.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When `CT_CONTACT` is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If `CT_CONTACT` is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the **CT_IP_AUTHENT** environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the **CT_CONTACT** environment variable is set. **CT_IP_AUTHENT** only has meaning if **CT_CONTACT** is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used, unless the `-a` flag or the `-n` flag is specified.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. When the `-V` flag is specified, this command's verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To give user `joe` on nodeA the ability to run the LP command `lpcommand1` on nodeA, run one of these commands on nodeA:

   ```
   chlpracl lpcommand1 joe@NODEID  x

   chlpracl lpcommand1 joe@LOCALHOST  x
   ```

2. nodeA and nodeB are in a peer domain. To give user `joe` on nodeB the ability to run the LP command `lpcommand1` on nodeB, run this command on nodeA:

   ```
   chlpracl -n nodeB lpcommand1 joe@LOCALHOST  x
   ```

   In this example, specifying `joe@NODEID` instead of `joe@LOCALHOST` gives `joe` on nodeA the ability to run the LP command `lpcommand1` on nodeB.

3. To give user `joe` on nodeA execute permission to the LP command `lpcommand1` and `bill` on nodeA administrator permission and write permission to the same resource on nodeA, run this command on nodeA:

   ```
   chlpracl lpcommand1 joe@LOCALHOST  x  bill@LOCALHOST  wa
   ```

4. To give user `joe` on nodeA administrator permission to the LP command `lpcommand1` on nodeA, overwriting the current ACLs for `lpcommand1` so that this is the only access allowed, run this command on nodeA:

   ```
   chlpracl -o lpcommand1 joe@LOCALHOST x
   ```

5. To give users `joe`, `bill`, and `jane` on nodeA the ability to run the LP command `lpcommand1` on nodeA, run this command on nodeA:

   ```
   chlpracl lpcommand1 -l joe@LOCALHOST  bill@LOCALHOST  jane@LOCALHOST  x
   ```

6. To delete access for `joe` on nodeA from the ACLs for the LP command `lpcommand1` on nodeA, run this command on nodeA:

   ```
   chlpracl -d lpcommand1 joe@LOCALHOST
   ```

7. To add a list of accesses that are in a file named `/mysecure/aclfile` on nodeA to the LP command `lpcommand1` on nodeA, run this command on nodeA:

   ```
   chlpracl -f /mysecure/aclfile lpcommand1
   ```

The contents of /mysecure/aclfile on nodeA could be:

```
joe@LOCALHOST              x
bill@LOCALHOST                   ax
jane@LOCALHOST             wx
```

8. To bypass the Resource ACL for the LP command lpcommand1 on nodeA, and use the Resource Shared ACL to control access to it, run this command on nodeA:

```
chlpracl -b lpcommand1
```

9. To bypass the Resource ACLs for all of the LP resources on nodeA, and use the Resource Shared ACL to control accesses, run this command on nodeA:

```
chlpracl -B
```

10. To deny all accesses to the LP command lpcommand1 on nodeA, run this command on nodeA:

```
chlpracl -x lpcommand1
```

Location

/opt/rsct/bin/chlpracl
Contains the chlpracl command

chlpriacl Command

Purpose

Changes the access controls for the least-privilege (LP) Resource Initial ACL.

Syntax

To add one or more accesses to the Resource Initial ACL or to overwrite the Resource Initial ACL with one or more accesses:

chlpriacl [-a | -n *host1*[,*host2*,...]] [-o] [-h] [-TV] *ID_1 perm1* [*ID_2 perm2*] ...

To add one or more accesses to the Resource Initial ACL or to overwrite the Resource Initial ACL with one or more accesses all using the same permissions:

chlpriacl [-a | -n *host1*[,*host2*,...]] -l [-o] [-h] [-TV] *ID_1* [*ID_2*...] *perm*

To delete one or more accesses from the Resource Initial ACL:

chlpriacl [-a | -n *host1*[,*host2*,...]] -d [-h] [-TV] *ID_1* [*ID_2*...]

To add accesses to (or remove accesses from) the Resource Initial ACL or to overwrite the Resource Initial ACL, with the accesses specified in a file:

chlpriacl [-a | -n *host1*[,*host2*,...]] [-o | -d] -f *file_name* [-h] [-TV]

To set the Resource Initial ACL to use the Resource Shared ACL or so that no permissions are allowed:

chlpriacl [-a | -n *host1*[,*host2*,...]] { -b | -x } [-h] [-TV]

Description

The chlpriacl command changes the access control list (ACL) that is associated with the least-privilege (LP) Resource Initial ACL. This command allows a user to be added to or removed from the Resource Initial ACL. This ACL is used to initialize a Resource ACL when the LP resource is created. The Resource Initial ACL can consist of ACL entries that define permissions to the LP resource or it can indicate that the

Resource Shared ACL should be used to control access instead of the Resource ACL. One Resource Initial ACL exists on each node for the IBM.LPCommands class.

To add accesses to the Resource Initial ACL, specify the ID and the permission the ID is to have. More than one ID and permission pair can be specified. If you want to add multiple IDs and they will all have the same permission, use the -l flag to indicate that the format of the command is a list of IDs followed by a single permission that applies to all of the IDs. If you use the -o flag, the IDs and permissions specified with the command will overwrite the existing accesses. The previously-defined accesses in the ACL are deleted.

To delete accesses from the Resource Initial ACL, use the -d flag and specify the IDs to be deleted.

Use the -f flag to indicate that the accesses are specified in a file. Each line of the file will be an ID and permission for that ID. If the -d flag is used with the -f flag, only the ID is needed on each line. Everything after the first space is ignored.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the -a flag. If you want this command to run on a subset of nodes in a domain, use the -n flag. Otherwise, this command runs on the local node.

Flags

-a
Changes the Resource Initial ACLs on all nodes in the domain. The CT_MANAGEMENT_SCOPE environment variable's setting determines the cluster scope. If CT_MANAGEMENT_SCOPE is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The chlpriacl command runs once for the first valid scope that the LP resource manager finds. For example, suppose a management domain and a peer domain exist and the CT_MANAGEMENT_SCOPE environment variable is not set. In this case, chlpriacl –a runs in the management domain. To run chlpriacl –a in the peer domain, you must set CT_MANAGEMENT_SCOPE to 2.

-b
Sets the Resource Initial ACL to indicate that the Resource ACL is bypassed and that the Resource Shared ACL is used for access control for the LP resource. Any ACL entries in the Resource Initial ACL are deleted. When a new LP resource is created, the Resource Shared ACL is used for it.

-d
Removes the ACL entry for the specified ID from the Resource Initial ACL.

-f *file_name*
Indicates that the accesses are specified in *file_name*. Each line of this file consists of an ID and the permission for that ID. If the -d flag is used with the -f flag, only the ID is needed on each line. Everything after the first space is ignored.

-l
Indicates that there is a list of IDs followed by a single permission that is used for all of the IDs.

-n *host1*[,*host2*,...]
Specifies the node in the domain on which the Resource Initial ACL should be changed. By default, the Resource Initial ACL is changed on the local node. This flag is valid only in a management domain or a peer domain. If CT_MANAGEMENT_SCOPE is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found.

-o
Indicates that the specified ACL entries overwrite any existing ACL entries for the Resource Initial ACL. Any ACL entries in the Resource Initial ACL are deleted.

-x
Sets the Resource Initial ACL to deny all accesses to the LP resource. Any ACL entries in the Resource Initial ACL are deleted. When a new LP resource is created, all accesses will be denied to it.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error.

-V
Writes the command's verbose messages to standard output.

Parameters

ID
Specifies the network identity of the user. If the same *ID* is listed more than once, the last permission specified is used. For a description of how to specify the network identity, see the `lpacl` information file.

perm
Specifies the permission allowed for *ID*. *perm* is specified as a string of one or more characters, where each character represents a particular permission. The valid values for *perm* are:

r
Read permission (consists of the q, l, e, and v permissions)

w
Write permission (consists of the d, c, s, and o permissions)

a
Administrator permission

x
Execute permission

q
Query permission

l
Enumerate permission

e
Event permission

v
Validate permission

d
Define and undefine permission

c
Refresh permission

s
Set permission

o
Online, offline, and reset permission

0
No permission

See the `lpacl` information file for a description of each permission and how it applies.

Security

To run the `chlpriacl` command, you need read and administrator permission in the Class ACL of the `IBM.LPCommands` resource class. Permissions are specified in the LP ACLs on the contacted system. See

the `lpacl` information file for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
The resource was not found.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When `CT_CONTACT` is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If `CT_CONTACT` is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the **CT_IP_AUTHENT** environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the **CT_CONTACT** environment variable is set. **CT_IP_AUTHENT** only has meaning if **CT_CONTACT** is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used, unless the `-a` flag or the `-n` flag is specified.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. When the `-V` flag is specified, this command's verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To give user `joe` on nodeA execute permission in the Resource Initial ACL on nodeA, run one of these commands on nodeA:

```
chlpriacl joe@NODEID  x

chlpriacl joe@LOCALHOST  x
```

2. nodeA and nodeB are in a peer domain. To give user `joe` on nodeB execute permission to the Resource Initial ACL on nodeB, run this command on nodeA:

```
chlpriacl -n nodeB joe@LOCALHOST  x
```

   In this example, specifying `joe@NODEID` instead of `joe@LOCALHOST` gives `joe` on nodeA execute permission to the Resource Initial ACL on nodeB.

3. To give user `joe` on nodeA execute permission and `bill` on nodeA administrator permission and read permission to the Resource Initial ACL on nodeA, run this command on nodeA:

```
chlpriacl joe@LOCALHOST  x  bill@LOCALHOST  ra
```

4. To give user `joe` on nodeA execute permission to the Resource Initial ACL on nodeA, overwriting the current ACLs so that this is the only access allowed, run this command on nodeA:

```
chlpriacl -o joe@LOCALHOST x
```

5. To give users `joe`, `bill`, and `jane` on nodeA read permission and write permission to the Resource Initial ACL on nodeA on nodeA, run this command on nodeA:

```
chlpriacl -l joe@LOCALHOST  bill@LOCALHOST jane@LOCALHOST  rw
```

6. To delete access for `joe` on nodeA from the Resource Initial ACL on nodeA, run this command on nodeA:

```
chlpriacl -d  joe@LOCALHOST
```

7. To add a list of accesses that are in a file named `/mysecure/aclfile` on nodeA to the Resource Initial ACL on nodeA, run this command on nodeA:

```
chlpriacl -f /mysecure/aclfile
```

   The contents of `/mysecure/aclfile` on nodeA could be:

```
joe@LOCALHOST              x
bill@LOCALHOST             rw
jane@LOCALHOST             rwa
```

8. To set the Resource Initial ACL on nodeA so it indicates that the Resource Shared ACL on nodeA is used to control accesses for newly-created LP resources on nodeA, run this command on nodeA:

```
chlpriacl -b
```

9. To set the Resource Initial ACL on nodeA so that it denies all accesses for newly-created LP resources on nodeA, run this command on nodeA:

```
chlpriacl -x
```

Location

/opt/rsct/bin/chlpriacl
Contains the `chlpriacl` command

chlprsacl Command

Purpose

Changes the access controls for the least-privilege (LP) Resource Shared ACL.

Syntax

To add one or more accesses to the Resource Shared ACL or to overwrite the Resource Shared ACL with one or more accesses:

`chlprsacl` [`-a` | `-n` *host1*[,*host2*,...]] [`-o`] [`-h`] [`-TV`] *ID_1 perm1* [*ID_2 perm2*] ...

To add one or more accesses to the Resource Shared ACL or to overwrite the Resource Shared ACL with one or more accesses all using the same permissions:

`chlprsacl` [`-a` | `-n` *host1*[,*host2*,...]] `-l` [`-o`] [`-h`] [`-TV`] *ID_1* [*ID_2*...] *perm*

To delete one or more accesses from the Resource Shared ACL:

`chlprsacl` [`-a` | `-n` *host1*[,*host2*,...]] `-d` [`-h`] [`-TV`] *ID_1* [*ID_2*...]

To add accesses to (or remove accesses from) the Resource Shared ACL or to overwrite the Resource Shared ACL, with the accesses specified in a file:

`chlprsacl` [`-a` | `-n` *host1*[,*host2*,...]] [`-o` | `-d`] `-f` *file_name* [`-h`] [`-TV`]

To set the Resource Shared ACL so that no permissions are allowed:

`chlprsacl` [`-a` | `-n` *host1*[,*host2*,...]] `-x` [`-h`] [`-TV`]

Description

The `chlprsacl` command changes the access control list (ACL) that is associated with the Resource Shared ACL. This command allows a user to be added to or removed from the Resource Shared ACL. This ACL:

- is used to control accesses to LP resources when the Resource ACL indicates that it (the Resource Shared ACL) has control
- can control access to one or more LP resources
- can consist of ACL entries that define permissions to the LP resources

One Resource Shared ACL exists on each node for the `IBM.LPCommands` class.

The `chlpracl` command is used to indicate that the access to an LP resource is controlled by the Resource Shared ACL. The `chlpriacl` command is used to indicate that accesses to newly-created LP resources are controlled by the Resource Shared ACL, by modifying the Resource Initial ACL.

To add accesses to the Resource Shared ACL, specify the ID and the permission the ID is to have. More than one ID and permission pair can be specified. If you want to add multiple IDs and they will all have the same permission, use the `-l` flag to indicate that the format of the command is a list of IDs followed by a single permission that applies to all of the IDs. If you use the `-o` flag, the IDs and permissions

specified with the command will overwrite the existing accesses. The previously-defined accesses in the ACL are deleted.

To delete accesses from the Resource Shared ACL, use the `-d` flag and specify the IDs to be deleted.

Use the `-f` flag to indicate that the accesses are specified in a file. Each line of the file will be an ID and permission for that ID. If the `-d` flag is used with the `-f` flag, only the ID is needed on each line. Everything after the first space is ignored.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the `-a` flag. If you want this command to run on a subset of nodes in a domain, use the `-n` flag. Otherwise, this command runs on the local node.

Flags

-a
Changes the Resource Shared ACLs on all nodes in the domain. The `CT_MANAGEMENT_SCOPE` environment variable's setting determines the cluster scope. If `CT_MANAGEMENT_SCOPE` is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The `chlprsacl` command runs once for the first valid scope that the LP resource manager finds. For example, suppose a management domain and a peer domain exist and the `CT_MANAGEMENT_SCOPE` environment variable is not set. In this case, `chlprsacl –a` runs in the management domain. To run `chlprsacl –a` in the peer domain, you must set `CT_MANAGEMENT_SCOPE` to 2.

-d
Removes the ACL entry for the specified ID from the Resource Shared ACL.

-f *file_name*
Indicates that the accesses are specified in *file_name*. Each line of this file consists of an ID and the permission for that ID. If the `-d` flag is used with the `-f` flag, only the ID is needed on each line. Everything after the first space is ignored.

-l
Indicates that there is a list of IDs followed by a single permission that is used for all of the IDs.

-n *host1*[,*host2*,...]
Specifies the node in the domain on which the Resource Shared ACL should be changed. By default, the Resource Shared ACL is changed on the local node. This flag is valid only in a management domain or a peer domain. If `CT_MANAGEMENT_SCOPE` is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found.

-o
Indicates that the specified ACL entries overwrite any existing ACL entries for the Resource Shared ACL. Any ACL entries in the Resource Shared ACL are deleted.

-x
Sets the Resource Shared ACL to deny all accesses to the LP resources that use the Resource Shared ACL. Any ACL entries in the Resource Shared ACL are deleted.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error.

-V
Writes the command's verbose messages to standard output.

Parameters

ID
Specifies the network identity of the user. If the same *ID* is listed more than once, the last permission specified is used. For a description of how to specify the network identity, see the `lpacl` information file.

perm
Specifies the permission allowed for *ID*. *perm* is specified as a string of one or more characters, where each character represents a particular permission. The valid values for *perm* are:

r
Read permission (consists of the q, l, e, and v permissions)

w
Write permission (consists of the d, c, s, and o permissions)

a
Administrator permission

x
Execute permission

q
Query permission

l
Enumerate permission

e
Event permission

v
Validate permission

d
Define and undefine permission

c
Refresh permission

s
Set permission

o
Online, offline, and reset permission

0
No permission

See the `lpacl` information file for a description of each permission and how it applies.

Security

To run the `chlprsacl` command, you need read and administrator permission in the Class ACL of the `IBM.LPCommands` resource class. Permissions are specified in the LP ACLs on the contacted system. See the `lpacl` information file for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
The resource was not found.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the **CT_IP_AUTHENT** environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the **CT_CONTACT** environment variable is set. **CT_IP_AUTHENT** only has meaning if **CT_CONTACT** is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used, unless the -a flag or the -n flag is specified.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. When the -V flag is specified, this command's verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To give user joe on nodeA execute permission in the Resource Shared ACL on nodeA, run one of these commands on nodeA:

```
chlprsacl joe@NODEID  x

chlprsacl joe@LOCALHOST  x
```

2. nodeA and nodeB are in a peer domain. To give user joe on nodeB execute permission to the Resource Shared ACL on nodeB, run this command on nodeA:

```
chlprsacl -n nodeB joe@LOCALHOST  x
```

   In this example, specifying joe@NODEID instead of joe@LOCALHOST gives joe on nodeA execute permission to the Resource Shared ACL on nodeB.

3. To give user joe on nodeA execute permission and bill on nodeA administrator permission and read permission to the Resource Shared ACL on nodeA, run this command on nodeA:

```
chlprsacl joe@LOCALHOST  x  bill@LOCALHOST  ra
```

4. To give user joe on nodeA execute permission to the Resource Shared ACL on nodeA, overwriting the current ACLs so that this is the only access allowed, run this command on nodeA:

```
chlprsacl -o joe@LOCALHOST x
```

5. To give users joe, bill, and jane on nodeA read permission and write permission to the Resource Shared ACL on nodeA on nodeA, run this command on nodeA:

```
chlprsacl -l joe@LOCALHOST  bill@LOCALHOST jane@LOCALHOST  rw
```

6. To delete access for joe on nodeA from the Resource Shared ACL on nodeA, run this command on nodeA:

```
chlprsacl -d  joe@LOCALHOST
```

7. To add a list of accesses that are in a file named /mysecure/aclfile on nodeA to the Resource Shared ACL on nodeA, run this command on nodeA:

```
chlprsacl -f /mysecure/aclfile
```

   The contents of /mysecure/aclfile on nodeA could be:

```
joe@LOCALHOST              x
bill@LOCALHOST             rw
jane@LOCALHOST             rwa
```

8. To set the Resource Shared ACL on nodeA so that it denies all accesses for LP resources that use it on nodeA, run this command on nodeA:

```
chlprsacl -x
```

Location

/opt/rsct/bin/chlprsacl
Contains the chlprsacl command

chlv Command

Purpose

Changes only the characteristics of a logical volume.

Syntax

To Change the Characteristics of a Logical Volume

chlv [**-a** *position*] [**-b** *badblocks*] [**-d** *schedule*] [**-R** *PreferredRead*] [**-e** *Range*] [**-L** *label*] [**-o y | n**] [**-p** *permission*] [**-r** *relocate*] [**-s** *strict*] [**-t** *type*] [**-u** *upperbound*] [**-v** *verify*] [**-w** *mirrorwriteconsistency*] [**-x** *maximum*] [**-T O | F**] [**-U** *userid*] [**-G** *groupid*] [**-P** *modes*] [**-m** *copyN=mirrorpool*] [**-M** *copyn*] [**-O** { **y** | **n** }] [**-k y** | **n**] *logicalvolume ...*

To Change the Name of a Logical Volume

chlv -n *newlogicalvolume logicalvolume*

Note:

1. Changing the name of a log logical volume requires that you run the **chfs -a** *log=LVName* on each file system using that log.
2. If the logical volume has a file system mounted, the file system is automatically updated with the new logical volume name only if it is a JFS2 file system. For all other file system types, the user have to run **unmount** and **mount** options after the completion of the **chlv** command to update the filesystem with the new logical volume name.
3. Bad block relocation policy of a logical volume is not supported on a volume group that is created with 4 KB block physical volumes.

Description

The changes you make with the **-a**, **-e**, **-s**, and **-u** flags take effect only when new partitions are allocated or partitions are deleted. The other flags take effect immediately.

To change the name of a logical volume, use the **-n** flag and use the *newlogicalvolume* parameter to represent the new logical volume name. Do not use other flags with this syntax.

If the *volume group* which contains logical volume being changed is in big vg format, **U**, **G**, and **P** flags can be used to set the ownership, group and permissions respectively, of the special device files. Only root user will be able to set these values. If the *volume group* is exported, these values can be restored upon import if **R** flag is specified with **importvg** command.

Note:

1. Changes made to the logical volume are not reflected in the file systems. To change file system characteristics, use the **chfs** command.
2. To use this command, you must either have root user authority or be a member of the **system** group.
3. Mirror Write Consistency (MWC) and Bad Block Relocation (BBR) are not supported in a concurrent setup with multiple active nodes accessing a disk at the same time. These two options must be disabled in this type of concurrent setup.

You could also use the System Management Interface Tool (SMIT) **smit chlv** fast path to run this command.

See the section "Administering a PowerHA cluster" in the PowerHA SystemMirror Administration Guide, 7.1 or later, for a discussion of the behavior of this command in a PowerHA cluster.

Flags

Note:

1. When changing the characteristics of a striped logical volume, the **-d**, and **-e** flags are not valid.
2. When changing the characteristics of a logical volume in a snapshot volume group or in a volume group that has a snapshot volume group, the **-a**, **-b**, **-d**, **-e**, **-G**, **-k**, **-o**, **-P**, **-r**, **-t**, **-U**, **-u**, **-v**, **-w**, **-x**, and **-s** flags are not valid.
3. The Logical Volume must be closed to run the **chlv** command with the **-b**, **-o**, **-p**, **-v**, **-w**, **-T**, and **-M** flags.

Item	Description
-a *position*	Sets the intraphysical volume allocation policy (the position of the logical partitions on the physical volume). The *position* variable is represented by one of the following: **m** Allocates logical partitions in the outer middle section of each physical volume. This is the default position. **c** Allocates logical partitions in the center section of each physical volume. **e** Allocates logical partitions in the outer edge section of each physical volume. **ie** Allocates logical partitions in the inner edge section of each physical volume. **im** Allocates logical partitions in the inner middle section of each physical volume.
-b *badblocks*	Sets the bad-block relocation policy. The *badblocks* variable is represented by one of the following: **y** Causes bad-block relocation to occur. **n** Prevents bad block relocation from occurring.
-d *schedule*	Sets the scheduling policy when more than one logical partition is written. Must use parallel or sequential to mirror striped lv. The *schedule* variable is represented by one of the following: **p** Establishes a parallel scheduling policy. **ps** Parallel write with sequential read policy. All mirrors are written in parallel but always read from the first mirror if the first mirror is available. **pr** Parallel write round robin read. This policy is similar to the parallel policy except an attempt is made to spread the reads to the logical volume more evenly across all mirrors. **s** Establishes a sequential scheduling policy. When specifying policy of parallel or sequential strictness, set to **s** for super strictness. **Note:** The **-R** flag overwrites the read policy of the **-d** flag. If the preferred copy is not available then the reads follows the scheduling policy.

Item	Description
-R *PreferredRead*	Changes preferred read copy of the logical volume. Always reads from the preferred copy if the preferred copy is available. If the preferred copy is not available, the reads follow the scheduling policy of the logical volume. The *PreferredRead* variable can be set to a value ranging from 0 to 3. Setting the *PreferredRead* variable to 0 disables the preferred read copy of the logical volume.
-e *range*	Sets the interphysical volume allocation policy (the number of physical volumes to extend across, using the volumes that provide the best allocation). The value of the *range* variable is limited by the *upperbound* variable, set with the **-u** flag, and is represented by one of the following: **x** Allocates logical partitions across the maximum number of physical volumes. **m** Allocates logical partitions across the minimum number of physical volumes.
-G *groupid*	Specifies group ID for the logical volume special file.
-k y|n	Changes the data encryption option of the logical volume. As a best practice, you must use the **hdcryptmgr** command to change the encryption option of the logical volume. You can specify the following values for this flag: **y** The data encryption option of the logical volume is enabled. The primary key of the logical volume must be initialized to access the logical volume. Use the **hdcryptmgr plain2crypt** command to initialize the primary key of the logical volume and encrypt the data of the logical volume. **n** The data encryption option of the logical volume is disabled. Use the **hdcryptmgr crypt2plain** to decrypt the encrypted data of the logical volume. **Note:** • The data encryption option must be enabled at the volume group level before you can enable the data encryption option for a logical volume. • The **-k** flag cannot be used if the volume group is varied on in the concurrent mode. • The **-k** flag is not supported on `boot`, `dump`, `paging`, and `aio_cache` logical volume type.
-L *label*	Sets the logical volume label. The maximum size of the *label* variable is 127 characters.

Item	Description
-m *copyN=mirrorpool*	Enables mirror pools to the copies of a logical volume. *N* is the copy number (1, 2, or 3). A mirror pool is assigned to a copy by using the *copyN=mirrorpool* parameter. Specify a mirror pool for each copy of the logical volume. To specify more than one *copyN=mirrorpool* pair, provide multiple **-m** *copyN=mirrorpool* flags.
-M *copyn*	Disables mirror pools on the specified copy for this logical volume. The **copyn** variable is the copy number (1, 2, or 3). It specifies which copy to disable mirror pools on. To disable mirror pools on more than one copy, provide multiple **-M** *copyn* flags.
-n *newlogicalvolume*	Changes the name of the logical volume to that specified by the *newlogicalvolume* variable. Logical volume names must be unique system wide and can range from 1 to 15 characters.
-o y | n	Turns on/off serialization of overlapping IOs. If serialization is turned on then overlapping IOs are not allowed on a block range, and only a single IO in a block range is processed at any one time. Most applications like file systems and databases do serialization, and hence serialization should be turned `off`. The default for new logical volumes is `off`.
-O y | n	Changes the infinite retry option of the logical volume. **n** Disables the infinite retry option of the logical volume. The failing I/O on the logical volume is not retried. **y** Enables the infinite retry option of the logical volume. The failed I/O request is retried until it is successful. **Note:** 1. The infinite retry option is ignored for an LV when an active mirror write consistency is set. The infinite retry option must be enabled at the volume group level to work for a logical volume with active mirror write consistency turned on. 2. Infinite retry is not supported in a GLVM environment.
-p *permission*	Sets the access permission to read-write or read-only. The *permission* variable is represented by one of the following: **w** Sets the access permission to read-write. **r** Sets the access permission to read-only. **Note:** Mounting a JFS file system on a read-only logical volume is not supported.
-P *modes*	Specifies permissions (file modes) for the logical volume special file.

Item	Description
-r *relocate*	Sets the reorganization flag to allow or prevent the relocation of the logical volume during reorganization. The *relocate* variable is represented by one of the following: **y** Allows the logical volume to be relocated during reorganization. If the logical volume is striped, the **chlv** command will not let you change the relocation flag to **y**. **n** Prevents the logical volume from being relocated during reorganization.
-s *strict*	Determines the strict allocation policy. Copies of a logical partition can be allocated to share or not to share the same physical volume. The *strict* variable is represented by one of the following: **y** Sets a strict allocation policy, so copies of a logical partition cannot share the same physical volume. **n** Does not set a strict allocation policy, so copies of a logical partition can share the same physical volume. **s** Sets a super strict allocation policy, so that the partitions allocated for one mirror cannot share a physical volume with the partitions from another mirror **Note:** When changing a non super strict logical volume to a super strict logical volume, you must use the **-u** flag.
-t *type*	Sets the logical volume type. The maximum size is 31 characters. If the logical volume is striped, you cannot change *type* to boot.
-T O | F	The **-T O** option indicates that the logical volume control block does not occupy the first block of the logical volume. Therefore, the space is available for application data. Applications can identify this type of logical volume with the **IOCINFO** `ioctl` operation. The logical volume has a device subtype of `DS_LVZ`. A logical volume created without this option has a device subtype of `DS_LV`. **Tip:** The **-T** flag does not change any behavior of a logical volume beyond the reported subtype.
-U *userid*	Specifies user ID for the logical volume special file.
-u *upperbound*	Sets the maximum number of physical volumes for new allocation. The value of the *upperbound* variable should be between one and the total number of physical volumes. When using super strictness, the upper bound indicates the maximum number of physical volumes allowed for each mirror copy. When using striped logical volumes, the upper bound must be multiple of *stripewidth*.

Item	Description
-v *verify*	Sets the write-verify state for the logical volume. Causes all writes to the logical volume either to be verified with a follow-up read or not to be verified with a follow-up read. The *verify* variable is represented by one of the following: **y** Causes all writes to the logical volume to be verified with a follow-up read. **n** Causes all writes to the logical volume not to be verified with a follow-up read.
-w *mirrorwriteconsistency*	**y or a** Turns on *active* mirror write consistency which ensures data consistency among mirrored copies of a logical volume during normal I/O processing. **p** Turns on *passive* mirror write consistency which ensures data consistency among mirrored copies during volume group synchronization after a system interruption. **Note:** This function is available only on **big** type and **scalable** type of volume groups. **n** No mirror write consistency. See the **-f** flag of the **syncvg** command.
-x *maximum*	Sets the maximum number of logical partitions that can be allocated to the logical volume.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To change the interphysical volume allocation policy of logical volume `lv01`, enter:

   ```
   chlv   -e m
   lv01
   ```

   The interphysical volume allocation policy is set to minimum.
2. To change the type of logical volume `lv03`, enter:

   ```
   chlv   -t copy  lv03
   ```
3. To change the permission of logical volume `lv03` to read-only, enter:

   ```
   chlv   -p r  lv03
   ```

   Logical volume `lv03` now has read-only permission.

4. To change the type to `paging` and the maximum number of physical volumes for logical volume `lv03`, enter:

```
chlv  -t paging  -u 10 lv03
```

   The change in the type of logical volume takes effect immediately, but the change in the maximum number of physical volumes does not take effect until a new allocation is made.
5. To change the allocation characteristics of logical volume `lv07`, enter:

```
chlv  -a e  -e x  -r y  -s n  -u 5 lv07
```

6. To change the *PreferredRead* copy of logical volume `testlv` to 3, enter:

```
chlv -R 3 testlv
```

Files

Item	Description
/usr/sbin	Directory where **chlv** command is located.

chlvcopy Command

Purpose

Marks or unmarks mirror copy as a split mirror.

Syntax

chlvcopy [**-f**] { **-B** [**-s**] } | { **-b** [**-c** *copy*] [**-f**] [**-P**] [**-l** *newlvname*] [**-w**] } LV *name*

Description

Note:

1. To use this command, you must either have root user authority or be a member of the system group.
2. If persistence is used either by using the **-P** flag or by creating a child backup logical volume device by using the **-l** flag, it will cause the volume group to be usable only on AIX 4.3.2 or later. This is true even after removal of split mirror copy designation of the parent logical volume and the child backup logical volumes.
3. For **chlvcopy** to be successful in a concurrent volume group environment, all the concurrent nodes must be at AIX 4.3.2 or later.
4. The **chlvcopy** command is not allowed if the logical volume is in a volume group that has a snapshot volume group or a snapshot volume group.
5. **chfs** should be used to create a split mirror copy when a filesystem resides on the logical volume to be copied.
6. The **chlvcopy** command is not supported on encrypted logical volume.

All partitions of a logical volume must be fresh before **chlvcopy** can mark a mirror copy as a split mirror. Only one copy may be designated as an online split mirror copy.

Although the **chlvcopy** command can mark online split mirror copies on logical volumes that are open (including logical volumes containing mounted file systems), this is not recommended unless the application is at a known state at the time the copy is marked as a split mirror. The split mirror copy is internally consistent at the time the **chlvcopy** command is run, but consistency is lost between the logical volume and the split mirror copy if the logical volume is accessed by multiple processes simultaneously and the application is not at a known state. When marking an open logical volume, data may be lost or

corrupted. Logical volumes should be closed before marking online split mirror copies in order to avoid a potential corruption window.

If the persistence flag is not set to prevent the loss of backup data, the volume group should be set to not automatically varyon and the **-n** flag should be used with **varyonvg** to prevent stale partitions from being resynced. If the persistence flag (**-P**) is set, the following applies: In the event of a crash while an online split mirror copy exists (or multiples exist), the existence of copies is retained when the system is rebooted.

Flags

Item	Description
-b	Marks a mirror copy as a split mirror copy.
-c *copy*	Mirror copy to mark as split mirror copy. The allowed values of copy are 1, 2, or 3. If this option is not specified the default for copy is the last mirror copy of the logical volume.
-B	Unmarks a mirror as split mirror copy. It will also attempt to remove the child backup logical volume, if one was created with the **-l** option.
-f	Forces split mirror copy even if there are stale partitions. If used with the **-B** option, the child backup logical volume if one was created with the **-l** option, will be removed with the **force** option.
-l *newlvname*	New name of the backup logical volume. Specifying the **-l** flag also sets the persistence option, allowing applications to access split mirror copy via *newlvname*.
-P	Maintains information about the existence of an online split mirror copy across a reboot and also allows other nodes (in a concurrent mode environment) to be aware of the existence of the online split mirror copy.
-s	Starts a background **syncvg** for the logical volume.
-w	Allows split mirror copy to be writable (default is to create the split mirror copy as READ ONLY).
LV *name*	Logical volume to act on.

chmaster Command

Purpose

The **chmaster** command executes the **ypinit** command and restarts the NIS daemons to change a controller server.

Syntax

/usr/etc/yp/chmaster [**-s** *HostName* [**,** *HostName ...*]] [**-O** | **-o**] [**-E** | **-e**] [**-P** | **-p**] [**-U** | **-u**] [**-C** | **-c**] [**-I** | **-B** | **-N**]

Description

The **chmaster** command invokes the **ypinit** command to update the NIS maps for the current domain, assuming that the domain name of the system is currently set. After the **ypinit** command completes successfully, the **chmaster** command comments or uncomments the entries in the **/etc/rc.nfs** file for the **ypserv** command, **yppasswdd** command, **ypupdated** command, and **ypbind** command.

You could also use the System Management Interface Tool (SMIT) **smit chmaster** fast path to run this command.

Flags

Item	Description
-B	Updates the **/etc/rc.nfs** file to start the appropriate daemons, invokes the **ypinit** command, and starts the daemons.
-C	Starts the **ypbind** daemon along with the **ypserv** daemon. This flag is the default.
-c	Suppresses the start of the **ypbind** daemon.
-E	Exits from the **ypinit** command and the **chmaster** command if errors are encountered. This flag is the default.
-e	Suppresses an exit from the **ypinit** command and the **chmaster** command if errors are encountered.
-I	Directs the **chmaster** command to change the **/etc/rc.nfs** file to start the appropriate daemons on the next system restart. The execution of the **ypinit** command occurs when this command is invoked.
-N	Invokes the **ypinit** command and starts the appropriate daemons. No changes are made to the **/etc/rc.nfs** file.
-O	Overwrites existing maps for this domain.
-o	Prevents the overwriting of NIS maps. This flag is the default.
-P	Starts the **yppasswdd** daemon along with the **ypserv** daemon.
-p	Suppresses the start of the **yppasswdd** daemon. This flag is the default.
-s *HostName* [**,** *HostName*]	Specifies the worker host names for the worker for this controller server. The **chmaster** command automatically adds the current host to this list.
-U	Starts the **ypupdated** daemon along with the **ypserv** daemon.
-u	Suppresses the start of the **ypupdated** daemon. This flag is the default.

Examples

To invoke the **ypinit** command to rebuild the NIS maps for the current domain, enter:

```
chmaster -s chopin -O -p -u -B
```

In this example, the `chmaster` command overwrites the existing maps and the **yppasswdd** and **ypupdated** daemons are not started. The host name `chopin` is specified to be a worker server.

Files

Item	Description
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.
/var/yp/*domainname*	Contains the NIS maps for the NIS domain.

chmod Command

Purpose

Changes file modes.

Syntax

To Change File Modes Symbolically

chmod [**-R**] [**-h**] [**-f**] [[**u**] [**g**] [**o**] | [**a**]] { { **-** | **+** | **=** } [**r**] [**w**] [**x**] [**X**] [**s**] [**t**] } { *File ...* | *Directory ...* }

To Change File Modes Numerically

chmod [**-R**] [**-h**] [**-f**] *PermissionCode* { *File ...* | *Directory ...* }

Description

The **chmod** command modifies the mode bits and the extended access control lists (ACLs) of the specified files or directories. The mode can be defined symbolically or numerically (absolute mode).

When a symbolic link is encountered and you have not specified the **-h** flag, the **chmod** command changes the mode of the file or directory pointed to by the link and not the mode of the link itself. If you specify the **-h** flag, the **chmod** command prevents this mode change.

If you specify both the **-h** flag and the **-R** flag, the **chmod** command descends the specified directories recursively, and when a symbolic link is encountered, the mode of the file or directory pointed to by the link is not changed.

Flags

Item	Description
-f	Suppresses all error reporting except invalid permissions and usage statements.
-h	Suppresses a mode change for the file or directory pointed to by the encountered symbolic link. **Note:** This behavior is slightly different from the behavior of the **-h** flag on the **chgrp** and **chown** commands because mode bits cannot be set on symbolic links.
-R	Descends only directories recursively, as specified by the pattern *File...*|*Directory...*. The **-R** flag changes the file mode bits of each directory and of all files matching the specified pattern. See Example 6. When a symbolic link is encountered and the link points to a directory, the file mode bits of that directory are changed but the directory is not further traversed.

Symbolic Mode

To specify a mode in symbolic form, you must specify three sets of flags.

Note: Do not separate flags with spaces.

The first set of flags specifies who is granted or denied the specified permissions, as follows:

Item	Description
u	File owner.
g	Group and extended ACL entries pertaining to the file's group.
o	All others.

Item	Description
a	User, group, and all others. The **a** flag has the same effect as specifying the **ugo** flags together. If none of these flags are specified, the default is the **a** flag and the file creation mask (umask) is applied.

The second set of flags specifies whether the permissions are to be removed, applied, or set:

Item	Description
-	Removes specified permissions.
+	Applies specified permissions.
=	Clears the selected permission field and sets it to the permission specified. If you do not specify a permission following **=**, the **chmod** command removes all permissions from the selected field.

The third set of flags specifies the permissions that are to be removed, applied, or set:

Item	Description
r	Read permission.
w	Write permission.
x	Execute permission for files; search permission for directories.
X	Execute permission for files if the current (unmodified) mode bits have at least one of the user, group, or other execute bits set. The **X** flag is ignored if the *File* parameter is specified and none of the execute bits are set in the current mode bits. Search permission for directories.
s	Set-user-ID-on-execution permission if the **u** flag is specified or implied. Set-group-ID-on-execution permission if the **g** flag is specified or implied.
t	For directories, indicates that only file owners can link or unlink files in the specified directory. For files, sets the **save-text** attribute.

Numeric or Absolute Mode

The **chmod** command also permits you to use octal notation for the mode. The numeric mode is the sum of one or more of the following values:

Item	Description
4000	Sets user ID on execution.
2000	Sets group ID on execution.
1000	Sets the link permission to directories or sets the **save-text** attribute for files.
0400	Permits read by owner.
0200	Permits write by owner.
0100	Permits execute or search by owner.
0040	Permits read by group.
0020	Permits write by group.
0010	Permits execute or search by group.
0004	Permits read by others.
0002	Permits write by others.

Item	Description
0001	Permits execute or search by others.

Notes:

1. Specifying the mode numerically disables any extended ACLs. Refer to "Access control Lists" in *Operating system and device management* for more information.
2. Changing group access permissions symbolically also affects the AIXC ACL entries. The group entries in the ACL that are equal to the owning group of the file are denied any permission that is removed from the mode. Refer to "Access control Lists" in *Operating system and device management* for more information.
3. You can specify multiple symbolic modes separated with commas. Operations are performed in the order they appear from left to right.
4. You must specify the mode symbolically or use an explicit 4-character octal with a leading zero (for example, 0755) when removing the set-group-ID-on-execution permission from directories.
5. For a non-AIXC ACL associated file system object, any request (either symbolically or numerically) that results in a operation to change the base permissions bits (rwxrwxrwx) in mode bits results in replacement of the existing ACL with just the mode bits.
6. The save-text attribute can only be set by the root user, but it can be removed by regular users.

Exit Status

This command returns the following exit values:

Item	Description
0	The command executed successfully and all requested changes were made.
>0	An error occurred.

Security

Access Control

This program should be installed as a normal user program in the Trusted Computing Base.

Only the owner of the file or the root user can change the mode of a file.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To add a type of permission to several files:

   ```
   chmod  g+w  chap1  chap2
   ```

   This adds write permission for group members to the files chap1 and chap2**.**
2. To make several permission changes at once:

   ```
   chmod  go-w+x  mydir
   ```

This denies group members and others the permission to create or delete files in `mydir` (**go-w**) and allows group members and others to search `mydir` or use it in a path name (**go+x**). This is equivalent to the command sequence:

```
chmod  g-w  mydir
chmod  o-w  mydir
chmod  g+x  mydir
chmod  o+x  mydir
```

3. To permit only the owner to use a shell procedure as a command:

   ```
   chmod  u=rwx,go= cmd
   ```

   This gives read, write, and execute permission to the user who owns the file (**u=rwx**). It also denies the group and others the permission to access `cmd` in any way (**go=**).

   If you have permission to execute the `cmd` shell command file, then you can run it by entering:

   ```
   cmd
   ```

   > **Note:** Depending on the **PATH** shell variable, you may need to specify the full path to the `cmd` file.

4. To use Set-ID Modes:

   ```
   chmod  ug+s  cmd
   ```

   When the `cmd` command is executed, the effective user and group IDs are set to those that own the `cmd` file. Only the effective IDs associated with the child process that runs the `cmd` command are changed. The effective IDs of the shell session remain unchanged.

   This feature allows you to permit access to restricted files. Suppose that the `cmd` program has the Set-User-ID Mode enabled and is owned by a user called `dbms`. The user `dbms` is not actually a person, but might be associated with a database management system. The user `betty` does not have permission to access any of `dbms`'s data files. However, she does have permission to execute the `cmd` command. When she does so, her effective user ID is temporarily changed to `dbms`, so that the `cmd` program can access the data files owned by the user `dbms`.

   This way the user `betty` can use the `cmd` command to access the data files, but she cannot accidentally damage them with the standard shell commands.

5. To use the absolute mode form of the **chmod** command:

   ```
   chmod  644  text
   ```

   This sets read and write permission for the owner, and it sets read-only mode for the group and others. This also removes all extended ACLs that might be associated with the file.

6. To recursively descend directories and change file and directory permissions given the tree structure:

   ./dir1/dir2/file1

   ./dir1/dir2/file2

   ./dir1/file1

   enter this command sequence:

   ```
   chmod -R 777 f*
   ```

   which will change permissions on ./dir1/file1.

   But given the tree structure of:

   ./dir1/fdir2/file1

   ./dir1/fdir2/file2

./dir1/file3

the command sequence:

```
chmod -R 777 f*
```

will change permissions on:

./dir1/fdir2

./dir1/fdir2/file1

./dir1/fdir2/file2

./dir1/file3

File

Item	Description
/usr/bin/chmod	Contains the **chmod** command .

chmp Command

Purpose

Changes the characteristics of a mirror pool.

Syntax

chmp **-A** [**-c** *aiocachelvname*] [**-h** *highwatermark*] **-m** *mirrorpoolname vgname*

chmp **-h** *highwatermark* **-m** *mirrorpoolname vgname*

chmp **-S** [**-f**] **-m** *mirrorpoolname vgname*

Description

The **chmp** command can perform the following operations:

- Configure a mirror pool for asynchronous mirroring using the **-A** flag.
- Set the high watermark of the I/O-cache logical volume with the **-h** flag.
- Detect a change in size of the I/O-cache logical volume and take appropriate actions.
- Change the mirror pool from asynchronous mirroring to synchronous mirroring with the **-S** flag.
- Change the I/O-cache logical volume that is used for asynchronous mirroring.

Note:

1. All disks in all mirror pools must be accessible to be configured for asynchronous mirroring.
2. After a mirror pool is configured for asynchronous mirroring, some active disks are needed from each mirror pool to convert the mirror pool from asynchronous mirroring to synchronous mirroring. If you want to remove one or more mirror pools from a site that is down, disable asynchronous mirroring using the **chmp** command with the **-S** and **-f** flags.
3. Asynchronous mirroring is only supported on nonconcurrent scalable volume groups with mirror pools set to be super strict.
4. You must disable the auto-on and bad-block relocation options of the volume group.
5. The volume group cannot be a snapshot volume group. The volume group cannot contain active paging-space logical volumes.
6. The volume group must be varied on to make mirror pool changes.

7. You must use passive mirror write consistency for the **aio_cache** logical volume if it is mirrored.

Flags

Item	Description
-A	Configures a mirror pool for asynchronous mirroring.
-c *aiocachelvname*	Specifies the name of an asynchronous I/O-cache logical volume. The logical volume must be of the **aio_cache** type and must not reside in the mirror pool that is specified with the **-m** flag. If you do not specify the **-c** flag, the **chmp** command attempts to find the appropriate logical volume of the **aio_cache** type.
-f	Forces a mirror pool from asynchronous mirroring to synchronous mirroring, even if the remote I/O cache is not accessible.
-h *highwatermark*	Specifies the I/O-cache high watermark. The value is the percent of I/O cache size. The default value is 100%. The flag also detects an increase in size of the I/O-cache logical volumes and takes the appropriate action.
-m *mirrorpoolname*	Specifies the mirror pool name.
-S	Changes a mirror pool from asynchronous mirroring to synchronous mirroring.

Parameters

Item	Description
vgname	Specifies the volume group name where the mirror pool resides.

Examples

1. To set up an asynchronous mirroring for the mirror pool, enter the following sequence of commands:
   a. Create a scalable volume group with the mirror pool set to be super strict on local disks hdisk1, hdisk2, and hdisk3:

   ```
   mkvg -f -S -M s -y gmvg1 hdisk1 hdisk2 hdisk3
   ```

   b. Disable the volume group auto-on and bad-block relocation:

   ```
   chvg -a n -b n gmvg1
   ```

   c. Add the local disks into mirror pool MP1:

   ```
   chpv -p MP1 hdisk1 hdisk2 hdisk3
   ```

   d. Create a logical volume for user data:

   ```
   mklv -b n -p copy1=MP1 -y user_data_lv gmvg1 10
   ```

   e. Add the remote physical-volume devices hdisk4, hdisk5, and hdisk6 to mirror pool MP2 in volume group gmvg1:

   ```
   extendvg -f -p MP2 gmvg1 hdisk4 hdisk5 hdisk6
   ```

   f. Add the remote mirror copy in the volume group using the **mirrorvg** command:

   ```
   mirrorvg -c 2 -p copy2=MP2 gmvg1
   ```

   g. Add a logical volume of the **aio_cache** type in the local mirror pool:

Note: A mirror pool can contain only one I/O-cache logical volume. If the I/O-cache logical volume is mirrored, each copy must be in a local mirror pool.

```
mklv -t aio_cache -w p -p copy1=MP1 -y mp1_aiolv gmvg1 1
```

h. Set up asynchronous mirroring for mirror pool MP2:

```
chmp -A -c mp1_aiolv  -h 80 -m MP2 gmvg1
```

2. To change the mirror pool from asynchronous mirroring to synchronous mirroring, enter the following command:

```
chmp -S -m MP2 gmvg1
```

3. To change the mirroring attributes, such as high watermark, enter the following command:

```
chmp -h 90 -m MP2 gmvg
```

4. To replace the I/O-cache logical volume with a different I/O-cache logical volume, enter the following sequence of commands:

a. Change the mirror pool from asynchronous mirroring to synchronous mirroring:

```
chmp -S -m MP2 gmvg1
```

b. Remove the current I/O-cache logical volume `mp1_aiolv` that resides in mirror pool MP1:

```
rmlv mp1_aiolv
```

c. Create a new I/O-cache logical volume in mirror pool MP1:

```
mklv -t aio_cache -w p -p copy1=MP1 -y mp1_new_aiolv gmvg1 1
```

d. Set up asynchronous mirroring for mirror pool MP2 using the new I/O-cache logical volume:

```
chmp -A -c mp1_new_aiolv -h 90 -m MP2 gmvg1
```

chnamsv Command

Purpose

Changes TCP/IP-based name service configuration on a host.

Syntax

chnamsv [**-a"***Attribute***=***Value* ...**"** | **-A** *FileName*]

Description

The **chnamsv** high-level command changes a TCP/IP-based name service configuration on a host. The command changes the **/etc/resolv.conf** file only. The command does not change the name server database.

If you change the name service configuration for a client, the **chnamsv** command calls the **namerslv** low-level command to change the **resolv.conf** configuration file appropriately.

You could also use the System Management Interface Tool (SMIT) **smit namerslv** fast path to run this command.

Flags

Item	Description
-A *FileName*	Specifies name of file containing the named server initialization information.
-a"*Attribute***=***Value...***"**	Specifies a list of attributes and their corresponding values to be used for updating the named server initialization files in the database. Attributes can be either of the following: **domain** The domain name of the name server. **nameserver** The Internet address of the name server.

Examples

1. To update the named server initialization files, enter the command in the following format:

   ```
   chnamsv -a "domain=austin.century.com nameserver=192.9.200.1"
   ```

   In this example the domain name and name server address are updated. The previous domain is overwritten and a new nameserver entry is appended.
2. To update name server initialization files according to information in another file, enter the command in the following format:

   ```
   chnamsv -A namsv.file
   ```

   In this example, the file that contains the updated information is `namsv.file`.

Files

Item	Description
/etc/ resolv.conf	Contains DOMAIN name server information for local resolver routines.

chnfs Command

Purpose

Changes the configuration of the system to invoke a specified number of **nfsd** daemons or to change NFS global configuration values.

Syntax

chnfs [-b *NumberofBiod*] [**-n** *NumberOfNfsd*] [**-l** *NumberOfLockd*] [**-I** | **-B** | **-N**] [-s | -S] [-v | -V] [-r *v4_root_node*] [-p *v4_public_node*] [-L *v4_lease_time*] [-R {on|off|*host*[+*host*]}] [-g on | off] [-x *xtend_cnt*] [-P *SS_pathname*] [-G add | remove]

Description

The **chnfs** command invokes the number of **nfsd** daemons specified. The **chnfs** command does this by changing the objects in the SRC database. The `chnfs` command also is used to enable or disable the use of advanced security methods by NFS or to enable or disable the use of NFS Version 4. These changes take place at different times depending on the flags chosen.

Note: The **chnfs** command does not change the number of biod threads. To change the number of biod threads, use the NFS-specific -o biods=*n* option of the mount command. For example, to specify that an NFS mount use 16 biod threads, type:

```
mount -obiods=16 server:/tmp /mnt
```

By default, a v2 mount uses 7 biod threads, and a v3 mount and a v4 mount use 32 biod threads.

Flags

Item	Description
-B	Temporarily stops the daemons currently running on the system, modifies the SRC database code to reflect the new number, and restarts the daemons indicated. This flag is a default.
-b *NumberofBiod*	Specifies the number of biod threads on the client. This option has no effect and should not be used.
-G add | remove	Controls the NFSv4 Grace Period bypass. When add is specified as the value of this flag, the grace period is bypassed regardless of how the -g option is specified and the **-gpbypass** flag is added to the **nfsd** argument. When remove is specified as the value of this flag, the **-gpbypass** flag is removed from the **nfsd** argument.
-g on|off	Controls the NFSv4 Grace Period enablement. The possible values are on or off. When no -g option is specified, the grace period is disabled by default.
-I	Changes the objects in the SRC database so that the number of daemons specified will be run during the next system restart.
-L *v4_lease_time*	Specifies the lease time that the state manager uses when granting a lock to a client. This flag sets the NFS Version 4 lease time in seconds. The lease time also affects the length of the grace period, the time when a client is deemed dead or expired, and the duration of time that a client has before getting timed out. The valid range is from 10 to 600 seconds. The default value is 120 seconds. This flag is valid only for NFS Version 4.
-l *NumberOfLockd*	Specifies the number of **lockd** daemons to run on the system.
-N	Temporarily stops the daemons currently running on the system and restarts the number of daemons indicated.
-n *NumberOfNfsd*	Specifies the number of **nfsd** daemons to run on the system.
P *SS_pathname*	Specifies the location for stable storage. If grace period is enabled, the state manager begins logging in state information in this pathname. If the filesystem is small, the state manager also allocates space initially. The default location for the stable storage pathname is **/var/adm/nfsv4**.
-p *v4_public_node*	Changes the NFS Version 4 public directory to the specified directory. The directory must be a subdirectory of the root directory. The public directory cannot be changed if any directories are currently exported for Version 4 use.
-R {on | off | *host*[+*host*]}	Enables or disables NFS Version 4 replication. If replication is enabled, replica locations can be specified for Version 4 exports. If replication is not enabled, attempts to export a directory with replica locations will fail. If any directories are exported for NFS Version 4 use, the replication mode cannot be changed. Changing the replication mode of the NFS server can cause errors on clients holding filehandles issued under the previous replication mode. If the *host*[+*host*] form is used, replication is enabled and the host list is used as the replica locations for the nfsroot.

Item	Description
-r *v4_root_node*	Changes the NFS Version 4 root location to the specified directory. Version 4 clients that mount / will see the specified directory as the server's root. The public directory cannot be changed if any directories are currently exported for Version 4 use.
-S	Enable RPCSEC_GSS. This enables NFS to use the enhanced security offered by RPCSEC_GSS, such as Kerberos 5.
-s	Disable RPCSEC_GSS. This disables the use of RPCSEC_GSS methods by NFS.
-V	Enable NFS Version 4.
-v	Disable NFS Version 4.
-x *xtend_cnt*	Controls the NFSv4 Grace Period automatic extension. The *extend_cnt* parameter specifies the total number of automatic extensions allowed for the grace period. If no -x option is specified, the number of allowed automatic extensions defaults to 1. A single extension cannot extend the grace period for more than the length of the NFSv4 lease period. The NFSv4 subsystem uses runtime metrics (such as the time of the last successful NFSv4 reclaim operation) to detect reclamation of the state in progress, and extends the grace period for a length of time up to the duration of the given number of iterations.

Examples

To set the number of **nfsd** daemons to 10, enter:

```
chnfs    -n    10    -I
```

This change will be made for the next system restart.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

chnfsdom Command

Purpose

Displays or changes the local NFS domain.

Syntax

chnfsdom [*LocalDomain*]

Description

The chnfsdom command changes the local NFS domain of the system. The local NFS domain is stored in the /etc/nfs/local_domain file. If no argument is specified, the command displays the current local NFS domain.

Parameters

Item	Description
LocalDomain	The new domain name.

Security

Users must have root authority.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
`/etc/nfs/local_domain`	Stores the local NFS domain name.

chnfsexp Command

Purpose

Changes the options used to export a directory to NFS clients.

Syntax

/usr/sbin/chnfsexp -d *Directory* [**-V** *ExportedVersion*] [**-f** *Exports_file*] [**-e** *ExternalName*] [**-t** { **rw** | **ro** | **remove** } {**rm -h** *HostName* [**,***HostName ...*] }] [**-a** *UID*] [**-r** *HostName* [**,** *HostName ...*]] [**-c** *HostName* **,** *HostName ...*]] [**-D** {yes | no}] [**-s** | **-n**] [**-S** *flavor*] [-G *rootpath@host*[+*host*][**:** *rootpath@host*[+*host*]]] [-g *rootpath@host*[+*host*][**:** *rootpath@host*[+*host*]]] [**-o** *Ordering*] [**-x**] [**-X**] [**-I** | **-B** | **-N**] [**-P** | **-p**] [**-v** *number* [**,** *number ...*]]

Description

The **chnfsexp** command takes as a parameter a directory that is currently exported to NFS clients and changes the options used to export that directory. The options specified on the command line will replace those currently being used.

Flags

Item	Description
-a *UID*	Uses the *UID* parameter as the effective user ID only if a request comes from an unknown user. The default value of this option is -2. **Note:** Root users (uid 0) are always considered "unknown" by the NFS server, unless they are included in the root option. Setting the value of *UID* to -1 disables anonymous access. The *UID* parameter can be either uid or username.
-B	Updates the entry in the **/etc/exports** file and the **exportfs** command is executed to again export the directory immediately.

Item	Description
-c *HostName* [**,** *HostName*] ...	Gives mount access to each of the clients listed. A client can either be a host or a netgroup. The default is to allow all hosts access.
-d *Directory*	Specifies the exported directory that is to be changed.
-D {yes | no}	Enables or disables file delegation for the specified export. This option overrides the system-wide delegation enablement for this export. The system-wide enablement is done through **nfso**.
-e *ExternalName*	Exports the directory specified by the *ExternalName* parameter. The external name must begin with the `nfsroot` name. This option is useful if you have run the `chnfs -r` command to change root to something other than `/`. See the description of the `/etc/exports` file for a description of the `nfsroot` name. This option applies only to directories exported for access by the NFS version 4 protocol.
-f *Exports_file*	Specifies the full path name of the exports file to use if other than the **/etc/exports** file.
-G *rootpath@host*[+*host*] [:*rootpath@host*[+*host*]]	A namespace referral will be created at the specified path. The referral directs clients to the specified alternate locations where they can continue operations. A referral is a special object. If a nonreferral object exists at the specified path, the export is disallowed and an error message is printed. If nothing exists at the specified path, a referral object is created there that includes the path name directories leading to the object. A referral cannot be specified for the `nfsroot`. The name `localhost` cannot be used as a *hostname*. The `-G` option is allowed only for version 4 exports. If the export specification allows version 2 or version 3 access, an error message will be printed and the export will be disallowed. The administrator should ensure that appropriate data exists at the referral locations. **Note:** A referral or replica export can only be made if replication is enabled on the server. Use `chnfs -R on` to enable replication.

Item	Description
-g *rootpath@host*[*+host*] [:*rootpath@host*[*+host*]]	The specified directory will be marked with replica information. If the server becomes unreachable by an NFS client, the client can switch to one of the specified servers. This option is only accessible using NFS version 4 protocol, and version 4 access must be specified in the options. Because the directory is being exported for client access, specifying NFS version 2 or version 3 access will not cause an error, but the request will simply be ignored by the version 2 or version 3 server. This option cannot be specified with the `-G` flag. Only the host part of each specification is verified. The administrator must ensure that the specified *rootpaths* are valid and that the target servers contain appropriate data. If the directory being exported is not in the replica list, that directory will be added as the first replica location. The administrator should ensure that appropriate data exists at the replica locations. The `-g` option is available only on AIX 5.3 with 5300-03 or later. **Note:** A referral or replica export can only be made if replication is enabled on the server. Use `chnfs -R on` to enable replication.
-h *Hostname* [**,** *HostName*] ...	Specifies which hosts have read-write access to the directory. This option is valid only when the directory is exported read-mostly.
-I	Adds an entry in the **/etc/exports** file so that the next time the **exportfs** command is run, usually during system restart, the directory will be exported.
-N	Does not modify the entry in the **/etc/exports** file but the **exportfs** command is run with the correct parameters so that the export is changed.
-n	Does not require client to use the more secure protocol. This flag is the default.
-o *Ordering*	Defines how the alternate locations list is generated from the servers that are specified on the **refer** or **replicas** option of the **exportfs** command. The option applies only to directories exported for access by NFS version 4 protocol. The *Ordering* parameter has the following values: **full** All of the servers are scattered to form the combinations of alternate locations. **partial** The first location of all combinations is fixed to the first server that is specified on the **refer** or **replicas** option of the **exportfs** command. The remaining locations besides the first location are scattered as if they are scattered using the `scatter=full` method. **none** No scatter is to be used. The value can also be used to disable scattering if you previously enabled it.
-P	Specifies that the exported directory is to be a public directory.

Item	Description
-p	Specifies that the exported directory is not a public directory.
-r *HostName* [**,** *HostName*] ...	Gives root users on specified hosts access to the directory. The default is for no hosts to be granted root access.
-s	Requires clients to use a more secure protocol when accessing the directory.
-S *flavor*	May be used in conjunction with the **-c**, **-t**, or **-r** options to specify which occurrence of the option to change. Most **exportfs** options can be clustered using the **sec** option. Any number of **sec** stanzas may be specified, but each security method can be specified only once. If the entry in **/etc/exports** specified by the **-d** option contains a clause of the specified flavor, then that clause is updated to reflect the new parameters. Otherwise, a new **sec=** clause with the specified parameters will be appended to the current options list. Allowable flavor values are: **sys** UNIX authentication. **dh** DES authentication. **none** Use the anonymous ID if it has a value other than -1. Otherwise, a weak auth error is returned. **krb5** Kerberos. Authentication only. **krb5i** Kerberos. Authentication and integrity. **krb5p** Authentication, integrity, and privacy.
-t *Type*	Specifies one of the following types of mount access allowed to clients: **rw** Exports the directory with read-write permission. This is the default. **ro** Exports the directory with read-only permission. **remove** You must specify the **-t remove** option with the **-S** *flavor* option. Both the security flavor and the type of mount access (**rw**, **ro**, or **rm**) from the existing NFS export for the specified security flavor are removed. **rm** Exports the directory with read-mostly permission. If this type is chosen, the **-h** flag must be used to specify hosts that have read-write permission.
-v *number* [**,** *number ...*]	The directory specified by the **-d** option is made available to clients using the specified NFS versions. Valid values are 2, 3, or 4.

Item	Description
-V *ExportedVersion*	Specifies the version of the exported directory that is to be changed. Valid version numbers are 2, 3 and 4.
-x	Accepts the replica location information specified with the **-g** option as-is. Does not insert the server's primary hostname into the list if it is not present. This flag is intended for use with servers with multiple network interfaces. If none of the server's hostnames are in the replica list, NFSv4 clients might treat the location information as faulty and discard it.
-X	Enables the primary host name to be automatically inserted into the replica list. If you do not specify the primary host name of the server in the replica list, the host name is added as the first replica location.

Examples

1. To change the list of hosts that have access to an exported directory and to make this change occur immediately and upon each subsequent system restart, enter:

   ```
   chnfsexp     -d     /usr     -t     rw     -c     host1,host3,host29,grp3,grp2     -B
   ```

   In this example, the `chnfsexp` command changes the attributes of the `/usr` directory to give read and write permission to the `host1`, `host3`, and `host29` hosts, and the `grp3` and `grp2` netgroups.
2. To change the list of hosts that have access to an exported directory, to specify the path name of the exports file, and to make this change occur immediately and upon each subsequent system restart, enter:

   ```
   chnfsexp     -d     /usr     -t     rw     -c     host1,host3,host29,grp3,grp2
                -f     /etc/exports.other     -B
   ```

   In this example, the `chnfsexp` command changes the attributes of the `/usr` directory to give read and write permission to the `host1`, `host3`, and `host29` hosts: the `grp3` and `grp2` netgroups; and specifies the path name of the exports file as `/etc/exports.other`.
3. To change the version accessibility of the **/common/documents** directory to allow access only to clients using NFS version 4 protocol, enter:

   ```
   chnfsexp -d /common/documents -v 4
   ```

4. To change the root access of the **/common/documents** directory to `client1` and `client2` for clients using krb5 access, enter:

   ```
   chnfsexp -d /common/documents -S krb5 -r client1,client2
   ```

5. To change the options for the **/common/documents** directory that is exported only as version 3, enter the following command:

   ```
   chnfsexp -d /common/documents -V 3 -S krb5
   ```

6. To do a full scatter for the alternate locations specified in **refer** or **replicas** option for the **/common/documents** directory, enter the following command:

   ```
   chnfsexp -d /common/documents -o full
   ```

7. To add a list of alternate replica locations and do a partial scatter for the **/common/doc** directory, enter the following command:

   ```
   chnfsexp -d /common/doc -g /common/doc@s1:/common/doc@s2:/common/doc@s3 -o partial
   ```

Files

Item	Description
/etc/exports	Lists directories the server can export.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

chnfsim Command

Purpose

Changes NFS foreign identity mappings.

Syntax

For user and group related foreign identity mappings

`chnfsim` `-a` | `-l` | `-s` | `-x` `-u` | `-g` [`-i` *Identity*] [`-n` *name* `-d` *domain*]

For realm-to-domain mappings

`chnfsim` `-a` | `-l` | `-x` [`-r` *realm* `-d` *domain*]

To configure a system to use EIM

`chnfsim` `-c` `-a` | `-l` | `-x` [`-t` *type* `-h` *hostname*[:*port*] `-e` *EIMdomain* `-f` *EIMsuffix* `-b` *admin_DN* `-w` *admin_password* `-W` *access_password*]

To remove EIM configuration from a system

`chnfsim` `-C`

Description

The `chnfsim` command administers NFS foreign identity mappings using the Enterprise Identity Mapping (EIM) layer of an LDAP server. To use this command, the `bos.eim.rte` and `ldap.client` filesets must be installed. Additionally, if the machine is to be the EIM LDAP server, the `ldap.server` fileset must also be installed.

After changing identity mappings on the system, run the **nfsrgyd -f** command to flush the systems' identity cache.

You must first configure a system to use EIM with the `-c` and the **-a** flags before attempting to use any other function. All mapping data are stored and retrieved from the EIM LDAP server.

The `chnfsim` command is used to add, list, and remove an EIM configuration for NFS. The `chnfsim` command is then used to add and remove owner and owner group strings to user and group identities. It can list the identity mappings associated with a user or group, and can search for the mapping identity associated with a name and domain.

The `chnfsim` command is also used to add and remove Kerberos realm to NFS domain mappings, and can list the current realm to domain mappings.

Flags

Item	Description
`-a`	Add operation.
`-b`	Specifies the LDAP administrator distinguished name. The default value is `admin`.
`-c`	Configure operation.
`-C`	Remove EIM configuration.
`-d`	Specify the NFS domain part of a NFS V4 owner string.
`-e`	Specify the EIM domain of the EIM LDAP server used for NFS mapping.
`-f`	Specify the EIM directory suffix of the EIM LDAP server used for NFS mapping.
`-g`	Specify a group-based operation.
`-h`	Specify the hostname and port of the EIM LDAP server used for NFS mapping.
`-i`	Specify the mapping identity. This is a unique string that describes a particular owner or owner group.
`-l`	List operation.
`-n`	Specify the owner or owner group name of a NFS V4 owner string.
`-r`	Specify the Kerberos realm.
`-s`	Search operation.
`-t`	Specify the type of EIM LDAP server. **p | P** Primary LDAP server. **s | S** Secondary (default) LDAP server.
`-u`	Specify a user-based operation.
`-w`	Specify the EIM administrator password.
`-W`	Specify the EIM access-only user password.
`-x`	Remove operation.

Action Matrix

Item	Description
Operation	**Flags (Optional flags in parentheses)**
`-c`	Displays current EIM configuration of the system. **-a -t -h -e -f -w (-b -W)** Configures the system for EIM use. The `-w` flag is required if the specified *hostname* is the local system. If the *hostname* is not the local system, at least one of the `-w` or the `-W` flag must be specified. The NFS client or server can be configured for more than one EIM LDAP replica server. **-l -h** Lists the configuration details of the server *hostname*[:*port*] from the configuration file. **-x -h** Deletes the configuration details of the server *hostname*[:*port*] from the configuration file.

Item	Description
`-a`	**-u -i (-n -d)** Adds the user mapping identity. If the `-n` and `-d` flags are specified, that identity mapping is associated to the user mapping identity. **-g -i (-n -d)** Adds the group mapping identity. If the `-n` and `-d` flags are specified, that identity mapping is associated to the group mapping identity. **-r -d** Adds a realm-to-domain mapping.
`-x`	**-u -i (-n -d)** Removes the user mapping identity. If the `-n` and `-d` flags are specified, only that identity mapping is removed from the user mapping identity **-g -i (-n -d)** Removes the group mapping identity. If the `-n` and `-d` flags are specified, only that identity mapping is removed from the group mapping identity **-r -d** Removes a realm-to-domain mapping.
`-l`	Lists all realm-to-domain mappings. **-u -i** Lists all identity mappings associated with the specified user mapping identity. **-g -i** Lists all identity mappings associated with the specified group mapping identity.
`-s`	**-u -n -d** Searches for user mapping identities associated with the specified identity mapping. **-g -n -d** Searches for group mapping identities associated with the specified identity mapping.
`-C`	Removes all of the EIM LDAP server entries from the configuration file.

Exit Status

0
Request was successful.

EACCES
Not enough permissions to access data.

ENOENT
The mapping identity, name, domain, or realm was not found in the database; or the configuration file was not found.

EBUSY
EIM server is unable to allocate internal objects.

ECONVERT
Data conversion error.

EINVAL
Input parameter was not valid.

ENOMEM
Unable to allocate memory.

ENOTCONN
LDAP connection has not been made.

EUNKNOWN
Unknown exception occurred.

Examples

1. To display the current EIM configuration for NFS, use the following command:

   ```
   chnfsim -c
   ```

2. To configure a system to use EIM for NFS foreign identity mapping, use the following command:

   ```
   chnfsim -c -a -t P -h foos.com -e nfs -f nfseim -w mypasswd -W access_passwd
   ```

   Note: If the *hostname* specified is the local system, the `chnfsim` command also sets up an LDAP server to run EIM.

3. To configure a client system to use EIM for NFS foreign identity mapping, use the following command:

   ```
   chnfsim -c -a -t P -h foos.com -e nfs -f nfseim -W access_passwd
   ```

   Note: This configures the client with the primary LDAP server (for read-only access). Here, the specified host name is not the local system.

4. To list the configuration details of a server from the configuration file, use the following command:

   ```
   chnfsim -c -l -h foos.com:1080
   ```

5. To delete the configuration details of a server from the configuration file, use the following command:

   ```
   chnfsim -c -x -h foos.com:1080
   ```

6. To add a user identity mapping that specifies "John Doe" to "jdoe@com.com", use the following command:

   ```
   chnfsim -a -u -i "John Doe" -n jdoe -d com.com
   ```

   Note: This command will create an EIM identity for "John Doe" if one does not already exist.

7. To remove the user identity mapping that specifies "John Doe" to "jdoe@com.com", use the following command:

   ```
   chnfsim -x -u -i "John Doe" -n jdoe -d com.com
   ```

8. To remove all identity mappings for the user "John Doe", use the following command:

   ```
   chnfsim -x -u -i "John Doe"
   ```

9. To list all identity mappings for the user "John Doe", use the following command:

   ```
   chnfsim -l -u -i "John Doe"
   ```

10. To add a realm-to-domain mapping that specifies "realm1" maps to "domain1", use the following command:

    ```
    chnfsim -a -r realm1 -d domain1
    ```

11. To remove the realm-to-domain mapping that specifies "realm1" maps to "domain1", use the following command:

    ```
    chnfsim -x -r realm1 -d domain1
    ```

12. To list all realm-to-domain mappings, use the following command:

    ```
    chnfsim -l
    ```

13. To search for the user mapping identity associated with "jdoe@com.com", use the following command:

```
chnfsim -s -u -n jdoe -d com.com
```

14. To remove all EIM configuration from a system, use the following command:

```
chnfsim -C
```

Note: This does not remove the underlying LDAP database or entries.

Files

Item	Description
`/usr/sbin/chnfsim`	Location of the `chnfsim` command.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

chnfsmnt Command

Purpose

Changes the options used to mount a directory from an NFS server.

Syntax

/usr/sbin/chnfsmnt **-f** *PathName* **-d** *RemoteDirectory* **-h** *RemoteHost* [**-t** { **rw** | **ro** }] [**-m** *MountTypeName*] [**-w** { **fg** | **bg** }] [**-X** | **-x**] [**-S** | **-H**] [**-Y** | **-y**] [**-Z** | **-z**] [**-e** | **-E**] [**-a** | **-A**] [**-j** | [**-J**] [**-q** | [**-Q**] [**-g** | [**-G**] [**-s** | **-n**] [**-I** | **-B** | **-N**] [**-r** *TimesToRetry*] [**-R** *NumRetrans*] [**-b** *ReadBufferSize*] [**-c** *WriteBufferSize*] [**-o** *TimeOut*] [**-P** *PortNumber*] [**-u** *AcRegMin*] [**-U** *AcRegMax*] [**-v** *AcDirMin*] [**-V** *AcDirMax*] [**-T** *AcTimeO*] [**-p** *NumBiods*] [**-K** { *any* | *2* | *3* }] [**-k** { *any* | *tcp* | *udp* }] [**-M** *security_methods*] [**-i** { *dio* | *cio* [,*cior*] }]

Description

The **chnfsmnt** command changes the mount options of a currently mounted file system. However, before you can change the attributes of a mount, the **/etc/filesystems** file must contain an entry for the file system. This command unmounts the directory, changes the specified options, and mounts the directory with the new options.

Flags

Item	Description
-A	The **/etc/filesystems** entry for this file system will specify that it should be automatically mounted at system restart.
-a	The **/etc/filesystems** entry for this file system specifies that it should not be automatically mounted at system restart. This is the default.
-B	Modifies the entry in the **/etc/filesystems** file and remounts the file system using the flags and parameters specified. This flag is the default.
-b *ReadBufferSize*	Indicates the size of the read buffer in *N* bytes.

Item	Description
-c *WriteBufferSize*	Indicates the size of the write buffer in *N* bytes.
-d *RemoteDirectory*	Specifies the directory that will be mounted on the path name specified.
-E	Allows keyboard interrupts on hard mounts.
-e	Prevents keyboard interrupts on hard mounts. This flag is the default.
-f *PathName*	Specifies the mount point for the directory.
-G	Directs any file or directory created on the file system to inherit the group ID of the parent directory.
-g	Does not direct new files or directories created on the file system to inherit the group ID of the parent directory. This is the default.
-H	Makes the mount a hard mount, which causes the client to continue trying until the server responds.
-h *RemoteHost*	Specifies the NFS server that is exporting the directory.
-I	Changes the entry in the **/etc/filesystems** file but does not remount the directory.
-i	Specifies I/O mode for the mount. The options are: ***dio*** Specifies direct I/O mode. ***cio*** Specifies concurrent I/O mode. ***cior*** Specifies concurrent I/O with read-only mode.
-J	Indicates that **acl**s are used on this mount.
-j	Indicates that **acl**s are not used on this mount. This is the default.
-K	Specifies the NFS version used for this NFS mount. The options are: ***any*** Uses the mount command to determine the correct match, first attempting the highest NFS version available. **2** Specifies NFS Version 2. **3** Specifies NFS Version 3.
-k	Specifies the transport protocol used for the mount. The options are: ***any*** Uses the mount command to select the protocol to use. TCP protocol is the preferred protocol. ***tcp*** Specifies the TCP protocol. ***udp*** Specifies the UDP protocol.

Item	Description
-M *security_methods*	A list of security methods to use when attempting the mount. A comma separated list of the values `sys`, dh, `krb5`, `krb5i`, `krb5p`, which correspond to UNIX, DES, Kerberos 5, Kerberos 5 with integrity, and Kerberos 5 with privacy. Multiple values are allowed, but are only meaningful with NFS version 4 mounts. If multiple methods are given for a version 2 or 3 protocol mount, the first method will be used. For a NFS version 4 mount, the methods will be tried in listed order.
-m *MountTypeName*	Corresponds to the *type* field in the stanza of the entry in the **/etc/filesystems** file. When the **mount -t** command *MountTypeName* is issued, all of the currently unmounted file systems with a field type equal to the string are mounted.
-N	Prevents modification of the corresponding entry in the **/etc/filesystems** file if it exists. If the directory is currently mounted, it is unmounted and then mounted again with the flags and parameters specified.
-n	Instructs the mount not to use a more secure protocol. This flag is the default.
-o *TimeOut*	Indicates the length of the NFS time out in *N* tenths of a second.
-P *PortNumber*	Indicates the IP port number for the server.
-p *NumBiods*	Specifies the number of **biod** daemons that are allowed to work on a particular file system. The default is 7 for NFS version 2 and 32 for NFS version 3 and NFS version 4.
-Q	Requests that no posix pathconf information be exchanged and made available on an NFS Version 2 mount. Requires a mount Version 2 **rpc.mountd** at the NFS server.
-q	Specifies that no posix pathconf information is exchanged if mounted as an NFS Version 2 mount. This is the default.
-r *TimeToRetry*	Indicates the number of times to retry a mount. The default is 1000.
-R *NumRetrans*	Specifies, for a soft mount, the number of times that a request is to be transmitted if it is not acknowledged by the server. If the request goes unacknowledged after *NumRetrans* transmissions, the client gives up on the request. If this flag is not specified, the default value of 3 is used.
-S	Makes the mount a soft mount, which means that the system returns an error if the server does not respond.
-s	Instructs the mount to use a more secure protocol.
-T*AcTimeO*	Sets minimum and maximum time allowed for regular files and directories to *AcTimeO* seconds. If this option is specified, the other cached attribute times are overridden.
-t	Specifies whether the directory will be mounted as read-write or read-only. **rw** Mounts the directory read-write. This type is the default for the system. **ro** Mounts the directory read-only.
-U *AcRegMax*	Holds cached attributes for no more than *AcRegMax* seconds after file modification.

Item	Description
-u *AcRegMin*	Holds cached attributes for at least *AcRegMin* seconds after file modification.
-V *AcDirMax*	Holds cached attributes for no more than *AcDirMax* seconds after directory update.
-v *AcDirMin*	Holds cached attributes for at least *AcDirMin* seconds after directory update.
-w { **fg** | **bg** }	Indicates whether the mount should be attempted in the foreground (fg) or background (bg). If **bg** is specified and the attempt to mount the directory fails, the mount will be tried again in the background. The **fg** parameter is the default.
-X	Specifies that the server does support long device numbers. This is the default.
-x	Specifies that the server does not support long device numbers.
-Y	Indicates that the execution of **suid** and **sgid** programs are allowed in this file system. This is the default.
-y	Indicates that the execution of **suid** and **sgid** programs is not allowed in this file system.
-Z	Indicates that device access through this mount is allowed. This is the default.
-z	Indicates that device access through this mount is not allowed.

Examples

To change a mount to read-only, enter:

```
chnfsmnt -f /usr/man -d /usr/man -h host1 -t ro
```

In this example, the `chnfsmnt` command changes the attributes of the mounted directory to read-only.

Files

Item	Description
/etc/filesystems	Lists the remote file systems to be mounted during the system restart.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

chnfsrtd Command

Purpose

Changes the local NFS realm-to-domain mappings.

Syntax

`chnfsrtd` [`-a` *RealmDomain*] [`-e` *OldRealm OldDomain NewRealm NewDomain*] [`-r` *RealmDomain*]

Description

The `chnfsrtd` command administers the local realm-to-domain mappings of the system. The local realm-to-domain mappings are stored in the `/etc/nfs/realm.map` file.

Note: Use the `chnfsdom` command to list the current realm-to-domain mappings.

Flags

Item	Description
`-a` *RealmDomain*	Adds a new realm-to-domain mapping.
`-e` *OldRealm OldDomain NewRealm NewDomain*	Edits an existing realm-to-domain mapping.
`-r` *RealmDomain*	Removes a realm-to-domain mapping.

Security

Users must have root authority to use the `chnfsrtd` command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To add a new realm-to-domain mapping, type:

   ```
   chnfsrtd -a realm1 domain1
   ```

   This commands appends `realm1 domain1` to the `/etc/nfs/realm.map` file.
2. To remove a realm-to-domain mapping, type the following:

   ```
   chnfsrtd -r realm2 domain2
   ```

   This command removes `realm2 domain2` from the `/etc/nfs/realm.map` file, if that mapping exists.
3. To edit an existing realm-to-domain mapping, type:

   ```
   chnfsrtd -e realm3 domain3 realm4 domain4
   ```

   This command changes the `realm3 domain3` mapping to `realm4 domain4` in the `/etc/nfs/realm.map` file, if that mapping exists.

Files

Item	Description
`/etc/nfs/realm.map`	Stores the local realm-to-domain mappings.

chnfssec Command

Purpose

Changes the default security flavor used by the network file system (NFS) client.

Syntax

chnfssec [-a | -r] *comma-separated-list*

Description

The chnfssec command administers the default security flavors used by the NFS client. These defaults are stored in the /etc/nfs/security_default file. Use the chnfssec command (without flags) to list the current security flavors. The **/etc/nfs/security_default** file must exist for the **chnfssec** command to list or remove security flavors. Otherwise, the **chnfssec** command fails, and returns an error.

The valid security flavors available are: ,

```
sys          UNIX style (uids, gids)
dh           DES style (encrypted timestamps)
krb5         Kerberos 5, no integrity or privacy
krb5i        Kerberos 5, with integrity
krb5p        Kerberos 5, with privacy
```

Flags

Item	Description
-a	Sets a new list of security flavors.
-r	Removes a set of security flavors.

Parameters

Item	Description
comma-separated-list	sys, dh, krb5, krb5i, krb5p are the available flavors.

Security

Users must have root authority to use the chnfssec command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To add a list of security flavors, type:

   ```
   chnfssec -a krb5,krb5i,sys
   ```

   This command tells the NFS client to first use krb5, then krb5i, and lastly sys security.
2. To remove a security flavor, type the following:

   ```
   chnfssec -r krb5,sys
   ```

   This command removes krb5 and sys from the list of security flavors the NFS client will use.

Files

Item	Description
/etc/nfs/security_default	Stores the default NFS security flavors.

chnlspath Command

Purpose

Modify the value of the secure **NLSPATH** system configuration variable.

Syntax

chnlspath [-p] *NlspathValue*

Description

The **chnlspath** command is used to modify the secure **NLSPATH** system configuration variable.

Flags

Item	Description
-p *NlspathValue*	Specifies the path that the secure **NLSPATH** system configuration variable is set to. In this flag, the **-p** flag is optional.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

chown Command

Purpose

Changes the owner or group associated with a file.

Syntax

chown [**-f**] [**-h**] [**-R**] *Owner* [**:***Group*] { *File ...* | *Directory ...* }

chown -R [**-f**] [**-H** | **-L** | **-P**] *Owner* [**:***Group*] { *File ...* | *Directory ...* }

Description

The **chown** command changes the owner of the file or directory specified by the *File* or *Directory* parameter to the user specified by the *Owner* parameter. The value of the *Owner* parameter can be a user name from the user database or a numeric user ID. Optionally, a group can also be specified. The value of the *Group* parameter can be a group name from the group database or a numeric group ID.

Only the root user can change the owner of a file. You can change the group of a file only if you are a root user or if you own the file. If you own the file but are not a root user, you can change the group only to a group of which you are a member.

Although the **-H**, **-L** and **-P** flags are mutually exclusive, specifying more than one is not considered an error. The last flag specified determines the behavior that the command will exhibit.

When a symbolic link is encountered and you have not specified the **-h** flag, the **chown** command changes the ownership of the file or directory pointed to by the link and not the ownership of the link itself.

If you specify the **-h** flag, the **chown** command has the opposite effect and changes the ownership of the link itself and not that of the file or directory pointed to by the link.

If you specify the **-R** flag, the **chown** command recursively descends the specified directories.

If you specify both the **-h** flag and the **-R** flag, the **chown** command descends the specified directories recursively, and when a symbolic link is encountered, the ownership of the link itself is changed and not that of the file or directory pointed to by the link.

Flags

Item	Description
-f	Suppresses all error messages except usage messages.
-h	Changes the ownership of an encountered symbolic link and not that of the file or directory pointed to by the symbolic link.
-H	If the **-R** option is specified and a symbolic link referencing a file of type directory is specified on the command line, the **chown** command shall change the user ID (and group ID, if specified) of the directory referenced by the symbolic link and all files in the file hierarchy below it.
-L	If the **-R** option is specified and a symbolic link referencing a file of type directory is specified on the command line or encountered during the traversal of a file hierarchy, the **chown** command shall change the user ID (and group ID, if specified) of the directory referenced by the symbolic link and all files in the file hierarchy below it.
-P	If the **-R** option is specified and a symbolic link is specified on the command line or encountered during the traversal of a file hierarchy, the **chown** command shall change the owner ID (and group ID, if specified) of the symbolic link if the system supports this operation. The **chown** command shall not follow the symbolic link to any other part of the file hierarchy.
-R	Descends directories recursively, changing the ownership for each file. When a symbolic link is encountered and the link points to a directory, the ownership of that directory is changed but the directory is not further transversed. If the **-h**, **-H**, **-L** or **-P** flags are not also specified, when a symbolic link is encountered and the link points to a directory, the group ownership of that directory is changed but the directory is not traversed further.

Exit Status

This command returns the following exit values:

Item	Description
0	The command executed successfully and all requested changes were made.
>0	An error occurred.

Security

Access Control

This program should be installed as a normal user program in the Trusted Computing Base.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To change the owner of the file `program.c`:

```
chown jim program.c
```

The user access permissions for `program.c` now apply to `jim`. As the owner, `jim` can use the **chmod** command to permit or deny other users access to `program.c`.

2. To change the owner and group of all files in the directory `/tmp/src` to owner `john` and group `build`:

```
chown -R john:build /tmp/src
```

Files

Item	Description
/usr/bin/chown	The **chown** command
/etc/group	File that contains group IDs
/etc/passwd	File that contains user IDs

chpasswd Command

Purpose

Changes password for users.

Syntax

`chpasswd` [`-R` *load_module*] [`-e`] [`-f` *flags* | `-c`]

Description

The `chpasswd` command administers users' passwords. The root user can supply or change users' passwords specified through standard input. Each line of input must be of the following format.

```
username:password
```

Only root users can set passwords with this command.

By default, the `chpasswd` command sets the ADMCHG flag for the users. The `-f` option may be used with other valid flags to override the default. The `-c` option clears all password flags.

The password field can be cleartext or a value encrypted with the crypt algorithm. The `-e` option indicates that passwords are of encrypted format. Please note that all passwords in a batch must conform to the same format.

You can set LDAP user passwords in an ldap_auth environment by using the `chpasswd` command and specifying **-R** *LDAP*. However, when you specify the **-e** option for the encrypted format, the `chpasswd` command-crypted format and LDAP server-crypted format must match.

Flags

Item	Description
`-c`	Clears all password flags.
`-e`	Specifies that the passwords are of encrypted format.
`-f` *flags*	Specifies the comma separated list of password flags to set. Valid flag values are: ADMIN, ADMCHG, and/or NOCHECK. Refer to the `pwdadm` command documentation for details about these values.
`-R` *load_module*	Specifies the loadable I&A module used to change users' passwords.

Security

Access Control

Only root users should have execute (x) access to this command. The command should have the trusted computing base attribute.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To set passwords for users from the command line, type:

   ```
   chpasswd
   ```

   Followed by entering `username:password` pairs, one pair per line. Enter CTRL+D when finished.

   ```
   user1:passwd1
   user2:passwd2
   CTRL+D
   ```

2. To set passwords for users contained in a file named `mypwdfile`, type the following:

   ```
   cat mypwdfile | chpasswd
   ```

   Note that `mypwdfile` must contain `username:password` pairs; one pair per line. For example:

   ```
   user1:passwd1
   user2:passwd2
   ...
   ```

Files

Mode	File	Description
	`/etc/user/bin/chpasswd`	Location of the `chpasswd` command.
rw	`/etc/passwd`	
rw	`/etc/security/passwd`	
r	`/etc/security/user`	

chpath Command

Purpose

Changes the operational status of paths to an MultiPath I/O (MPIO) capable device, or changes an attribute associated with a path to an MPIO capable device.

Syntax

chpath **-l** *Name* **-s** *OpStatus* [**-p** *Parent*] [**-w** *Connection*] [**-i** *PathID*]

chpath **-l** *Name* **-p** *Parent* [**-w** *Connection*] [**-P**] **-a** *Attribute=Value* [**-a** *Attribute=Value* ...] [**-g**]

chpath **-l** *Name* **-i** *PathID* [**-P**] **-a** *Attribute=Value* [**-a** *Attribute=Value* ...]

chpath **-h**

Description

The **chpath** command either changes the operational status of paths to the specified device (the **-l** *Name* flag) or it changes one, or more, attributes associated with a specific path to the specified device. The required syntax is slightly different depending upon the change being made.

The first syntax shown above changes the operational status of one or more paths to a specific device. The set of paths to change is obtained by taking the set of paths which match the following criteria:

- The target device matches the specified device.
- The parent device matches the specified parent (**-p** *Parent*), if a parent is specified.
- The connection matches the specified connection (**-w** *Connection*), if a connection is specified.
- The path status is **PATH_AVAILABLE**.

The operational status of a path refers to the usage of the path as part of MPIO path selection. The value of **enable** indicates that the path is to be used while **disable** indicates that the path is not to be used. It should be noted that setting a path to **disable** impacts future I/O, not I/O already in progress. As such, a path can be disabled, but still have outstanding I/O until such time that all of the I/O that was already in progress completes. As such, if **-s disable** is specified for a path and I/O is outstanding on the path, this fact will be output.

Disabling a path affects path selection at the device driver level. The **path_status** of the path is not changed in the device configuration database. The **lspath** command must be used to see current operational status of a path.

The second syntax shown above changes one or more path specific attributes associated with a particular path to a particular device. Note that multiple attributes can be changed in a single invocation of the **chpath** command; but all of the attributes must be associated with a single path. In other words, you cannot change attributes across multiple paths in a single invocation of the **chpath** command. To change attributes across multiple paths, separate invocations of **chpath** are required; one for each of the paths that are to be changed.

Flags

Item	Description
-a *Attribute*=*Value*	Identifies the attribute to change as well as the new value for the attribute. The *Attribute* is the name of a path specific attribute. The *Value* is the value which is to replace the current value for the *Attribute*. More than one instance of the **-a** *Attribute*=*Value* can be specified in order to change more than one attribute.
-g	Forces the change path operation to take place on a locked device.
-h	Displays the command usage message.
-i *PathID*	Indicates the ID of the path that is affected by the change. This flag is used to uniquely identify a path.
-l *Name*	Specifies the logical device name of the target device for the path(s) affected by the change. This flag is required in all cases.
-p *Parent*	Indicates the logical device name of the parent device to use in qualifying the paths to be changed. This flag is required when changing attributes, but is optional when change operational status.

Item	Description
-P	Changes the path's characteristics permanently in the ODM object class without actually changing the path. The change takes affect on the path the next time the path is unconfigured and then configured (possibly on the next boot).
-w *Connection*	Indicates the connection information to use in qualifying the paths to be changed. This flag is optional when changing operational status. When changing attributes, it is optional if the device has only one path to the indicated parent. If there are multiple paths from the parent to the device, then this flag is required to identify the specific path being changed.
-s *OpStatus*	Indicates the operational status to which the indicated paths should be changed. The operational status of a path is maintained at the device driver level. It determines if the path will be considered when performing path selection. The allowable values for this flag are: **enable** Mark the operational status as **enabled** for MPIO path selection. A path with this status will be considered for use when performing path selection. Note that enabling a path is the only way to recover a path from a **failed** condition. **disable** Mark the operational status as **disabled** for MPIO path selection. A path with this status will not be considered for use when performing path selection. This flag is required when changing operational status. When used in conjunction with the **-a** *Attribute*=*Value* flag, a usage error is generated.

Security

Privilege Control: Only the **root** user and members of the **system** group have execute access to this command.

Auditing Events:

Event	Information
DEV_Change	The **chpath** command line.

Examples

1. To disable the paths between **scsi0** and the **hdisk1** disk device, enter:

   ```
   chpath -l hdisk1 -p scsi0 -s disable
   ```

   The system displays a message similar to one of the following:

   ```
   paths disabled
   ```

or

```
some paths disabled
```

The first message indicates that all PATH_AVAILABLE paths from **scsi0** to **hdisk1** have been successfully disabled. The second message indicates that only some of the PATH_AVAILABLE paths from **scsi0** to **hdisk1** have been successfully disabled.

Files

Item	Description
/usr/sbin/chpath	Contains the **chpath** command.

Related Information

The **lspath** command, **mkpath** command, **rmpath** command.

chprtsv Command

Purpose

Changes a print service configuration on a client or server machine.

Syntax

chprtsv **-c** | **-s** [**-d** | **-i**] [**-h"***HostName...***"** | **-H** *FileName*] [**-x"***HostName...***"** | **-X** *FileName*] [**-q"***QEntry***"** **-v** *DeviceName* **-a"***Attribute* **=***Value...***"** **-b"***Attribute* **=***Value...***"** | **-A** *FileName*]

Description

The **chprtsv** high-level command changes print service configuration on a client or server machine.

To change print service for a client, the **chprtsv** command does the following:

1. Disables the client spool queue with the **chque** and **chquedev** commands.
2. Changes the appropriate entries in the **/etc/qconfig** file with the **chque** and **chquedev** commands.
3. Enables the client spool queue with the **chque** and **chquedev** commands.

To change print service for a server, the **chprtsv** command does the following:

1. Calls the **ruser** low-level command to change remote users configured on the print server, if necessary.
2. Calls the **chque** and **chquedev** commands to change the print queues and entries in the **qconfig** file, if necessary.
3. Calls the SRC **refresh** command to restart the **lpd** and **qdaemon** servers.

If you want to change the attributes of a queue, you must specify the queue name and the attributes associated with the queue. If you want to change the attributes of the queue device, you must specify queue name, queue device name, and the attributes associated with the queue device.

The changes you make with the **chprtsv -i** command go into effect on the system database and on the current active system.

If you want the changes you make to go into effect at system startup time without affecting the current system, use the **chprtsv -d** command to change only TCP/IP and its associated network interfaces in the system database only.

Flags

Item	Description
-A *FileName*	Specifies the name of the file containing **qconfig** command-related entries.
-a "*Attribute* **=***Value*...**"**	Specifies a list of attributes with corresponding values to be used for updating the spooler's **qconfig** file or object class. The list should be enclosed in quotes. Valid attribute types follow: **acctfile (true/false)** Identifies the file used to save **print** accounting information. The default value of **false** suppresses accounting. If the named file does not exist, no accounting is done. **device** Identifies the symbolic name that refers to the device stanza. **discipline** Defines the queue-serving algorithm. The default, **fcfs**, means first come, first served. A value of **sjn** means shortest job next. **host** Specifies the name of the host from which to print. (The name of this host must be the same as the name specified by the *HostName* variable.) **l_statfilter** Translates long queue-status information from non-AIX format to AIX format. **s_statfilter** Translates short queue-status information from non-AIX format to AIX format. **up (true/false)** Defines the state of the queue. The default **true** indicates that it is running. A value of **false** indicates that it is not.

Item	Description
-b "*Attribute* **=***Value...***"**	Specifies a list of attributes with corresponding values for device stanza corresponding values to be used for updating the spooler's **qconfig** file or object class. The list should be enclosed in quotes. Valid attribute types follow: **access (write/both)** Specifies the type of access the backend has to the file specified by the **file** field. The **access** file has a value of **write** if the backend has write access to the file, or a value of **both** if the backend has both read and write access. This field is ignored if the file field has a value of **false**. **align (true/false)** Specifies whether the backend sends a form-feed control before starting the job if the printer has been idle. The default is **false**. **backend** Specifies the full path name of the backend, optionally followed by the flags and parameters to be passed to it. **feed** Specifies the number of separator pages to print when the device becomes idle, or takes a value of **never**, which indicates that the backend is not to print separator pages. **file** Identifies the special file where the output of the backend is to be redirected. The default values of **false** indicates no redirection. In this case, the backend opens the output file. **header (never/always/group)** Specifies whether a header page prints before each job or group of jobs. The default is a value of **never** which indicates no header page. To produce a header page before each job, specify a value of **always**. To produce a header before each group of jobs for the same user, specify a value of **group**. **trailer (never/always/group)** Specifies whether a trailer page prints after each job or group of jobs. The default value of **never** indicates no trailer page. To produce a trailer page after each job, specify a value of **always**. To produce a trailer after each group of jobs for the same user, specify a value of **group**.
-c	Specifies to the **chprtsv** command to reconfigure print service for a client machine.
-d	Specifies that changes be reflected in the system database only, so that they can take effect at the next system startup.
-H *FileName*	Specifies the name of a file containing a list of host names to be included.
-h"*HostName...***"**	Specifies a list of host names to be included on the current list of remote users who can use the print server. Note that the queuing system does not support multibyte host names.
-i	Specifies that the change be reflected not only in the database, but also in the current running system.
-q"*QEntry***"**	Specifies a **qconfig** file entry to be removed.

Item	Description
-s	Specifies that print service reconfiguration is to be performed for a server machine.
-v *DeviceName*	Specifies a list of device stanzas to be changed.
-X *FileName*	Specifies the name of a file containing a list of host names to be excluded.
-x"*HostName...***"**	Specifies a list of host names to be excluded on the current list of remote users who can use the print server.

Examples

To reconfigure a print server, specify that the changes will take effect at the next startup, specify the file containing the host names, and then exclude some of those hosts, enter:

```
chprtsv -s -d -H ruser.inc -x "host1,host2,host3"
```

Files

Item	Description
/etc/qconfig	Contains configuration information for the printer queuing system.
/etc/hosts.lpd	Specifies foreign hosts that can print on the local host.

chps Command

Purpose

Changes the attributes of a paging space.

Syntax

chps [**-t** *ps_helper*] [**-s** *LogicalPartitions* | **-d** *LogicalPartitions*] [**-f**] [**-c** *ChecksumSize*] [**-a** { **y** | **n** }] *PagingSpace*

Description

The **chps** command changes the attributes of a paging space. The *PagingSpace* parameter specifies the name of the paging space to be changed.

To change the size of a Network File System (NFS) paging space, the size of the file that resides on the server must first be changed and then the **swapon** command used to notify the client of the change in size of the paging space.

Note: There is a paging space limit of 64 GB per device.

If the **-t** flag is specified, the argument will be assumed to be a third-party helper executable. If the helper executable is present in the /sbin/helpers/pagespace path then it will be spawned passing all the arguments and with the **-c** flag to specify **chps** command. The /etc/swapspaces path will be modified accordingly if the helper executable returns zero. The helper executable must change the attributes. If the helper program doesn't exist in the /sbin/helpers/pagespace path, the **chps** command will display the usage error. The helper executable must exit with a 0 if successful and a non-zero if it fails.

You could also use the System Management Interface Tool (SMIT) **smit chps** fast path to run this command.

Note: The primary paging space is hardcoded in the boot record. Therefore, the primary paging space will always be activated when the system is restarted. The **chps** command is unable to deactivate the primary paging space.

Flags

Item	Description
-a	Specifies to use a paging space at the next system restart. **y** Specifies that the paging space is active at subsequent system restarts. **n** Specifies that the paging space is inactive at subsequent system restarts.
-d *LogicalPartitions*	Specifies the number of logical partitions to subtract.
-c *ChecksumSize*	Specifies the size of the checksum to use for the paging space, in bits. Valid options are 0 (checksums disabled), 8, 16 and 32. If **-c** is not specified, it will default to 0. The **chps** command with this option will fail on a swapped on paging space unless **-f** is used.
-f	Specifies that the checksum size set by **-c** will be used for the next swapon of the paging space. This option has no effect if **-c** is not used or if the paging space is not swapped on.
-s *LogicalPartitions*	Specifies the number of logical partitions to add.
-t	Specifies to use the helper program under /sbin/helpers/pagespace directory. ***ps_helper*** Name of the helper program for a third party device.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To change the size of the `myvg` paging space, enter:

   ```
   chps  -s 4 myvg
   ```

   This adds four logical partitions to the `myvg` paging space.
2. To define the PS02 paging space as configured and active at subsequent system restarts, enter:

   ```
   chps  -a y PS02
   ```

   This specifies that the PS02 paging space is to be active at subsequent system restarts.
3. To set the checksum size of the `myvg` paging space to 1 byte, enter:

   ```
   chps  -c 8 mypg
   ```

   This sets the myvg paging space checksum size to 8 bits, if it is not swapped on.
4. To change the size of the `myvg` paging space using helper program foo enter:

```
chps  -t foo –s4 myps
```

This adds four logical partitions to myps by calling the helper program foo.

Files

Item	Description
/etc/swapspaces	Specifies the paging space devices and their attributes.

chpv Command

Purpose

Changes the characteristics of a physical volume in a volume group.

Syntax

chpv [**-h** *hotspare*] [**-a** *allocation*] [**-v** *availability* [**-f**]][**-c**] [**-p** *mirrorpool*] [**-P**] [**-m** *mirrorpool*] *physicalvolume ...* [**-C** *hdiskname*]

Description

The **chpv** command changes the state of the physical volume in a volume group by setting allocation permission to either allow or not allow allocation and by setting the availability to either available or removed. This command can also be used to clear the boot record for the given physical volume. Characteristics for a physical volume remain in effect unless explicitly changed with the corresponding flag.

Note: To use this command, you must either have root user authority or be a member of the **system** group.

You can also use the System Management Interface Tool (SMIT) **smit chpv** fast path to run this command.

Flags

Item	Description
-a *allocation*	Sets the allocation permission for additional physical partitions on the physical volume specified by the *physicalvolume* parameter. Either allows (yes) the allocation of additional physical partitions on the physical volume, or prohibits (no) the allocation of additional physical partitions on the physical volume. The *allocation* variable can be either: **y** Allows the allocation of additional physical partitions on the physical volume. **n** Prohibits the allocation of additional physical partitions on the physical volume. The logical volumes that reside on the physical volume can still be accessed.
-c	Clears the boot record of the given physical volume.
-C *hdiskname*	Clears the owning volume manager from a disk. This flag is only valid when running as the root user. This command will fail to clear LVM as the owning volume manager if the disk is part of an imported LVM volume group.
-f	Forces a physical volume with open logical volumes to be unavailable (removed), unless the physical volume has active paging or a dump device.

Item	Description
-h *hotspare*	Sets the sparing characteristics of the physical volume so that the physical volume can be used as a hot spare. Also sets the allocation permission for physical partitions on the physical volume specified by the *physicalvolume* parameter. This flag has no meaning for non-mirrored logical volumes. The *hotspare* variable can be either: **y** Marks the disk as a hot spare disk within the volume group it belongs to and prohibits the allocation of physical partitions on the physical volume. The disk must not have any partitions allocated to logical volumes to be successfully marked as a hot spare disk. **n** Removes the disk from the hot spare pool for the volume group in which it resides and allows allocation of physical partitions on the physical volume.
-m *mirrorpool*	Changes the name of the mirror pool that is assigned to the specified disk to the value of the *mirrorpool* parameter.
-p *mirrorpool*	Assigns the physical volume to a mirror pool. The name of a mirror pool can be up to 15 characters in length. After mirror pools are enabled in a volume group, the volume group can no longer be imported into a version of AIX (before AIX Version 6.1) that does not support mirror pools.
-P	Removes the physical volume from the mirror pool that is being assigned. The physical volume can only be removed from the mirror pool if it has partitions that are allocated to a logical volume where mirror pools are enabled.
-v *availability*	Sets the availability of the physical volume. If you set the availability to closed, logical input and output to the physical volume are stopped. Access to physical volume data by the file system or the virtual memory manager is stopped, but you can continue to use the system management commands. The *availability* variable can be either: **a** Makes a physical volume available for logical input and output. **r** Makes a physical volume unavailable (removed) for logical input and output. If the physical volume is required in order to maintain a volume group quorum, an error occurs and the physical volume remains open.

Examples

1. To close physical volume hdisk3, enter:

   ```
   chpv  -v r hdisk3
   ```

   The physical volume is closed to logical input and output until the **-v a** flag is used.
2. To open physical volume hdisk3, enter:

   ```
   chpv  -v a hdisk3
   ```

   The physical volume is now open for logical input and output.
3. To stop the allocation of physical partitions to physical volume hdisk3, enter:

   ```
   chpv  -a n hdisk3
   ```

   No physical partitions can be allocated until the **-a y** flag is used.
4. To clear the boot record of a physical volume hdisk3, enter:

```
chpv -c hdisk3
```

Files

Item	Description
/usr/sbin	Directory where the **chpv** command resides.
/tmp	Directory where temporary files are stored while the command is running.

chque Command

Purpose

Changes the queue name.

Syntax

chque -q *Name* [**-a '***Attribute***=***Value***'** ...]

Description

The **chque** command changes the queue name by changing the stanza in the **qconfig** file specified by the **-q** flag. Within that stanza, each attribute that matches one of the *Attribute* **=** *Value* pairs given on the command line will be replaced by the one on the command line. If no match is found, the *Attribute* **=** *Value* pair is added to the end of the stanza. The device attribute cannot be changed.

You could also use the System Management Interface Tool (SMIT) **smit chque** fast path to run this command.

Recommendation: To edit the **/etc/qconfig** file, use the **chque**, **mkque**, **rmque**, **chquedev**, **mkquedev**, and **rmquedev** commands or SMIT. Further, it is recommended to run these commands during slow or off-peak time.

If manual editing of the **/etc/qconfig** file is necessary, you can first issue the **enq -G** command to bring the queuing system and the **qdaemon** to a halt after all jobs are processed. Then you can edit the **/etc/qconfig** file and restart the **qdaemon** with the new configuration.

Flags

Item	Description
-a '*Attribute* **=** *Value*'	Specifies the **'***Attribute* **=** *Value***'** to be added or replaced by the one entered on the command line. For a list of valid attributes, refer to the **/etc/qconfig** file.
-q *Name*	Specifies the current *Name* of the queue and of the stanza in the **qconfig** file that is to be changed.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To change the name of the host to `fred` for queue `lp0`, enter:

```
chque  -qlp0  -a 'host = fred'
```

Files

Item	Description
/usr/bin/chque	Contains the **chque** command.
/etc/qconfig	Contains the configuration file.

chquedev Command

Purpose

Changes the printer or plotter queue device names.

Syntax

chquedev -q*Name* **-d***Name* [**-a'***Attribute* **=** *Value*'...]

Description

The **chquedev** command changes the printer or plotter queue device names by changing the device stanza in the **qconfig** file specified by the **-d**, and **-q** flags. Within that stanza, each attribute that matches one of the **'***Attribute* **=** *Value*' flags given on the command line is replaced by the one entered on the command line. If no match is found, **'***Attribute* **=** *Value***'** is added to the end of the stanza.

You could also use the System Management Interface Tool (SMIT) **smit chquedev** fast path to run this command.

Recommendation: To edit the **/etc/qconfig** file, use the **chque**, **mkque**, **rmque**, **chquedev**, **mkquedev**, and **rmquedev** commands or SMIT. Further, it is recommended to run these commands during slow or off-peak time.

If manual editing of the **/etc/qconfig** file is necessary, you can first issue the **enq -G** command to bring the queuing system and the **qdaemon** to a halt after all jobs are processed. Then you can edit the **/etc/qconfig** file and restart the **qdaemon** with the new configuration.

Flags

Item	Description
-a '*Attribute* **=** *Value***'**	Specifies the stanza lines to change or add. For a list of valid attributes, see the **qconfig** file.
-d *Name*	Specifies the device *Name* in the queue to be changed.
-q *Name*	Specifies the queue *Name* in which to change the device stanza.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To change the ps device stanza on the lp0 queue to contain the line 'backend = piobe -x -y', enter:

```
chquedev -qlp0 -d ps -a 'backend = piobe -x -y'
```

Note: The **-x** flag and the **-y** flag in this example are flags for the **piobe** command.

Files

Item	Description
/usr/bin/chquedev	Contains the chquedev command.
/etc/qconfig	Contains the configuration file.

chrepos Command

Purpose

Replaces a disk, used as the repository disk by the cluster or site, with another disk.

Syntax

chrepos [-n *cluster_name*] [-r [+*New_reposDiskName* | +*New_reposDiskName*,-*Old_reposDiskName*]]

Description

The **chrepos** command allows the disk currently used by the cluster or site as the repository disk to be replaced with a different disk.

In a multisite environment, the **chrepos** command can only replace the repository disk for the local site. The **chrepos** cannot be used to replace a repository disk at the remote site.

Flags

Item	Description
-n *cluster_name*	Specifies the name of cluster to be processed.
-r +*New_reposDiskName*	Specifies the name of new repository disk to be used for replacing the existing repository disk. This syntax can only be used to clean up and to complete a previously failed replace operation that used the **-r** +New_reposDiskName,-Old_reposDiskName syntax.
-r +*New_reposDiskName*,-*Old_reposDiskName*	Specifies the name of new repository disk to be added and the name of old repository disk to be removed.

Examples

1. To replace the **hdiskY** disk with the **hdiskX** disk in a cluster named **cl1**:

   ```
   chrepos -n cl1 -r +hdiskX,-hdiskY
   ```

2. To replace the existing repository disk with the **hdiskX** disk in the cluster called **cl1**:

   ```
   chrepos -n cl1 -r +hdiskX
   ```

chresponse Command

Purpose

Adds or deletes the actions of a response or renames a response.

Syntax

To add an action to a response:

`chresponse -a -n` *action* [`-d` *days_of_week*[`,`*days_of_week...*]] [`-t` *time_of_day*[`,`*time_of_day...*]] [`-s` *action_script*] [`-r` *return_code*] [**-b** | [`-e a | A | b | e | r`]] [`-o`] [`-E` *env_var=value*[`,`*env_var=value...*]] [`-u`] [`-h`] [`-TV`] *response*[`:`*node_name*]

To delete an action from a response:

`chresponse -p -n` *action* [`-h`] [`-TV`] *response*[`:`*node_name*]

To rename a response:

`chresponse -c` *new_response* [`-h`] [`-TV`] *response*[`:`*node_name*]

To unlock or lock a response:

`chresponse` {`-U` | `-L`} [`-h`] [`-TV`] *response*[`:`*node_name*]

Description

The `chresponse` command adds an action to a response or deletes an action from a response. Actions define commands to be run when the response is used with a condition and the condition occurs. The `chresponse` command can also be used to rename a response.

If a particular response is needed for system software to work properly, it may be locked. A locked response cannot be modified or removed until it is unlocked. If the response you specify on the `chresponse` command is locked, it will not be modified; instead an error will be generated informing you that the response is locked. To unlock a response, you can use the `-U` flag. However, since a response is typically locked because it is essential for system software to work properly, you should exercise caution before unlocking it. To lock a response so it cannot be modified, use the `-L` flag.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node.

Flags

-a
Adds the action specification to *response*.

-b
Specifies that the response, and all actions to be defined in this response, support event batching. For event batching, multiple events can be batched or grouped together and passed to a response. The actions of the response are directed to a file that contains the details for the batched events. A response that supports event batching can only be used for conditions that specify the events are to be batched.

The **-b** flag cannot be specified with the **-e** flag.

-p
Deletes *action* from *response*.

-c *new_response*
Specifies a new name to assign to the response. The new name must not already exist. The new name replaces the current name. The *new_response* name is a character string that identifies the response. If the name contains spaces, it must be enclosed in quotation marks. A name cannot consist of all spaces, be null, or contain embedded double quotation marks.

-n *action*
Specifies the name of the action. When the `-a` flag is used, this is the name of the action being defined. When the `-p` flag is used, this is the name of the action to be deleted. Action names must be unique within a response. Only one action can be defined at a time.

-d *days_of_week*[,*days_of_week...*]

Specifies the days of the week when the action being defined can be run. *days_of_week* and *time_of_day* together define the interval when the action can be run.

Enter the numbers of the days separated by a plus sign (+) or as a range of days separated by a hyphen (-). More than one *days_of_week* parameter can be specified, but the parameters must be separated by a comma (`,`). The number of *days_of_week* parameters specified must match the number of *time_of_day* parameters specified. The default is all days. If no value is specified but a comma is entered, the default value is used. The values for each day follow:

1
Sunday

2
Monday

3
Tuesday

4
Wednesday

5
Thursday

6
Friday

7
Saturday

-t *time_of_day*[,*time_of_day...*]
Specifies the time range when *action* can be run, consisting of the start time followed by the end time, separated by a hyphen. *days_of_week* and *time_of_day* together define the interval when the action can be run.

The time is in 24–hour format (HHMM), where the first two digits represent the hour and the last two digits represent the minutes. The start time must be less than the end time because the time is specified by day of the week. More than one *time_of_day* parameter can be specified, but the parameters must be separated by a comma (`,`). The number of *days_of_week* parameters specified must match the number of *time_of_day* parameters specified. The default is `0000-2400`. If no value is specified but a comma is entered, the default value is used.

-s *action_script*
Specifies the fully-qualified path for the script or command to run for the action being defined. See the `displayevent`, `logevent`, `notifyevent`, and `wallevent` commands for descriptions of predefined response scripts that are provided with the application.

-r *return_code*
Specifies the expected return code for *action_script*. The actual return code of *action_script* is compared to the expected return code. A message is written to the audit log indicating whether they match. If the `-r` flag is not specified, the actual return code is written to the audit log, and no comparison is performed.

-e a | A | b | e | r
Specifies the type of event that causes the action being defined to run:

a
Specifies an event. This is the default value.

A
Specifies any type of event (event, error event, or rearm event).

b
Specifies both an event and a rearm event.

e
Specifies an error event.

r
Specifies a rearm event.

More than one event type can be specified, for example: `-e ae`.

The **-e** flag cannot be specified with the **-b** flag.

-o
Directs all standard output from *action_script* to the audit log. The default is not to keep standard output. Standard error is always directed to the audit log.

-E *env_var*=*value*[,*env_var*=*value*...]
Specifies any environment variables to be set before *action_script* is run. If multiple *env_var*=*value* variables are specified, they must be separated by commas.

-u
Specifies that the action is to be run when a monitored resource becomes undefined.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization use only.

-V
Writes the command's verbose messages to standard output.

-U
Unlocks a response so it can be modified or removed. If a response is locked, this is typically because it is essential for system software to work properly. For this reason, you should exercise caution before unlocking it. When unlocking a response using the **-U** flag, no other operation can be preformed by this command.

-L
Locks a response so it cannot be modified or removed. When locking a response using the `-L` flag, no other operation can be performed by this command.

Parameters

response
Specifies the name of the response to be changed.

node_name
Specifies the node where the response is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

Security

The user of the `chresponse` command needs write permission to the `IBM.EventResponse` resource class on the node where the response is defined. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. In this example, the action named "E-mail root" cannot be the only action. To delete "E-mail root" from the response named "E-mail root anytime", run this command:

```
chresponse -p -n "E-mail root" "E-mail root anytime"
```

2. In this example, the action named "E-mail root" will be used Monday through Friday from 8 AM to 6 PM, will use the command `/opt/rsct/bin/notifyevent root`, will save standard output in the audit log, and will expect return code 5 from the action. To add "E-mail root" to the response named "E-mail root anytime", run this command:

```
chresponse  -a  -n "E-mail root" -d 2-6 -t 0800-1800 \
-s "/opt/rsct/bin/notifyevent root" -o  -r 5  \
"E-mail root anytime"
```

3. To rename the response "E-mail root anytime" to "E-mail root and admin anytime", run this command:

```
chresponse  -c "E-mail root and admin anytime" "E-mail root anytime"
```

These examples apply to management domains:

1. To delete the action named "E-mail root" from the response named "E-mail root anytime" that is defined on the management server, run this command on the management server:

```
chresponse -p -n "E-mail root" "E-mail root anytime"
```

2. In this example, the action named "E-mail root" will be used Monday through Friday from 8 AM to 6 PM, will use the command `/opt/rsct/bin/notifyevent root`, will save standard output in the audit log, and will expect return code 5 from the action. To add "E-mail root" to the response "E-mail root anytime" that is defined on the management server, run this command on the management server:

```
chresponse  -a  -n "E-mail root" -d 2-6 -t 0800-1800 \
-s "/opt/rsct/bin/notifyevent root" -o  -r 5  \
"E-mail root anytime"
```

3. To delete the action named "E-mail root" from the response named "E-mail root anytime" that is defined on the managed node nodeB, run this command on the management server:

```
chresponse -p -n "E-mail root" "E-mail root anytime":nodeB
```

These examples apply to peer domains:

1. In this example, the action named "E-mail root" will be used Monday through Friday from 8 AM to 6 PM, will use the command `/opt/rsct/bin/notifyevent root`, will save standard output in the audit log, and will expect return code 5 from the action. To add "E-mail root" to the response "E-mail root anytime" that is defined on node nodeA in the domain, run this command on any node in the domain:

```
chresponse  -a  -n "E-mail root" -d 2-6 -t 0800-1800 \
-s "/opt/rsct/bin/notifyevent root" -o  -r 5  \
"E-mail root anytime":nodeA
```

2. To delete the action named "E-mail root" from the response named "E-mail root anytime" that is defined on node nodeA in the domain, run this command on any node in the domain:

```
chresponse -p -n "E-mail root" "E-mail root anytime":nodeA
```

Location

/opt/rsct/bin/chresponse

chrmcacl Command

Purpose

Updates the resource monitoring and control (RMC) ACL file.

Syntax

`chrmcacl [ -a | -d | -r | -h ]`

Description

This command is used to update the RMC ACL file (`/var/ct/cfg/ctrmc.acls`). If this file does not exist, `chrmcacl` copies the default ACL file from `/opt/rsct/cfg/ctrmc.acls` to `/var/ct/cfg/ctrmc.acls`. This command reads update information from standard input. This input must be in ACL file format, so it must consist of one or more stanzas, in which each stanza begins with a stanza name that is followed by zero or more stanza lines. A stanza is terminated by a blank line, a comment line, another stanza, or end-of-file. See the description of the RMC ACL file in the *Administering RSCT* for details.

With no flags specified, `chrmcacl` does whole stanza addition, replacement, or deletion. If the input stanza does not exist in the ACL file, it is added. If the input stanza has a match in the ACL file, the input stanza replaces the existing ACL file stanza. If the input stanza contains no stanza lines and has a match in the ACL file, the existing ACL file stanza is removed.

If the `-a`, `-r`, or `-d` flag is specified, `chrmcacl` does individual stanza line addition, replacement, or deletion. Stanza lines are matched based on the user identifier and object type tokens, in the stanza line, within matching stanzas. Matches must be exact; in other words, there is no wildcard matching.

When the `-a` flag is used, the permissions specified in the input stanza line are added to the permissions from the matching stanza line in the ACL file. If this results in an effective change in permissions, the new permissions are updated in the ACL file. If there is no matching stanza line in the ACL file, the input stanza line is added to the matching stanza in the ACL file.

When the `-r` flag is used, the input stanza line unconditionally replaces the matching stanza line in the ACL file. If there is no matching stanza line in the ACL file, the input stanza line is added to the matching stanza in the ACL file. For the `-a` and `-r` flags, if the input stanza has no match in the ACL file, the complete input stanza is added to the ACL file.

When the `-d` flag is used, any matching stanza lines in the ACL file are deleted. If, as a result, the matching stanza in the ACL file has no stanza lines, the stanza is removed from the ACL file.

As a by-product of this command, the stanza lines within each stanza are ordered from the most specific user identifiers and object types to less specific user identifiers and object types.

The `chrmcacl` command employs file locking, which is used by other RSCT components, to serialize updates and prevent file corruption. Therefore, it is recommended that you use this command to update the ACL file, rather than by modifying the file directly.

When the ACL file is updated, the previous version is first saved as `/var/ct/cfg/ctrmc.acls.orig`. If there are no effective changes or if there are any errors, the ACL file is not updated.

Changes to the ACL file take effect the next time the RMC subsystem is started. To get the ACL file changes to take effect immediately, run this command:

```
refresh -s ctrmc
```

Flags

-a
Adds the permissions of the input stanza lines to the matching stanza lines within the matching ACL file stanzas.

-d
Deletes the matching stanza lines within the matching ACL file stanzas.

-r
Replaces the matching stanza lines within the matching ACL file stanzas with the input stanza lines.

-h
Writes the command usage statement to standard error.

Files

/opt/rsct/cfg/ctrmc.acls
Default location of the `ctrmc.acls` file

/var/ct/cfg/ctrmc.acls
Location of the modifiable `ctrmc.acls` file

/var/ct/cfg/ctrmc.acls.orig
Location of the previous version of the modifiable `ctrmc.acls` file

Standard input

This command reads update information from standard input.

Standard error

Error messages are written to standard error.

When the `-h` flag is specified, this command usage statement is written to standard error.

Exit status

0
The command has run successfully.

1
The command was not successful.

Security

Privilege control: only the `root` user must have execute (x) access to this command.

Implementation specifics

This command is part of the `rsct.core` fileset for AIX and `rsct.core-3.1.0.0-0.`*platform*`.rpm` package for Linux®, Solaris, and Windows, where *platform* is `i386`, `ppc`, `ppc64`, `s390`, or `x86_64`.

Location

/opt/rsct/install/bin/chrmcacl

Examples

1. If the `/var/ct/cfg/ctrmc.acls` file already contains the `IBM.Sensor` stanza, but not the `OTHER` stanza, and given the following input to `chrmcacl` (with no flags specified):

```
IBM.Sensor
    joe@Host1.CoX.com   *    rw
    Host1.CoX.com       *    r

OTHER
    Host1.CoX.com       C    r
```

the IBM.Sensor stanza is replaced by the input stanza and the OTHER stanza is added to the file upon successful completion of the command.

2. With the /var/ct/cfg/ctrmc.acls file that is a result of example 1 and given the following input to chrmcacl (with no flags specified):

```
IBM.Sensor

OTHER
    Host1.CoX.com       *    r
```

the IBM.Sensor stanza is deleted and the OTHER stanza is replaced by the input stanza upon successful completion of the command.

3. With the /var/ct/cfg/ctrmc.acls file that is a result of example 2 and given the following input to chrmcacl (with the -a flag specified):

```
OTHER
    Host1.CoX.com       *    w
```

the OTHER stanza in the file is:

```
OTHER
    Host1.CoX.com       *    rw
```

upon successful completion of the command.

4. With the /var/ct/cfg/ctrmc.acls file that is a result of example 3 and given the same input to chrmcacl as in example 3 (with the -r flag specified), the OTHER stanza in the file is:

```
OTHER
    Host1.CoX.com       *    w
```

upon successful completion of the command.

5. Given the following stanza in the /var/ct/cfg/ctrmc.acls file:

```
IBM.Sensor
    joe@Host1.CoX.com   C    rw
    joe@Host1.CoX.com   R    r
    Host1.CoX.com       *    r
```

and the following input to chrmcacl (with the -d flag specified):

```
IBM.Sensor
    joe@Host1.CoX.com   R    r
```

the IBM.Sensor stanza in the file is:

```
IBM.Sensor
    joe@Host1.CoX.com   C       rw
        Host1.CoX.com   *       r
```

upon successful completion of the command.

chsignpolicy Command

Purpose

Changes the value for the digital signature policy option for trusted installation and update of the AIX operating system.

Syntax

```
chsignpolicy [-R]
chsignpolicy [-p]
chsignpolicy [-s low|medium|high|none]
```

Description

Starting with release IBM AIX 7.2 with Technology Level 4, the software packages of the AIX operating system, which are delivered in the `installp` format, are digitally signed. The installation process verifies the digital signatures of the software package and takes action based on the digital signature policy. You can use the **chsignpolicy** command to modify the digital signature policy. By setting an appropriate value of the `digital signature policy` option, you can ensure that the software packages that are being used for installation or upgrade, have not been altered after the software packages are shipped by IBM.

Notes:

- The Digital Signature Policy settings are not enforced during an operating system installation. You must first complete the operating system installation to ensure that the system is running after installation.
- You must successfully install IBM AIX 7.2 with Technology Level 4 or later on your system, for trusted installation of software package to work.

Flags

-R
Lists the values of the Digital Signature Policy.

-p
Prints the settings of the current Digital Signature Policy.

-s

Sets the Digital Signature Policy for future installation operations. You must specify one of the following values for this flag:

none
Indicates that the AIX operating system does not check the digital signatures of the software packages that are being installed or updated. This value is the default setting.

low
Indicates that the AIX operating system checks the signatures of the software packages that are being installed or updated. If the signature verification fails, the installation process displays a warning message, but the installation continues.

The warning message that is similar to the following example is displayed:

```
INFO: Package <full path to package name> failed signature verification.
```

medium
Indicates that the AIX operating system checks the digital signatures of the software packages that are being installed or updated. If the signature verification fails, the installation process prompts you whether you want to continue the installation. You must confirm the installation for each fileset that failed signature verification. Otherwise, the software package is not installed successfully.

The warning message is similar to the following example:

```
WARNING: Package <full path to package name> failed signature verification. Continue?
(y/n)
```

high
Indicates that the AIX operating system checks the digital signatures of the software packages that are being installed or updated. If the digital signature verification fails, the installation of the software package fails.

The failure message is similar to the following example:

```
FAILURE: Package <full path to package name> failed signature verification.
```

Examples

- To set the Digital Signature Policy to `low`, run the following command:

```
chsignpolicy -s low
```

If an unsigned software package is installed, the following message is displayed in the installation log, but the software package installation will be successful:

```
INFO: Package <full path to package name> failed signature verification.
```

- To display the current Digital Signature Policy, run the following command:

```
chsignpolicy -p
```

An output similar to the following example is displayed:

```
#signpolicy
 low
```

- To display available Digital Signature Policies, run the following command:

```
chsignpolicy -R
```

An output similar to the following example is displayed:

```
none
low
medium
high
```

Security

Only the root user can run this command.

Files

/usr/sbin/chsignpolicy

chrole Command

Purpose

Changes role attributes.

Syntax

chrole [**-R** *load_module*] *Attribute*=*Value ... Name*

Description

The **chrole** command changes attributes for the role identified by the *Name* parameter. The role name must already exist. To change an attribute, specify the attribute name and the new value with the *Attribute*=*Value* parameter.

If you specify a single incorrect attribute or attribute value with the **chrole** command, the command does not change any attribute.

You can also use the System Management Interface Tool (SMIT) **smit chrole** fast path to run this command.

If the system is configured to use multiple domains for the role database, role modification is performed according to the order specified by the **secorder** attribute of the roles database stanza in the **/etc/nscontrol.conf** file. Only the first matching role is modified. Duplicate roles from the remaining domains are not modified. Use the **-R** flag to modify the role from a specific domain.

When the system is operating in enhanced Role Based Access Control (RBAC) mode, modifications made to the role database are not used for security considerations until the database is sent to the kernel security tables through the **setkst** command.

Flags

Item	Description
-R *load_module*	Specifies the loadable module to use for the role modification.

Attributes

If you have the proper authority, you can set the following user attributes:

Item	Description
auditclasses	List of roles's audit classes. The *Value* parameter is a list of comma-separated classes or a value of ALL to indicate all audit classes.
auth_mode	Specifies the authentication that is required to assume the role when the **swrole** command is used. You can specify the following values: **NONE** No authentication is required. **INVOKER** The invoker of the **swrole** command is required to enter their own password to assume the role. The **INVOKER** value is the default value.
authorizations	List of additional authorizations required for this role beyond those defined by the roles in the **rolelist** attribute. The *Value* parameter is a list of authorization names, separated by commas.
dfltmsg	Contains the default role-description text to use if message catalogs are not in use.
groups	List of groups to which a user should belong, in order to effectively use this role. This attribute is for information only and does not automatically make the user a member of the list of groups. The *Value* parameter is a list of group names, separated by commas.
hostsenabledrole	Specifies the hosts which can download the role definition to the Kernel Role table by using the **setkst** command. This attribute must be used in a networked environment where the role attributes are shared by multiple hosts.
hostsdisabledrole	Specifies the hosts which cannot download the role definition to the Kernel Role table using the **setkst** command. This attribute is intended to be used in a networked environment where the role attributes are shared by multiple hosts.

Item	Description
id	Specifies the unique numeric ID for the role. You must specify the **id** attribute. **Attention**: Do not modify the attribute value after the role is assigned to a user.
msgcat	Contains the file name of the message catalog that holds the one-line descriptions of system roles. The *Value* parameter is a character string.
msgnum	Contains the index into a message catalog for a description of the role. The *Value* parameter is an integer.
msgset	Contains the message set that includes the role description in the message catalog.
rolelist	Lists the roles implied by this role. The *Value* parameter is a list of role names, separated by commas. When specified with the **-R** flag, the roles stanza in the `nscontrol.conf` file is overridden by the **-R** flag.
screens	Lists the SMIT screen identifiers allowing roles to be mapped to various SMIT screens. The *Value* parameter is a list of SMIT screen identifiers, separated by commas.
visibility	Specifies the role's visibility status to the system. The *Value* parameter is an integer. Possible values are: **1** The role is enabled, displayed, and selectable. Authorizations contained in this role are applied to the user. If the attribute does not exist or has no value, the default value is 1. **0** The role is enabled and displayed as existing, but *not* selectable through a visual interface. Authorizations contained in this role are applied to the user. **-1** The role is disabled. Authorizations contained in this role are *not* applied to the user.

Security

The **chrole** command is a privileged command. You must assume a role that has the following authorization to run the command successfully.

Item	Description
aix.security.role.change	Required to run the command.

Auditing Events

Event	Information
ROLE_Change	role, attribute

Files Accessed

Mode	File
rw	**/etc/security/roles**
r	**/etc/security/user.roles**

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To change the authorizations of the `ManagePasswds` role to `aix.security.passwd`, use the following command:

```
chrole authorizations=aix.security.passwd ManagePasswds
```

2. To change the authorizations of the `ManagePasswds` role in LDAP to `aix.security.passwd`, use the following command:

```
chrole -R LDAP authorizations=aix.security.passwd ManagePasswds
```

Files

Item	Description
/etc/security/roles	Contains the attributes of roles.
/etc/security/user.roles	Contains the role attribute of users.

chroot Command

Purpose

Changes the root directory of a command.

Syntax

chroot *Directory Command*

Description

> **Attention:** If special files in the new root directory have different major and minor device numbers than the real root directory, it is possible to overwrite the file system.

The **chroot** command can be used only by a user operating with root user authority. If you have root user authority, the **chroot** command changes the root directory to the directory specified by the *Directory* parameter when performing the *Command*. The first / (slash) in any path name changes to *Directory* for the specified *Command* and any of its children.

The *Directory* path name is always relative to the current root. Even if the **chroot** command is in effect, the *Directory* path name is relative to the current root of the running process.

A majority of programs may not operate properly after the **chroot** command runs. For example, the commands that use the shared libraries are unsuccessful if the shared libraries are not in the new root file system. The most commonly used shared library is the **/usr/ccs/lib/libc.a** library.

The **ls -l** command is unsuccessful in giving user and group names if the current root location makes the **/etc/passwd** file beyond reach. In addition, utilities that depend on localized files (**/usr/lib/nls/***) may also be unsuccessful if these files are not in the new root file system. It is your responsibility to ensure that all vital data files are present in the new root file system and that the path names accessing such files are changed as necessary.

Note: Ensure that the `/usr/sbin/execerror` command is available on the new root file system so that descriptive error messages are returned in the event of a `chroot` failure. Otherwise, if there is an error, `chroot` returns `Killed` and nothing more.

Parameters

Item	Description
Command	Specifies a command to run with the **chroot** command.
Directory	Specifies the new root directory.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

> **Attention:** The commands in the following examples may depend on shared libraries. Ensure that the shared libraries are in the new root file system before you run the **chroot** command.

1. To run the **pwd** command with the **/usr/bin** directory as the root file system, enter:

```
mkdir /usr/bin/lib
cp /usr/ccs/lib/libc.a /usr/bin/lib
cp /usr/lib/libcrypt.a /usr/bin/lib
chroot /usr/bin pwd
```

2. To run a Korn shell subshell with another file system as the root file system, enter:

```
chroot /var/tmp /usr/bin/ksh
```

   This makes the directory name / (slash) refer to the `/var/tmp` for the duration of the `/usr/bin/ksh` command. It also makes the original root file system inaccessible. The file system on the `/var/tmp` file must contain the standard directories of a root file system. In particular, the shell looks for commands in the `/bin` and `/usr/bin` files on the `/var/tmp` file system.

   Running the `/usr/bin/ksh` command creates a subshell that runs as a separate process from your original shell. Press the END OF FILE (Ctrl-d) key sequence to end the subshell and go back to where you were in the original shell. This restores the environment of the original shell, including the meanings of the . (current directory) and the / (root directory).

3. To create a file relative to the original root, not the new one, enter:

```
chroot directory Command > file
```

Files

Item	Description
/etc/passwd	Specifies file that contains basic user attributes.
/usr/ccs/lib/libc.a	Specifies the standard I/O library and the standard C library.
/usr/ccs/lib/libcurses.a	Specifies the curses library.
/usr/lib/liblvm.a	Specifies the LVM (Logical Volume Manager) library.
/usr/ccs/lib/libm.a	Specifies the math library.
/usr/lib/libodm.a	Specifies the ODM (Object Data Manager) library.
/usr/sbin/chroot	Contains the **chroot** command.

chrsrc Command

Purpose

Changes the persistent attribute values of a resource or a resource class.

Syntax

To change the persistent attribute values of a *resource,* using data that is...

- entered on the command line:

  `chrsrc -s "`*selection_string*`" [ -a | -N { `*node_file* `| "-" } ] [-v] [-h] [-TV]` *resource_class attr=value...*

  `chrsrc -r [-v] [-h] [-TV]` *resource_handle attr=value...*

- predefined in an input file:

  `chrsrc -f` *resource_data_input_file* `-s "`*selection_string*`" [-a | -N {` *node_file* `| "-" } ] [-v] [-h] [-TV]` *resource_class*

  `chrsrc -f` *resource_data_input_file* `-r [-v] [-h] [-TV]` *resource_handle*

To change the persistent attribute values of a *resource class,* using data that is...

- entered on the command line:

  `chrsrc { -c | -C` *domain_name...* `} [-v [-a] [-h] [-TV]` *resource_class attr=value...*

- predefined in an input file:

  `chrsrc -f` *resource_data_input_file* `{ -c | -C` *domain_name...* `} [-v] [-a] [-h] [-TV]` *resource_class*

Description

The `chrsrc` command changes the persistent attribute values of a resource or a resource class. By default, this command changes the persistent attribute values of a *resource.* Use the `-r` flag to change only the persistent attribute values of the resource that is linked with *resource_handle*. Use the `-s` flag to change the persistent attribute values of all of the resources that match *selection_string*. To change the persistent attributes of a *resource class,* use the `-c` flag.

Instead of specifying multiple node names in *selection_string*, you can use the `-N` *node_file* flag to indicate that the node names are in a file. Use `-N "-"` to read the node names from standard input.

The `chrsrc` command cannot change dynamic attributes, nor can it change persistent attributes that are designated as `read_only`. To verify that all of the attribute names that are specified on the command line or in *resource_data_input_file* are defined as persistent attributes and are *not* designated as `read_only`, use the `-v` flag. When the `chrsrc` command is run with the `-v` flag, the specified attributes are not changed, but are instead merely verified to be persistent and not designated as `read_only`. Once you run `chrsrc -v` to verify that the attributes that are specified on the command line or in *resource_data_input_file* are valid, you can issue the `chrsrc` command without the `-v` flag to actually change the attribute values. Note, however, that just because an attribute "passes" when `chrsrc -v` is run does not ensure that the attribute can be changed. The underlying resource manager that controls the specified resource determines which attributes can be changed by the `chrsrc` command. After `chrsrc` is run without the `-v` flag, an error message will indicate whether any specified attribute could not be changed.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM `nodegrp` command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference.*

Flags

-a
Specifies that this command applies to all of the nodes in the cluster. The CT_MANAGEMENT_SCOPE environment variable determines the scope of the cluster. If CT_MANAGEMENT_SCOPE is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and CT_MANAGEMENT_SCOPE is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-c
Changes the persistent attribute values for *resource_class*.

-C *domain_name...*
Changes the class attributes of a globalized resource class on one or more RSCT peer domains that are defined on the management server. Globalized classes are used in peer domains and management domains for resource classes that contain information about the domain.

To change class attributes of a globalized resource class on all peer domains defined on the management server, use the `-c` flag with the `-a` flag instead of `-C`.

-f *resource_data_input_file*
Specifies the name of the file that contains resource attribute information.

-N { *node_file* | "-" }
Specifies that node names are read from a file or from standard input. Use `-N` *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (#) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use `-N "-"` to read the node names from standard input.

The CT_MANAGEMENT_SCOPE environment variable determines the scope of the cluster. If CT_MANAGEMENT_SCOPE is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and CT_MANAGEMENT_SCOPE is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-r
Changes the persistent attribute values for the specific resource that matches *resource_handle*.

-s "*selection_string*"
Changes the persistent attribute values for all of the resources that match *selection_string*. *selection_string* must be enclosed within either double or single quotation marks. If *selection_string* contains double quotation marks, enclose it in single quotation marks, for example:

```
-s 'Name == "testing"'
```

```
-s 'Name ?= "test"'
```

Only persistent attributes can be listed in a selection string. For information on how to specify selection strings, see the *RSCT: Administration Guide*.

-v
Verifies that all of the attribute names specified on the command line or in the input file are defined as persistent attributes and are *not* designated as `read_only`. The `chrsrc` command does *not* change any persistent attribute values when you use this flag.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

attr=value...
Specifies one or more pairs of attributes and their associated values. *attr* is any defined persistent attribute name. Use the `lsrsrcdef` command to display a list of the defined persistent attributes and their datatypes for the specified resource. The value specified must be the appropriate datatype for the associated attribute. For example, if `NodeNumber` is defined as a `Uint32` datatype, enter a positive numeric value.

Do not specify this parameter if you run `chrsrc` with the `-f` flag.

resource_class
Specifies a resource class name. Use the `lsrsrcdef` command to display a list of defined resource class names.

resource_handle
Specifies a resource handle that is linked with the resource that you want to change. Use the `lsrsrc` command to display a list of valid resource handles. The resource handle must be enclosed within double quotation marks, for example:

```
"0x4017 0x0001 0x00000000 0x0069684c 0x0d4715b0 0xe9635f69"
```

Security

The user needs write permission for the *resource_class* specified in `chrsrc` to run `chrsrc`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for information about the ACL file and how to modify it.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
No resources were found that match the selection string.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To change the `Int32`, `Uint32` and `SD` persistent resource attributes in resource class `IBM.Foo` for the resources that have a `Name` equal to `c175n05`, enter:

```
chrsrc -s 'Name == "c175n05"' IBM.Foo \
Int32=-9999 Uint32=9999\
SD='["testing 1 2 3",1,{2,4,6}]'
```

2. To change the `Int32`, `Uint32` and `SD` resource attributes in resource class `IBM.Foo` for the resource that has a `Name` starting with `c175n`, using *resource_data_input_file* with the following contents:

```
PersistentResourceAttributes::
resource 1:
     Int32  = -9999
     Uint32 = 9999
     SD     = ["testing 1 2 3",1,{2,4,6}]
```

enter:

```
chrsrc -f /tmp/IBM.Foo.chrsrc \
-s 'Name ?= "c175n"' IBM.Foo
```

3. To change the Name persistent resource attribute for the resource that has a resource handle equal to "0x0001 0x4005 0x35ae868c 0x00000000 0xfeef2948 0x0d80b827", enter:

```
chrsrc -r "0x0001 0x4005 0x35ae868c 0x00000000 0xfeef2948 0x0d80b827" Name="c175n05"
```

4. To change the Int32, Uint32 and SD persistent resource attributes in resource class IBM.Foo for the resources that have a Name equal to Test_Name on nodes node1.linwood.com and node2.linwood.com in the cluster, using the /u/joe/common_nodes file:

```
# common node file
#
node1.linwood.com      main node
node2.linwood.com      backup node
#
```

as input, enter:

```
chrsrc -s 'Name == "Test_Name"' -N /u/joe/common_nodes IBM.Foo \
Int32=-9999 Uint32=9999 \
SD='["testing 1 2 3",1,{2,4,6}]'
```

Location

/opt/rsct/bin/chrsrc

chsec Command

Purpose

Changes the attributes in the security stanza files.

Syntax

chsec [**-f** *File*] [**-s** *Stanza*] [**-a** *Attribute* **=** *Value* ...]

Description

The **chsec** command changes the attributes stored in the security configuration stanza files. These security configuration stanza files have attributes that you can specify with the *Attribute* **=** *Value* parameter:

- **/etc/security/environ**
- **/etc/security/group**
- **/etc/security/audit/hosts**
- **/etc/security/lastlog**
- **/etc/security/limits**
- **/etc/security/login.cfg**
- **/usr/lib/security/mkuser.default**
- **/etc/nscontrol.conf**
- **/etc/security/passwd**
- **/etc/security/portlog**
- **/etc/security/pwdalg.cfg**

- **/etc/security/roles**
- **/etc/security/rtc/rtcd_policy.conf**
- **/etc/security/smitacl.user**
- **/etc/security/smitacl.group**
- **/etc/security/user**
- **/etc/security/user.roles**
- **/etc/secvars.cfg**

When modifying attributes in the **/etc/security/environ**, **/etc/security/lastlog**, **/etc/security/limits**, **/etc/security/passwd**, and **/etc/security/user** files, the stanza name specified by the *Stanza* parameter must either be a valid user name or `default`. When modifying attributes in the **/etc/security/group** file, the stanza name specified by the *Stanza* parameter must either be a valid group name or `default`. When modifying attributes in the **/usr/lib/security/mkuser.default** file, the *Stanza* parameter must be either `admin` or `user`. When modifying attributes in the **/etc/security/portlog** file, the *Stanza* parameter must be a valid port name. When modifying attributes in the **/etc/security/login.cfg** file, the *Stanza* parameter must either be a valid port name, a method name, or the **usw** attribute.

When modifying attributes in the **/etc/security/login.cfg** or **/etc/security/portlog** file in a stanza that does not already exist, the stanza is automatically created by the **chsec** command.

You cannot modify the **password** attribute of the **/etc/security/passwd** file using the **chsec** command. Instead, use the **passwd** command.

Only the root user or a user with an appropriate authorization can change administrative attributes. For example, to modify administrative group data, the user must be root or have GroupAdmin authorization.

Note: The **chsec** command changes local user attributes. It does not change non-local user attributes. You can use the **chsec** command to change remote user attributes. The **chsec** command does not update remote user attributes in local security stanza files.

Flags

Item	Description
-a *Attribute* **=** *Value*	Specifies the attribute to modify and the new value for that attribute. If you do not specify the value, the attribute is removed from the given stanza.
-f *File*	Specifies the name of the stanza file to modify.
-s *Stanza*	Specifies the name of the stanza to modify.

Security

Access Control

This command grants execute access only to the root user and the security group. The command has the trusted computing base attribute and runs the **setuid** command to allow the root user to access the security databases.

On a Trusted AIX system, only users with the **aix.mls.clear.write** authorization can modify clearance attributes. Only users with the **aix.mls.tty.write** authorization can modify the port attributes.

Auditing Events

Event	Information
USER_Change	user name, attribute
GROUP_Change	group name, attribute
PORT_Change	port, attribute

Files Accessed

Mode	File
rw	**/etc/security/environ**
rw	**/etc/security/group**
rw	**/etc/security/audit/hosts**
rw	**/etc/security/lastlog**
rw	**/etc/security/limits**
rw	**/etc/security/login.cfg**
rw	**/usr/lib/security/mkuser.default**
rw	**/etc/nscontrol.conf**
rw	**/etc/security/passwd**
rw	**/etc/security/portlog**
rw	**/etc/security/pwdalg.cfg**
rw	**/etc/security/roles**
rw	**/etc/security/rtc/rtcd_policy.conf**
rw	**/etc/security/smitacl.user**
rw	**/etc/security/smitacl.group**
rw	**/etc/security/user**
rw	**/etc/security/user.roles**

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand. To get the full functionality of the command, besides the **accessauths**, the role should also have the following authorizations:

- **aix.security.user.audit**
- **aix.security.role.assign**
- **aix.security**

To perform the **chsec** command on the **/etc/security/rtc/rtcd_policy.conf** file, the role should also have the following authorization:

- **aix.security.config**

Examples

1. To change the **/dev/tty0** port to automatically lock if 5 unsuccessful login attempts occur within 60 seconds, enter:

   ```
   chsec -f /etc/security/login.cfg -s /dev/tty0 -a logindisable=5 -a logininterval=60
   ```

2. To unlock the **/dev/tty0** port after it has been locked by the system, enter:

   ```
   chsec -f /etc/security/portlog -s /dev/tty0 -a locktime=0
   ```

3. To allow logins from 8:00 a.m. until 5:00 p.m. for all users, enter:

   ```
   chsec -f /etc/security/user -s default -a logintimes=:0800-1700
   ```

4. To change the CPU time limit of user `joe` to 1 hour (3600 seconds), enter:

```
chsec -f /etc/security/limits -s joe -a cpu=3600
```

Files

Item	Description
/usr/bin/chsec	Specifies the path to the **chsec** command.
/etc/security/environ	Contains the environment attributes of users.
/etc/security/group	Contains extended attributes of groups.
/etc/security/audit/hosts	Contains host and processor IDs.
/etc/security/group	Defines the last login attributes for users.
/etc/security/limits	Defines resource quotas and limits for each user.
/etc/security/login.cfg	Contains port configuration information.
/usr/lib/security/mkuser.default	Contains the default values for new users.
/etc/nscontrol.conf	Contains the configuration information of some name services.
/etc/security/passwd	Contains password information.
/etc/security/portlog	Contains unsuccessful login attempt information for each port.
/etc/security/pwdalg.cfg	Contains the configuration information for loadable password algorithms (LPA).
/etc/security/roles	Contains a list of valid roles.
/etc/security/rtc/rtcd_policy.conf	Contains the configuration information for the **rtcd** daemon.
/etc/security/smitacl.user	Contains user ACL definitions.
/etc/security/smitacl.group	Contains group ACL definitions.
/etc/security/user	Contains the extended attributes of users.
/etc/security/user.roles	Contains a list of roles for each user.
/etc/security/enc/LabelEncodings	Contains label definitions for the Trusted AIX system.
/etc/security/domains	Contains the valid domain definitions for the system.
/etc/secvars.cfg	Contains a stanza file.

chsecmode Command

Purpose

Changes the security mode and key types and initiates transition to the specified mode.

Syntax

chsecmode -c *mode* [**-m** *method*] [**-s** *type*] [**-f**] [**-x**] [**-h**]

Description

The **chsecmode** command sets the Reliable Scalable Cluster Technology (RSCT) security compliance mode to the **nist_sp800_131a** mode. A new generation method for the public and private keys, the symmetric key for message signing, and verification can also be specified. The NIST compliance mode can also be turned off by passing the mode as none.

If the key generation method is not specified, the current method is not changed even if the mode is still compliant with the specified new compliance mode. If the key generation method is not compliant, the **rsa2048_sha256** method is used for the **nist_sp800_131a** mode and the **rsa512** method is used for the none mode.

If the symmetric key type is default, the actual key type is chosen internally by RSCT for the specified compliance mode. In the **nist_sp800_131a** mode, the **aes256_sha256** key is used for the default symmetric key type. If the compliance mode is turned off, the appropriate symmetric key type is chosen based on the situation.

Flags

Item	Description
-c *mode*	Specifies the security compliance mode. The valid modes are: **nist_sp800_131a** and **none**.
-f	Generates new keys even if the key generation method has not changed.
-h	Displays the usage information for the **chsecmode** command.
-m *method*	Specifies an appropriate type, which is valid for the compliance mode that is used for generating the node's public or private keys. For the **nist_sp800_131a** mode, the following valid key generation methods are listed: • rsa2048_sha256 • rsa2048_sha512 • rsa3072_sha256 • rsa3072_sha512 For the non-NIST compliance mode **none**, any supported key generation methods are valid including the **rsa512** and **rsa1024** methods.

Item	Description
-s*type*	Specifies the cluster default symmetric key type. The following symmetric key types are valid for the **nist_sp800_131a** mode: • aes128_sha256 • aes128_sha512 • aes256_sha256 • aes256_sha512 For the non-NIST compliance mode none, any supported symmetric key types are valid including: • aes128_md5 • aes256_md5 • 3des_md5 • des_md5
-x	Forces the pending operation to be overwritten. If a pending change exists and the **-x** option is not specified, the **chsecmode** command fails if it is used for changing the security configuration.

Security

The **chsecmode** command permits only root to run the command.

Exit Status

0
Successful completion.

27
Invalid symmetric or asymmetric key error.

54
Invalid input parameter error.

55
THL file update failed error.

56
The **startsrc** command failed.

57
The **stopsrc** command failed.

58
The **refresh <subsystem>** command failed.

59
Invalid compliance mode error.

60
API error.

Examples

1. To enable NIST compilance mode with the compliant key generation method and the symmetric key type, enter:

```
chsecmode -c nist_sp800_131a
```

If the current method and the symmetric key types are compliant, they are not changed. If the current method and type are not compliant, the following values are used: the **rsa2048_sha256** mode for key generation method and the **aes256_sha256** mode for symmetric key type.

2. To enable the NIST compliance mode with the **rsa2048_sha512** key generation method, enter:

```
chsecmode -c nist_sp800_131a -m rsa2048_sha512
```

   If the current symmetric key is already compliant, it is not changed. If the current symmetric key is not compliant, it is replaced with the **aes256_sha256** key.

3. To enable the NIST compliance mode with the **rsa2048_sha512** key generation method and the **aes128_sha512** symmetric key, enter:

```
chsecmode -c nist_sp800_131a -m rsa2048_sha512 -s aes128_sha512
```

4. To disable NIST compliance mode, enter:

```
chsecmode -c none
```

   The current key generation method and symmetric key type is not changed.

5. To generate public and private keys by using the rsa512 key generation method, enter:

```
chsecmode -m rsa512
```

   If the current compliance mode is **nist_sp800_131a**, this operation is rejected. If the current compliance mode is none and the current key generation method is not **rsa512**, the current key generation method is replaced by **rsa512** and a new private or public key pairs are generated.

6. To force generate the public and private keys even if there is no change in the key generation method, enter:

```
chsecmode -m rsa512 -f
```

   If the current compliance mode is **nist_sp800_131a**, this operation is rejected. If the current compliance mode is none and the current key generation method is replaced by the **rsa512** method, a new private or public key pairs is generated, even if the current public or private keys are already in the **rsa512** method.

7. To overwrite or cancel any pending operation, enter:

```
chsecmode -x -c nist_sp800_131a
```

   If there is a pending compliance mode, the pending operation is ignored and a new compliance mode change to the **nist_sp800_131a** mode is started.

Location

Item	Description
/opt/rsct/bin/chsecmode	Contains the **chsecmode** command.

Files

Item	Description
/var/ct/cfg/ct_has.pkf	Default location of the cluster security services public key file for the node.
/var/ct/cfg/ct_has.qkf	Default location of the cluster security services private key file for the node.
/var/ct/cfg/ct_has.thl	Default location of the cluster security services trusted host list for the node.

chsensor Command

Purpose

Changes the attributes of a resource monitoring and control (RMC) sensor.

Syntax

`chsensor` [`-m`[`-i` *seconds*] [`-a` | `-n` *host1* [, *host2* , ...] | **-N** { *node_file* | "-" }] [`-h`] [`-v` | `-V`] *sensor_name attr1*=*value1* [*attr2*=*value2* ...]

Description

The `chsensor` command changes the attributes of a resource monitoring and control (RMC) sensor. Use the *sensor_name* parameter to specify which sensor you are changing.

The **chsensor** command runs on any node. If you want **chsensor** to run on all of the nodes in a domain, use the **-a** flag. If you want **chsensor** to run on a subset of nodes in a domain, use the **-n** flag. Instead of specifying multiple node names using the **-n** flag, you can use the **-N** *node_file* flag to indicate that the node names are in a file. Use **-N** *"–"* to read the node names from standard input.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

Flags

-a
Changes sensors that match the specified name on all nodes in the domain. The CT_MANAGEMENT_SCOPE environment variable determines the cluster scope. If CT_MANAGEMENT_SCOPE is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management domain and a peer domain exist, `chsensor -a` with CT_MANAGEMENT_SCOPE not set will run in the management domain. In this case, to run in the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-i *seconds*
Specifies the interval in which the sensor command is run to update the values of the sensor attributes. *seconds* is an integer value and must be greater than or equal to `10`. The sensor command is run at the specified interval only when a sensor resource is monitored. If the interval is set to `0`, the sensor command will not be automatically run. Using the `refsensor` command is independent of interval updates.

-m
Specifies that the resource to be changed is a microsensor resource.

-n *host1*[,*host2*...]
Specifies the node on which the sensor should be changed. By default, the sensor is changed on the local node. This flag is only appropriate in a management domain or a peer domain.

-N {*node_file* | "-"}
Specifies a file or standard input listing the nodes on which the sensor must be removed. This flag is only appropriate in a Cluster Systems Management (CSM) or a peer domain cluster.

-h
Writes the command's usage statement to standard output.

-v | **-V**
Writes the command's verbose messages to standard output.

Parameters

sensor_name
Specifies the name of the sensor to change.

***attr1=value1* [*attr2=value2* ...]**
Specifies one or more sensor or microsensor attributes and the their new values.

You can change the values of these sensor attributes:

Name
Specifies the new name of the sensor. If the new name is a string that contains spaces or special characters, it must be enclosed in quotation marks.

ControlFlags
Specifies whether special handling is required for this sensor. You can specify any combination of these values:

0
Indicates that no special handling is required. This is the default.

The sensor command runs at the interval that is defined for *sensor_name*. The **sensor** command does not run when monitoring begins or when the **lssensor** command is run. A sensor command is a command or script that the sensor resource manager runs to set and update a sensor's attribute values.

1
Indicates that the sensor command runs when monitoring begins. The sensor command also runs at the interval that is defined for *sensor_name*. The sensor command does not run when the **lssensor** command is run.

Specifying this value is not recommended, unless you expect the sensor command to run quickly. If the sensor command does not run quickly, it could block other requests to the sensor resource manager. These requests are not processed until the sensor command finishes running.

2
Indicates that output from the command in the `SavedData` field is not saved permanently to `SavedData` persistent resource attributes. If this value is not specified, the sensor resource manager updates data in the registry's resource table whenever the command's standard output contains the line: `SavedData="`*any-string*`"`.

3
Indicates a combination of values 1 and 2

4
Indicates that the sensor resource manager runs the command after monitoring is stopped.

5
Indicates a combination of values 1 and 4.

6
Indicates a combination of values 2 and 4.

7
Indicates a combination of values 1, 2, and 4.

8
Indicates that the sensor resource manager resets the dynamic attribute values after monitoring is stopped.

UserName
Specifies the name of a user whose privileges are used to run the command. The user should already be defined on the system.

Description
Provides a description of the sensor and what it is monitoring.

ErrorExitValue
Specifies which exit values are interpreted as errors, as follows:

0
No exit values are interpreted as errors.

1
Exit values other than `0` are interpreted as errors.

2
An exit value of `0` is interpreted as an error.

If the exit value indicates an error as specified by this attribute, no dynamic attribute values (except `ExitValue`) are updated.

You can change the values of these microsensor attributes:

Name
Specifies the new name of the microsensor. If the new name is a string that contains spaces or special characters, it must be enclosed in quotation marks.

Description
Provides a description of the microsensor and what it is monitoring.

Security

The user needs write permission for the `IBM.Sensor` resource class in order to run `chsensor`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0
The command has run successfully.

1
An incorrect combination of flags and parameters has been entered.

6
No sensor resources were found.

n
Based on other errors that can be returned by the RMC subsystem.

Environment Variables

CT_CONTACT
When the `CT_CONTACT` environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If this environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled.

The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Examples

1. To change the Name attribute of the SensorA sensor to Sensor1A, enter:

```
chsensor SensorA Name=Sensor1A
```

2. To change the update interval of the SensorA sensor to 10, enter:

```
chsensor -i 10 SensorA
```

3. To change the **Name** attribute of the **SensorA** sensor to **Sensor1A** on the nodes listed in the /u/joe/common_nodes file, enter:

```
chsensor -N /u/joe/common_nodes SensorA Name=Sensor1A
```

where /u/joe/common_nodes contains:

```
# common node file
#
node1.myhost.com    main node
node2.myhost.com    backup node
```

4. To change the **Name** attribute of microsensor **IBM.msensorq** to **IBM.MSensorQ**, enter:

```
chsensor -m IBM.msensorq Name=IBM.MSensorQ
```

Location

/opt/rsct/bin/chsensor

chserver Command

Purpose

Changes a subserver definition in the subserver object class.

Syntax

chserver **-t** *OldSubserver* [**-c** *CodePoint*] [**-s** *NewSubsystem*] [**-t** *NewSubserver*]

Description

The **chserver** command modifies an existing subserver definition in the subserver object class. It can change subserver types, the owning subsystem, or the subserver code point.

Flags

Item	Description
-c *CodePoint*	Specifies the *CodePoint* integer that identifies the subserver. This is the value used by the subsystem to recognize the subserver. The **chserver** command is unsuccessful if the *CodePoint* already exists for the existing subsystem name and no new subsystem name is entered. It is also unsuccessful if the *NewSubsystem* name and subserver *CodePoint* exist in the subserver object class. The limit for the *CodePoint* storage is the same as a short integer (1 through 32,768).
-s *NewSubsystem*	Specifies the name that uniquely identifies the *NewSubsystem* to the subserver it belongs to. The **chserver** command is unsuccessful if one of the following occurs: • The *NewSubsystem* name is not known in the subsystem object class. • The *NewSubsystem* name is known in the subsystem object class but uses signals as its communication method. • The *NewSubsystem* name already exists with the existing subserver *CodePoint* value in the Subserver Type object class, and no subserver *CodePoint* value is entered. • A new subserver *CodePoint* is entered, with the *NewSubsystem* name and subserver *CodePoint* already existing in the Subserver Type object class.
-t *NewSubserver*	Specifies the name that uniquely identifies the *NewSubserver*. The **chserver** command is unsuccessful if the *NewSubserver* type is already known in the subserver object class.
-t *OldSubserver*	Specifies the name that uniquely identifies the existing subserver. The **chserver** command is unsuccessful if the *OldSubserver* type is not known in the subserver object class.

Security

Auditing Events

If the auditing subsystem is properly configured and is enabled, the **chserver** command generates the following audit record (event) every time the command is run:

Event	Information
SRC_Chserver	Lists in an audit log the name of the subsystem and the fields that have been changed.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To change the subserver type, enter:

   ```
   chserver  -t old  -t new
   ```

   This changes the subserver type from the `old` subserver type to the `new` subserver type.
2. To change the owning subsystem, enter:

   ```
   chserver  -t old  -s srctest
   ```

This changes the owning subsystem to `srctest`.

3. To change the subserver type, subsystem, and subserver code point, enter:

   ```
   chserver  -t old  -t new  -s srctest  -c 1234
   ```

   This changes the subserver type from the `old` to the `new` subserver type, the owning subsystem to `srctest`, and the subserver code point to `1234`.

Files

Item	Description
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration object class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration object class.

chservices Command

Purpose

Changes the contents of the **/etc/services** file.

Syntax

To Add or Activate an Entry:

chservices [**-a**] **-v** *ServiceName* **-p** *protocol* **-n** *port* [**-u "***Alias ...***"**]

To Change an Entry:

chservices -c -v *ServiceName* **-p** *protocol* **-n** *port* [**-V** *NewServiceName*] [**-P** *NewProtocol*] [**-N** *NewPort*] [**-u "***Alias ...***"**]

To Deactivate an Entry:

chservices -d -v *ServiceName* **-p** *protocol* **-n** *port* [**-V** *NewServiceName*] [**-u** *Alias ...***"**]

Description

The **chservices** command adds, deletes, or changes entries in the **/etc/services** file. These entries are related to known services used in the DARPA Internet and also related to information used by the **inetd** server. The entries for the **inetd** server determine how the system handles Internet service requests.

The **chservices** command manipulates the following entries for known services:

- The official Internet service name specified by the *ServiceName* variable.
- The port number, specified by the *port* variable, used for the service.
- The transport protocol, specified by the *protocol* variable, used for the service.
- A list of unofficial names, specified by the *Alias* variable, used by the service.

Flags

Item	Description
-a	Adds or activates an entry in the **/etc/services** file. If the requested service exists in the file, the **-a** flag uncomments the line. If the line does not exist, the **-a** flag adds the line to the file. This is the default action.
-c	Changes an entry in the **/etc/services** file.
-d	Deactivates an entry in the **/etc/services** file by commenting the line in the file.

Item	Description
-N *NewPort*	Specifies a socket port number.
-n *port*	Specifies a socket port number.
-P *NewProtocol*	Specifies a new protocol name for a current protocol name.
-p *protocol*	Specifies the protocol.
-V *NewName*	Specifies a new service name.
-v *ServiceName*	Specifies the service name.
-u "*Alias...***"**	Specifies a list of aliases.

Note: Adding or keeping comments on lines modified with the **chservices** command is not supported.

Security

Access Control: Only the root user and members of the system group have access to this command.

Examples

1. To add the service, `gregsapp`, as a udp service on port 1423, enter:

```
chservices -a -v gregsapp -p udp -n 1423
```

2. To add the service, `gregsapp`, as a udp service on port 1423 with an alias of `fredsapp`, enter:

```
chservices -a -v gregsapp -p udp -n 1423 -u
"fredsapp"
```

3. To change the port of the service specified as `gregsapp` with a udp protocol to 1456, enter:

```
chservices -c -v gregsapp -p udp -N 1456
```

4. To deactivate the `gregsapp` service on udp port 1456 by commenting it out, enter:

```
chservices -d -v gregsapp -p udp -n 1456
```

Files

Item	Description
/usr/sbin/chservices	Contains the **chservices** command.
/etc/services	Contains services information for the **inetd** daemon.

chsh Command

Purpose

Changes a user's login shell.

Syntax

chsh [**-R** *load_module*] [*Name* [*Shell*]]

Description

The **chsh** command changes a user's login **shell** attribute. The **shell** attribute defines the initial program that runs after a user logs in to the system. This attribute is specified in the **/etc/passwd** file. By default, the **chsh** command changes the login shell for the user who gives the command.

The **chsh** command is interactive. When you run the **chsh** command, the system displays a list of the available shells and the current value of the **shell** attribute. Then, the system prompts you to change the shell. You must enter the full path name of an available shell.

If you have execute permission for the **chuser** command, you can change the login shell for another user. To change the login shell for another user, specify a *Name* parameter. Valid shells are defined in the usw stanza of the **/etc/security/login.cfg** file. The default list of valid shells is: **/usr/bin/ksh**, **/usr/bin/sh**, **/usr/bin/bsh**, **/usr/bin/csh** but your system manager may have defined more.

For users that are created with an alternate Identification and Authentication (I&A) mechanism, the **-R** flag can be used to specify the I&A load module used to create the user. Load modules are defined in the **/usr/lib/security/methods.cfg** file.

Flag

Item	Description
-R *load_module*	Specifies the loadable I&A module used to change the user's shell.

Exit Status

This command returns the following exit values:

Item	Description
0	The command runs successfully and all requested changes are made.
>0	An error occurred. The printed error message gives further details about the type of failure.

Security

Access Control

All users should have execute (x) access to this command since the program enforces its own access policy. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the **security** group with the **setgid** (SGID) bit set.

Files Accessed

Mode	File
x	**/usr/bin/chuser**
r	**/etc/security/login.cfg**
rw	**/etc/passwd**

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Limitations

Changing a user's shell may not be supported by all loadable I&A modules. If the loadable I&A module does not support changing a user's shell, an error is reported.

Examples

1. To change the shell that runs after you log in to the system, type:

   ```
   chsh
   ```

   Information similar to the following appears:

   ```
   current available shells:
    /usr/bin/sh
    /usr/bin/bsh
    /usr/bin/csh
    /usr/bin/ksh:
   current login shell:
    /usr/bin/ksh
   change (y/n)? >
   ```

   Indicate that a change should be made by entering y after the change `(y/n)?` prompt. Then, add the name of the shell you want when the `to?` prompt appears, as in the following example:

   ```
   change (y/n)? > y
   to? > /usr/bin/csh
   ```

   The next time you log in, the **/usr/bin/csh** shell appears.
2. To change the shell to `/usr/bin/ksh` for `kim`, type:

   ```
   chsh kim /usr/bin/ksh
   ```

3. To change the shell for LDAP I&A load module defined user `davis`, type:

   ```
   chsh -R LDAP davis
   ```

Files

Item	Description
/usr/bin/chsh	Specifies the path to the **chsh** command.
/usr/bin/chuser	Changes user information.
/etc/passwd	Contains the basic user attributes.
/etc/security/login.cfg	Contains login configuration information.

chslave Command

Purpose

Re-executes the **ypinit** command to retrieve maps from a controller server and re-starts the **ypserv** daemon to change the worker server.

Syntax

/usr/etc/yp/chslave [**-C** | **-c**] [**-O** | **-o**] [**-I** | **-B** | **-N**] *Master*

Description

The **chslave** command re-invokes the **ypinit** command to retrieve maps from the controller server you specify on the command line. The **ypserv** daemon is re-started after the **ypinit** command has completed successfully. The *Master* parameter specifies the host name of the controller server. The controller server specified can be the controller server currently in use or a new controller server that is configured and running.

You could also use the System Management Interface Tool (SMIT) **smit chslave** fast path to run this command.

Flags

Item	Description
-B	Invokes the **ypinit** command and starts the **ypserv** daemon. If the **ypserv** daemon is already running, this flag will cause the **ypinit** command to kill the daemon and then restart it. This flag is the default.
-C	Invokes the **ypinit** command with the **-n** flag. The **chslave** command continues on errors. This flag is the default.
-c	Stops execution when errors occur.
-I	Executes the **ypinit** command immediately but does not start or restart the **ypserv** daemon.
-O	Overwrites any maps that exist in the domain.
-o	Prevents the overwrite of maps that exist in the domain. This flag is the default.
-N	Invokes the **ypinit** command and restarts the **ypserv** daemon.

Examples

To retrieve maps from the controller server named `host91`, enter:

```
chslave -O -B host91
```

This will overwrite any existing maps for the current domain.

Files

Item	Description
/etc/rc.nfs	Contains the startup script for NFS and NIS daemons.
/var/yp/domainname	Contains the NIS maps for the NIS domain.

chsmbcred Command

Purpose

Changes the password for a specific server-user entry that is stored in the `/etc/smbcred` file to allow future mount operation of Server Message Block (SMB) client shares by using the stored credentials.

Syntax

```
chsmbcred -s server_name -u user_name [-p password]
```

Description

When you run the **chsmbcred** command, you must specify a server name and a username, and optionally you can specify a password. If the credentials for the specified server and user set is found in the `/etc/smbcred` file, the command line prompts for a password if the password is not specified with the **-p** flag. To replace the existing password for the server-user entry that is defined in the `/etc/smbcred` file, specify a password that is read as a hidden input and is stored in an encrypted format in the `/etc/smbcred` file.

Flags

-s *server_name*
Specifies the name of the remote host, which is an SMB server. The *server_name* parameter can be provided as a hostname, an IP address, or a fully qualified domain name.

-u *user_name*
Specifies the username for which the password will be changed with the specified password to enable access to the specified remote host.

-p *password*
Specifies the new password that will be used to change the existing password for the specific username of the specified remote host.

Exit status

0
The command completed successfully.

>0
An error occurred.

Example

To change the password that is stored in the `/etc/smbcred` file for `user1` to mount the SMB client file system on the `xxx.in.ibm.com` server, enter the following command:

```
chsmbcred -s xxx.in.ibm.com -u user1
```

Location

`/usr/sbin/chsmbcred`

Files

/etc/smbcred
Stores the credentials of the SMB client file system.

chssys Command

Purpose

Changes a subsystem definition in the subsystem object class.

Syntax

chssys -s *OldSubsystem* [**-a** *Arguments*] [**-e** *StandardError*] [**-i** *StandardInput*] [**-o** *StandardOutput*] [**-p** *Path*] [**-s** *NewSubsystem*] [**-t** *Synonym*] [**-u** *UserID*] [**-O** | **-R**] [**-d** | **-D**] [**-q** | **-Q**] [**-K** | [**-I** *MessageQueue* **-m** *MessageMtype* | **-f** *StopForce* **-n** *StopNormal* **-S**] [**-E** *Nice*] [**-G** *Group*] [**-w** *Wait*]

Description

The **chssys** command modifies an existing subsystem definition in the subsystem object class. If a new subsystem name is entered, the Subserver Type object class and the Notify object class are modified to reflect the new subsystem name.

> **Note:** Any auditing performed by the System Resource Controller (SRC) when actions are taken for the subsystem is logged against the login ID of the user who created the subsystem by using the **mkssys** command. For example, if you are logged in with root user authority, the subsystem is added with root user authority as the audit account.

Flags

Item	Description
-a *Arguments*	Specifies any arguments that must be passed to the program executed as the subsystem. These command *Arguments* are passed by the SRC to the subsystem according to the same rules used by the shell. Quoted strings are passed as a single argument, and blanks outside a quoted string delimit arguments. Single and double quotes can be used.
-d	Specifies that an inactive subsystem is displayed when the **lssrc -a** command request (status all) or the **lssrc -g** command request (status group) is made.
-D	Specifies that an inactive subsystem is not displayed when status all or status group requests are made.
-e *StandardError*	Specifies where the subsystem standard error data is placed.
-E *Nice*	Specifies the *Nice* value. The *Nice* parameter changes the execution priority of the subsystem. The valid values are 0 through 39 (ordinary *Nice* values mapped to all positive numbers). If the **-E** flag is not present, the subsystem priority defaults to 20. Values between 0 and 19 are reserved for users with root authority.
-f *StopForce*	Specifies the signal sent to the subsystem when a forced stop of the subsystem is requested. Use only when the subsystem uses signals for communication. The **chssys** command is unsuccessful if the *StopForce* parameter specifies an invalid signal. The **-n** and **-S** flags must follow this flag.
-G *Group*	Specifies that the subsystem belongs to the group specified by the *Group* parameter and responds to all group actions on the group.
-i *StandardInput*	Specifies where the subsystem *StandardInput* is routed. This field is ignored when the subsystem uses sockets for communication.
-K	Specifies that the subsystem uses sockets as its communication method.
-I *MessageQueue*	Specifies that the subsystem uses message queues as its communication method. The *MessageQueue* parameter specifies the message queue key for creating the message queue for the subsystem. Use the **ftok** subroutine with the subsystem path name as input to generate a unique key. The **-m** flag must follow this flag.
-m *MessageMtype*	Specifies the *MessageMtype* key that the subsystem expects on packets sent to the subsystem by the SRC. Use only when the subsystem uses message queues for communication. The *MessageMtype* must be greater than 0. This flag must be preceded by the **-l** flag.
-n *StopNormal*	Specifies the signal sent to the subsystem when a normal stop of the subsystem is requested. Use only when the subsystem uses signals for communication. The **chssys** command is unsuccessful if the *StopNormal* parameter specifies an invalid signal. This flag must be preceded by the **-f** flag and followed by the **-S** flag.
-o *StandardOutput*	Specifies where the subsystem *StandardOutput* is placed.
-O	Specifies that the subsystem is not restarted if it stops abnormally.
-p *Path*	Specifies the absolute *Path* to the subsystem program.
-q	Specifies that the subsystem can have multiple instances running at the same time.

Item	Description
-Q	Specifies that multiple instances of the subsystem are not allowed to run at the same time.
-R	Specifies that the subsystem is restarted if it stops abnormally.
-s *NewSubsystem*	Specifies the new name that uniquely identifies the subsystem. Any subservers or notify methods defined for the old subsystem's name are redefined for the *NewSubsystem* name. The **chssys** command is unsuccessful if the *NewSubsystem* name is already known in the subsystem object class.
-s *OldSubsystem*	Specifies the current name that uniquely identifies the subsystem. The **chssys** command is unsuccessful if the *OldSubsystem* name is not known in the subsystem object class.
-S	Specifies that the subsystem uses signals as its communication method. You cannot define subservers for the subsystem name when your communication method is signals. If a subserver is defined for the subsystem, the subserver definitions are deleted from the subserver object class. This flag must be preceded by the **-f** and **-n** flags.
-t *Synonym*	Specifies an alternate name for the subsystem. The **chssys** command is unsuccessful if the *Synonym* name is already known in the subsystem object class.
-u *UserID*	Specifies the user ID for the subsystem. The *UserID* that creates the subsystem is used for security auditing of that subsystem.
-w *Wait*	Specifies the time, in seconds, allowed to elapse between a stop cancel (**SIGTERM**) signal and a subsequent **SIGKILL** signal. Also used as the time limit for restart actions. If the subsystem stops abnormally more than twice in the time limit specified by the *Wait* value, it is not automatically restarted.

Security

Auditing Events

If the auditing subsystem is properly configured and is enabled, the **chssys** command generates the following audit record (event) every time the command is run:

Event	Information
SRC_Chssys	Lists in an audit log the name of the subsystem and the fields that have been changed.

For more information about properly selecting and grouping audit events, and configuring audit event data collection, see **Setting up Auditing** in *Security*.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To change the subsystem name, enter:

   ```
   chssys  -s srctest  -s inetd
   ```

   This changes the subsystem name from `srctest` to `inetd`.
2. To change the communication type to sockets, enter:

```
chssys  -s srctest  -K
```

This changes the communication type for the subsystem to sockets.

3. To change the communication type to message queues, enter:

```
chssys  -s srctest  -l 123456  -m 789
```

This changes the communication type for the subsystem to message queues, with a message queue key of 123456 and a subsystem message type of 789.

4. To change the communication type to signals, enter:

```
chssys  -s srctest  -S  -n 30  -f 31
```

This changes the communication type for the subsystem to signals, with a normal stop signal of 30 and a force stop signal of 31.

5. To change the command arguments, enter:

```
chssys  -s srctest  -a "-a 123 -b \"4 5 6\" -c '7 8 9'"
```

This places -a as the first argument, 123 as the second, -b as the third, 4 5 6 as the fourth, -c as the fifth, and 7 8 9 as the sixth argument to the srctest subsystem.

Files

Item	Description
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration object class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration object class.
/etc/objrepos/SRCnotify	Specifies the SRC Notify Method object class.
/dev/SRC	Specifies the **AF_UNIX** socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.

chsubserver Command

Purpose

Changes the contents of the /etc/inetd.conf file or similar system configuration file.

Syntax

To Add or Activate a Server or Subserver Entry:

chsubserver [**-a**] **-v** *ServiceName* **-p** *protocol* [**-t** *socket_type*][**-w** *WaitIndicator*] [**-u** *user*] [**-g** *program*] [**-r** *server*] [**-C** *ConfigFile*] [*program*] [*args*]

To Change a Server Entry:

chsubserver -c -v *ServiceName* **-p** *protocol* [**-t** *SocketType*] [**-w** *WaitIndicator*] [**-u** *user*] [**-g** *program*] [**-V** *NewServiceName*] [**-P** *NewProtocol*] [**-T** *NewSocketType*] [**-W** *NewWaitIndicator*] [**-U** *NewUser*] [**-G** *NewProgram*] [**-r** *server*] [**-C** *ConfigFile*] [*program*] [*args*]

To Deactivate a Server Entry or an inetd Subserver Entry:

chsubserver -d -v *ServiceName* **-p** *protocol* [**-t** *SocketType*] [**-w** *WaitIndicator*] [**-u** *user*] [**-g** *program*] [**-r** *server*] [**-C** *ConfigFile*] [*program*] [*args*]

Description

The **chsubserver** command adds, deletes, or changes entries in the **/etc/inetd.conf** system configuration file, which is the default, or a similar configuration file. These entries are related to known services used in the DARPA Internet and also related to information used by the **inetd** server. The entries for the **inetd** server determine how the system handles Internet service requests.

The **chsubserver** command also allows the user to refresh a server using the **-r** flag. The server specified is sent a **SIGHUP** signal to reread its configuration file. This allows you to edit the configuration file and have the changes take effect immediately.

Each service entry contains information about known services and information used by the **inetd** server. The **chsubserver** command manipulates the following entries for known services and for **inetd** server or other subserver information:

- The official Internet service name specified by the *ServiceName* variable.
- The transport protocol, specified by the *protocol* variable, used for the service.
- The type of socket, specified by the *SocketType* variable, associated with the service. The socket types associated with a service can be stream sockets or datagram sockets. Use only the **nowait** flag with stream sockets. Use either the **wait** or **nowait** flag with datagram sockets.
- A **wait** or **nowait** flag, specified by the *WaitIndicator* variable. The **wait** or **nowait** flag indicates whether the **inetd** server waits for a datagram server to release the socket before continuing to listen at the socket.
- The user name, specified by the *user* variable, that the **inetd** server uses to start a subserver.

You could also use the System Management Interface Tool (SMIT) **smit inetdconf** fast path to run this command.

Flags

Item	Description
-a	Adds or activates an entry in the configuration file. If the requested service exists in the configuration file, the **-a** flag uncomments the line. If the line does not exist, the **-a** flag adds the line to the configuration file. This is the default action.
-c	Changes an entry in the configuration file.
-C	Specifies a configuration file similar to **/etc/inetd.conf**.
-d	Deactivates an entry in the configuration file by commenting the line in the file.
-G *NewProgram*	Replaces the existing program to start.
-g *Program*	Specifies the program to start..
-P *NewProtocol*	Specifies a new protocol name for a current protocol name.
-p *protocol*	Specifies the protocol.
-r *server*	Sends a **SIGHUP** to the specified server.
-T *NewSocketType*	Replaces the existing type of socket, either a value of **stream** for stream sockets or a value of **dgram** for datagram sockets.
-t *SocketType*	Specifies a type of socket, either a value of **stream** for stream sockets or a value of **dgram** for datagram sockets.
-U *NewUser*	Replaces the existing user name.
-u *user*	Specifies a user name.
-V *NewName*	Specifies a new service name.

Item	Description
-v *ServiceName*	Specifies the service name.
-W *NewWaitIndicator*	Replaces the existing *WaitIndicator*.
-w *WaitIndicator*	Specifies either single-thread service with a value of **wait** or multithread service with a value of **nowait**.

Security

Access Control: Only the root user and members of the system group have access to this command.

Examples

1. To uncomment the uucp line in the /etc/inetd.conf file, enter:

   ```
   chsubserver -a -v uucp -p tcp
   ```

2. To add a line to the **/etc/inetd.conf** file that describes the gregserv service and runs the program `/usr/sbin/gregserv` as root over the udp protocol with stream sockets and arguments of ftpd, enter in one line:

   ```
   chsubserver -a -r inetd -v gregserv -p udp -t stream -w nowait -u
   root -g /usr/sbin/gregserv ftpd
   ```

   The **inetd** does not wait for confirmation. After adding the line to the file, the **inetd** program will be sent a **SIGHUP** signal.

3. To change the existing service from using stream sockets to using dgram sockets in the **/tmp/inetd.conf** file, enter in one line:

   ```
   chsubserver -c -v gregserv -p udp -t stream -T dgram -C /tmp/inetd.conf
   ```

4. To comment the gregserv service over udp in the **/etc/inetd.conf** file, enter:

   ```
   chsubserver -d -v gregserv -p udp
   ```

Files

Item	Description
/usr/sbin/chsubserver	Contains the **chsubserver** command.
/etc/inetd.conf	Contains configuration information for the **inetd** daemon.

chtcb Command

Purpose

Changes or queries the **trusted computing base** attribute of a file.

Syntax

chtcb { **on** | **off** | **query** } *File* ...

Description

The **chtcb** command changes or queries the **trusted computing base** (TCB) attribute of the files you specify with the *File* parameter. The following alternatives are valid:

Item	Description
on	Enables the **trusted computing base** attribute.
off	Disables the **trusted computing base** attribute, if set.
query	Displays the value of the **trusted computing base** attribute.

This command should be executed on the trusted path.

Security

Access Control: This command should grant execute (x) access to the root user and members of the security group. The command should have the **trusted computing base** attribute.

Examples

1. To identify the `plans` file as part of the trusted computing base (TCB), set the **trusted computing base** attribute to the on value by entering the following:

   ```
   chtcb on plans
   ```

   The `plans` file now can be executed from the trusted path.
2. To query whether the `plans` file is part of the trusted computing base (TCB), enter:

   ```
   chtcb query plans
   ```

   When the status appears, you know that the `plans` file is part of the trusted computing base if the TCB attribute is set to the **on** value.
3. To remove the `plans` file from the trusted computing base (TCB), enter:

   ```
   chtcb off plans
   ```

Files

Item	Description
/usr/sbin/chtcb	Contains the **chtcb** command.

chtun Command

Purpose

Changes a tunnel definition.

Syntax

chtun **-t** *tunnel_ID* **-v {4|6}** [**-s** *src_host_IP_address*] [**-d** *dst_host_IP_address*] [**-m** *pkt_mode*] [**-f** *fw_address* [**-x** *dst_mask*]] [**-e** *src_esp_algo*] [**-a** *src_ah_algo]*] [**-p** *src_policy*] [**-E** *dst_esp_algo*] [**-A** *dst_ah_algo]*] [**-P** *dst_policy*] [**-l** *lifetime*] [**-k** *src_esp_key*] [**-h** *src_ah_key*] [**-K** *dst_esp_key*] [**-H** *dst_ah_key*] [**-n** *src_esp_spi*] [**-u** *src_ah_spi*] [**-N** *dst_esp_spi*] [**-U** *dst_ah_spi*] [**-b** *src_enc_mac_algo*] [**-c** *src_enc_mac_key*] [**-B** *dst_enc_mac_algo*] [**-C** *dst_enc_mac_key*]

Description

Use the **chtun** command to change a definition of a tunnel between a local host and a tunnel partner host. If a flag is not specified, then the value given for the **gentun** command should stay the value for that field. It may also change the auto-generated filter rules created for the tunnel by the **gentun** command.

Flags

Item	Description
-A *dst_ah_algo]*	(**manual** tunnel only) Authentication algorithm, which is used by the destination for IP packet encryption. The valid values for **-A** depend on which authentication algorithms have been installed on the host. The list of all the authentication algorithms can be displayed by issuing the **ipsecstat -A** command.
-a *src_ah_algo]*	Authentication algorithm, used by source host for IP packet authentication. The valid values for **-a** depend on which authentication algorithms have been installed on the host. The list of all authentication algorithms can be displayed by issuing the **ipsecstat -A** command.
-B *dst_enc_mac_algo*	(**manual** tunnel only) Destination ESP Authentication Algorithm (New header format only). The valid values for **-B** depend on which authentication algorithms have been installed on the host. The list of all the authentication algorithms can be displayed by issuing the **ipsecstat -A** command.
-b *src_enc_mac_algo*	(**manual** tunnel only) Source ESP Authentication Algorithm (New header format only). The valid values for **-b** depend on which authentication algorithms have been installed on the host. The list of all the authentication algorithms can be displayed by issuing the **ipsecstat -A** command.
-C *dst_enc_mac_key*	(**manual** tunnel only) Destination ESP Authentication Key (New header format only). It must be a hexadecimal string started with "0x".
-c *src_enc_mac_key*	(**manual** tunnel only) Source ESP Authentication Key (New header format only). It must be a hexdecimal string started with "0x".
-d *dst_host_IP_address*	Destination Host IP address. For a host-host tunnel, this value is the IP address of the destination host interface to be used by the tunnel. For a host-firewall-host tunnel, this is the IP address of a destination host behind the firewall. A host name is also valid and the first IP address returned by the name server for the host name will be used.
-E *dst_esp_algo*	(**manual** tunnel only) Encryption algorithm, which is used by the destination for IP packet encryption. The valid values for **-E** depend on which encryption algorithms have been installed on the host. The list of all the encryption algorithms can be displayed by issuing the **ipsecstat -E** command.

Item	Description
-e *src_esp_algo*	Encryption algorithm, used by source host for IP packet encryption. The valid values for **-e** depend on which encryption algorithms have been installed on the host. The list of all encryption algorithms can be displayed by issuing the **ipsecstat -E** command.
-f *fw_address*	IP address of the firewall that is between source and destination hosts. A tunnel will be established between the source and the firewall. Therefore the corresponding tunnel definition must be made in the firewall host. A host name can also be specified with this flag, and the first IP address returned by name server for the host name will be used. The **-m** flag is forced to use default value (**tunnel**) if **-f** is specified.
-H *dst_ah_key*	The Key String for destination AH. The input must be a hexdecimal string started with "0x".
-h *src_ah_key*	The Key String for source AH. The input must be a hexdecimal string started with "0x".
-K *dst_esp_key*	The Key String for destination ESP. The input must be a hexdecimal string started with "0x".
-k *src_esp_key*	The Key String for the source ESP. It is used by the source to create the tunnel. The input must be a hexdecimal string started with "0x".
-l *lifetime*	Key Lifetime, specified in minutes. For **manual** tunnels, the value of this flag indicates the time of operability before the tunnel expires. The valid values for **manual** tunnels are 0 - 44640. Value 0 indicates that the **manual** tunnel will never expire.
-m *pkt_mode*	Secure Packet Mode. This value must be specified as **tunnel** or **transport**.
-N *dst_esp_spi*	(**manual** tunnel only) Security Parameter Index for the destination ESP.
-n *src_esp_spi*	(**manual** tunnel only) Security Parameter Index for source ESP. This SPI and the destination IP address is used to determine which security association to use for ESP.
-P *dst_policy*	(**manual** tunnel only) Destination policy, identifies how the IP packet authentication and/or encryption is to be used by destination. If the value of this flag is specified as **ea**, the IP packet gets encrypted before authentication. If specified as **ae**, it gets encrypted after authentication, whereas specifying **e** or **a** alone corresponds to the IP packet being encrypted only or authenticated only.

Item	Description
-p *src_policy*	Source policy, identifies how the IP packet authentication and/or encryption is to be used by source. If the value of this flag is specified as **ea**, the IP packet gets encrypted before authentication. If specified as **ae**, it gets encrypted after authentication, whereas specifying **e** or **a** alone corresponds to the IP packet being encrypted only or authenticated only.
-s *src_host_IP_address*	Source Host IP address, IP address of the local host interface to be used by the tunnel. A host name is also valid and the first IP address returned by name server for the host name will be used.
-t *tunnel_ID*	The tunnel identifier (ID), a locally unique, numeric identifier for a particular tunnel definition. The value must match an existing tunnel ID.
-U *dst_ah_spi*	(**manual** tunnel only) Security Parameter Index for the destination AH.
-u *src_ah_spi*	(**manual** tunnel only) Security Parameter Index for source AH. This SPI and the destination IP address is used to determine which security association to use for AH.
-v	The IP version for which the tunnel is created. For IP version 4 tunnels, use the value of **4**. For IP version 6 tunnels, use the value of **6**.
-x *dst_mask*	This flag is used for host-firewall-host tunnels. The value is the network mask for the secure network behind a firewall. The Destination host specified with the **-d** flag is a member of the secure network. The combination of the **-d** and **-x** flags allows source host communications with multiple hosts in the secure network through the source-firewall tunnel, which must be in tunnel Mode. This flag is valid only when **-f** is specified.
-y	(**manual** tunnel only) Replay prevention flag. Replay prevention is valid only when the ESP or AH header is using the new header format (see the **-z** flag). The valid values for the **-y** flag are Y (yes) and N (no).
-z	(**manual** tunnel only) New header format flag. The new header format reserves a field in ESP or AH header for replay prevention and also allows ESP authentication. The replay field is used only when the replay flag (**-y**) is set to Y. The valid values are Y (yes) and N (no).

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

chtz Command

Purpose

Changes the *TimeZoneInfo* (TZ) environment variable in the **/etc/environment** file.

Syntax

chtz *TimeZoneInfo*

Description

The **chtz** command is a high-level shell command that changes the TZ environment variable in the **/etc/environment** file. The **chtz** command returns a value of 0 if successful and nonzero if unsuccessful.

Files

Item	Description
/etc/environment	Contains variables specifying the basic environment for all processes.

chuser Command

Purpose

Changes user attributes.

Syntax

chuser [**-R** *load_module*] *Attribute*=*Value ... Name*

Description

Attention: Do not use the **chuser** command if you have a Network Information Service (NIS) database installed on your system.

The **chuser** command changes attributes for the user identified by the *Name* parameter. The user name must already exist. To change an attribute, specify the attribute name and the new value with the *Attribute*=*Value* parameter. The following files contain local user attributes that are set by this command:

- **/etc/passwd**
- **/etc/security/environ**
- **/etc/security/limits**
- **/etc/security/user**
- **/etc/security/user.roles**
- **/etc/security/audit/config**
- **/etc/group**
- **/etc/security/group**

To change attributes for a user with an alternate Identification and Authentication (I&A) mechanism, the **-R** flag can be used to specify the I&A load module that user is defined under. If the **-R** flag is not specified, the **chuser** command treats the user as a local user. Load modules are defined in the **/usr/lib/security/methods.cfg** file.

If you specify a single incorrect attribute or attribute value with the **chuser** command, the command does not change any attribute.

You can use the System Management Interface Tool (SMIT) **smit chuser** fast path to change user characteristics.

Changing the ID for an account can compromise system security and as a result one should not do so. However, when the ID is changed using the **chuser** command, ID collision checking is also controlled by the **dist_uniqid** attribute in the usw stanza of the `/etc/security/login.cfg` file. The behavior of ID collision control is the same as that described for the **mkuser** command.

Restrictions on Changing Users

To ensure the integrity of user information, some restrictions apply when using the **chuser** command. Only the root user or users with UserAdmin authorization can use the **chuser** command to perform the following tasks:

- Make a user an administrative user by setting the **admin** attribute to **true**.
- Change any attributes of an administrative user.
- Add a user to an administrative group.

An administrative group is a group with the **admin** attribute set to **true**. Members of the **security** group can change the attributes of non-administrative users and add users to non-administrative groups.

The **chuser** command manipulates local user data only. You cannot use it to change data in registry servers like NIS and DCE.

Flags

Item	Description
-R *load_module*	Specifies the loadable I&A module used to change the user's attributes.

Attributes

If you have the proper authority, you can set the following user attributes:

Item	Description
account_locked	Indicates if the user account is locked. Possible values include: **true** The user's account is locked. The values **yes**, **true**, and **always** are equivalent. The user is denied access to the system. **false** The user's account is not locked. The values **no**, **false**, and **never** are equivalent. The user is allowed access to the system. This is the default value.
admin	Defines the administrative status of the user. Possible values are: **true** The user is an administrator. Only the root user can change the attributes of users defined as administrators. **false** The user is not an administrator. This is the default value.
admgroups	Defines the groups that the user administrates. If the *domainlessgroups* attribute is set in the **/etc/secvars.cfg** file, the Lightweight Directory Access Protocol (LDAP) group can be assigned to the local user and vice versa. For more information, see **/etc/secvars.cfg** file. The *Value* parameter is a comma-separated list of group names.
auditclasses	Defines the user's audit classes. The Value parameter is a list of comma-separated classes, or a value of **ALL** to indicate all audit classes.

Item	Description
auth1	Defines the primary methods for authenticating the user. The *Value* parameter is a comma-separated list of *Method***;***Name* pairs. The *Method* parameter is the name of the authentication method. The *Name* parameter is the user to authenticate. If you do not specify a *Name* parameter, the name of the invoking login program is used. Valid authentication methods are defined in the **/etc/security/login.cfg** file. By default, the SYSTEM method and local password authentication are used. The NONE method indicates that no primary authentication check is made.
auth2	Defines the secondary methods used to authenticate the user. The *Value* parameter is a comma-separated list of *Method***;***Name* pairs. The *Method* parameter is the name of the authentication method. The *Name* parameter value is the user to authenticate. If this attribute is not specified, the default is NONE, indicating that no secondary authentication check is made. Valid authentication methods are defined in the **/etc/security/login.cfg** file. If you do not specify a *Name* parameter, the name of the invoking login program is used.
capabilities	Defines the system privileges (capabilities) which are granted to a user by the **login** or **su** commands. Valid **capabilities** are: **CAP_AACCT** Performed Advanced Accounting operations. **CAP_ARM_APPLICATION** A process has the ability to use the ARM (Application Response Measurement) services. **CAP_BYPASS_RAC_VMM** A process has the ability to bypass restrictions on VMM resource usage. **CAP_EWLM_AGENT** A process has the ability to use the EWLM (Enterprise Workload Manager) AIXsystem services. This capability is typically only granted to the userid that runs the EWLM product's Managed Server Component. **CAP_NUMA_ATTACH** A process has the ability to bind to specific resources. **CAP_PROPAGATE** All capabilities are inherited by child processes. The value is a comma-separated list of zero or more capability names.
core	Specifies the soft limit for the largest core file a user's process can create. The *Value* parameter is an integer representing the number of 512-byte blocks.
core_compress	Enables or disables core file compression. Valid values for this attribute are On and Off. If this attribute has a value of On, compression is enabled; otherwise, compression is disabled. The default value of this attribute is Off.
core_hard	Specifies the largest core file a user's process can create. The Value parameter is an integer representing the number of 512-byte blocks..
core_naming	Selects a choice of core file naming strategies. Valid values for this attribute are On and Off. A value of On enables core file naming in the form `core.`*`pid.time`*, which is the same as what the **CORE_NAMING** environment variable does. A value of Off uses the default name of **core**.

Item	Description
core_path	Enables or disables core file path specification. Valid values for this attribute are On and Off. If this attribute has a value of On, core files will be placed in the directory specified by **core_pathname** (the feature is enabled); otherwise, core files are placed in the user's current working directory. The default value of this attribute is Off.
core_pathname	Specifies a location to be used to place core files, if the **core_path** attribute is set to On. If this is not set and **core_path** is set to On, core files will be placed in the user's current working directory. This attribute is limited to 256 characters.
cpu	Identifies the soft limit for the largest amount of system unit time (in seconds) that a user's process can use. The *Value* parameter is an integer. All negative values are considered as unlimited.
cpu_hard	Identifies the largest amount of system unit time (in seconds) that a user's process can use. The Value parameter is an integer. The default value is -1 which turns off restrictions.
daemon	Indicates whether the user specified by the Name parameter can run programs using the **cron** daemon or the **src** (system resource controller) daemon. Possible values are: **true** The user can initiate **cron** and **src** sessions. This is the default. **false** The user cannot initiate **cron** and **src** sessions.
data	Specifies the soft limit for the largest data segment for a user's process. The Value parameter is an integer representing the number of 512-byte blocks. The minimum allowable value for this attribute is 1272. Specify -1 to make it unlimited.
data_hard	Specifies the largest data segment for a user's process. The Value parameter is an integer representing the number of 512-byte blocks. The minimum allowable value for this attribute is 1272. Specify -1 to make it unlimited.
dce_export	Allows the DCE registry to overwrite the local user information with the DCE user information during a DCE export operation. Possible values are: **true** Local user information is overwritten. **false** Local user information is not overwritten.
default_roles	Specifies the default roles for the user. The *Value* parameter, a comma-separated list of valid role names, can only contain roles assigned to the user in the **roles** attribute. You can use the **ALL** keyword to signify that the default roles for the user are all their assigned roles.

Item	Description
dictionlist	Defines the password dictionaries used by the composition restrictions when checking new passwords. The password dictionaries are a list of comma-separated absolute path names, evaluated from left to right. All dictionary files and directories must be write protected from all users except root. The dictionary files are formatted one word per line. The word starts in the first column and terminates with a newline character. Only 7 bit ASCII words are supported for passwords. The user names can be disallowed in the password field by adding an entry with the key word *$USER* in the dictionary files. **Note:** The key word *$USER* cannot be used as a part of any word or pattern for the entries in the dictionary files. Any password that matches with a pattern or regular expression mentioned in the dictionary file will be disallowed. To differentiate between a word and a pattern in the dictionary file, a pattern is indicated with * as the first character. For example, if an administrator wants to disallow any password ending with *123*, then this information needs to be mentioned in the dictionary file as the following entry: *.*123* The first part (*) is used to indicate a pattern entry and remaining part (*.*123*) forms the pattern. If you install the text processing tool on your system, the recommended dictionary file is the **/usr/share/dict/words** file.
domains	Defines the list of domains that the user belongs to.
expires	Identifies the expiration date of the account. The Value parameter is a 10-character string in the *MMDDhhmmyy* form, where *MM* = month, *DD* = day, *hh* = hour, *mm* = minute, and *yy* = last 2 digits of the years 1939 through 2038. All characters are numeric. If the *Value* parameter is 0, the account does not expire. The default is 0. See the **date** command for more information.
fsize	Defines the soft limit for the largest file a user's process can create or extend. The Value parameter is an integer representing the number of 512-byte blocks. To make files greater than 2G, specify -1. The minimum value for this attribute is 8192.
fsize_hard	Defines the largest file a user's process can create or extend. The Value parameter is an integer representing the number of 512-byte blocks. To make files greater than 2G, specify -1. The minimum value for this attribute is 8192.
gecos	Supplies general information about the user specified by the *Name* parameter. The *Value* parameter is a string with no embedded colon (:) character and no embedded newline character.
groups	Identifies the groups to which user belongs. If the *domainlessgroups* attribute is set in the **/etc/secvars.cfg** file, the LDAP group can be assigned to the local user and vice versa. For more information, see **/etc/secvars.cfg**. The *Value* parameter is a comma-separated list of group names.
histexpire	Defines the period of time (in weeks) that a user cannot reuse a password. The value is a decimal integer string. The default is 0, indicating that no time limit is set. Only an administrative user can change this attribute.

Item	Description
histsize	Defines the number of previous passwords that a user cannot reuse. The value is a decimal integer string. The default is 0. This attribute can have a value in the range 0 - 50. Only an administrative user can change this attribute.
home	Identifies the home directory of the user specified by the Name parameter. The *Value* parameter is a full path name.
id	Specifies the user ID. The *Value* parameter is a unique integer string. Changing this attribute compromises system security and, for this reason, you should not change this attribute.
login	Indicates whether the user can log in to the system with the **login** command. Possible values are: **true** The user can log in to the system. This is the default. **false** The user cannot log in to the system.
loginretries	Defines the number of unsuccessful login attempts allowed after the last successful login before the system locks the account. The value is a decimal integer string. A zero or negative value indicates that no limit exists. Once the user's account is locked, the user will not be able to log in until the system administrator resets the user's unsuccessful_login_count attribute in the **/etc/security/lastlog** file to be less than the value of loginretries. To do this, enter the following: `chsec -f /etc/security/lastlog -s username -a \` `unsuccessful_login_count=0`

Item	Description
logintimes	Defines the days and times that the user is allowed to access the system. The value is a comma-separated list of entries in one of the following formats: `[!]:<time>-<time>` `[!]<day>[-<day>][:<time>-<time>]` `[!]<month>[<daynum>][-<month>[<daynum>]][:<time>-<time>]` Possible values for <day> include mon, tues, w, THU, Friday, sat, and SUNDAY. Indicate the day value as any abbreviated day of the week; however, the abbreviation must be unique with respect to both day and month names. The range of days can be circular, such as Tuesday-Monday. Day names are case insensitive. Possible values for <time> include times specified in 24-hour military format. Precede the time value with a : (colon) and specify a string of 4 characters. Leading zeros are required. Thus, 0800 (8am) is valid while 800 is not valid. An entry consisting of only a specified time period applies to every day. The start hour must be less than the end hour. The time period cannot flow into the next day. Possible values for <month> include Jan, F, march, apr, and s. Indicate the month value as any abbreviated month; however, the abbreviation must be unique with respect to both day and month names. The range of months can be circular, such as September-June. Month names are case insensitive. Possible values for <daynum> include days 1-31 of a month. This value is checked against the specified month. Specify the month value as either a 1 or 2 character string. A month specified without a daynum value indicates the first or last day of the month, depending on if the month is the start or end month specified, respectively. Entries prefixed with ! (exclamation point) deny access to the system and are called DENY entries. Entries without the ! prefix allow access and are called ACCESS entries. The ! prefix applies to single entries and must prefix each entry. Currently, the system allows 200 entries per user. This attribute is internationalized. Month and day names can be entered and are displayed in the language specified by the locales variables set for the system. The relative order of the month and day values are also internationalized; the <month><daynum> and <daynum><month> formats are accepted. The system evaluates entries in the following order: 1. All DENY entries. If an entry matches the system time, the user is denied access and the ALLOW entries are not processed. 2. All ALLOW entries, if no DENY entries exist. If an ALLOW entry matches the system time, the user is allowed access. If an ALLOW entry does not match the system time, the user is denied access. If no ALLOW entry exists, the user is permitted to log in.

Item	Description
maxage	Defines the maximum age (in weeks) of a password. The password must be changed by this time. The value is a decimal integer string. The default is a value of 0, indicating no maximum age. Range: 0 to 52
maxexpired	Defines the maximum time (in weeks) beyond the **maxage** value that a user can change an expired password. After this defined time, only an administrative user can change the password. The value is a decimal integer string. The default is -1, indicating restriction is set. If the **maxexpired** attribute is 0, the password expires when the **maxage** value is met. If the **maxage** attribute is 0, the **maxexpired** attribute is ignored. Range: 0 to 52 (a root user is exempt from **maxexpired**)
maxrepeats	Defines the maximum number of times a character can be repeated in a new password. Since a value of 0 is meaningless, the default value of 8 indicates that there is no maximum number. The value is a decimal integer string. Range: 0 to 8
maxulogs	Specifies the maximum number of concurrent logins per user. If the concurrent login number for a user exceeds the maximum number of allowed logins, the login is denied.
minage	Defines the minimum age (in weeks) a password must be before it can be changed. The value is a decimal integer string. The default is a value of 0, indicating no minimum age. Range: 0 to 52
minalpha	Defines the minimum number of alphabetic characters that must be in a new password. The value is a decimal integer string. The default is a value of 0, indicating no minimum number. Range: 0 to 8
mindiff	Defines the minimum number of characters required in a new password that were not in the old password. The value is a decimal integer string. The default is a value of 0, indicating no minimum number. Range: 0 to 8
minlen	Defines the minimum length of a password. The value is a decimal integer string. The default is a value of 0, indicating no minimum length. The maximum value allowed is 8. This attribute is determined by for more information **minlen** and/or **'minalpha + minother'**, whichever is greater. **'minalpha + minother'** should never be greater than 8. If **'minalpha + minother'** is greater than 8, then the effective value for **minother** is reduced to **'8 - minalpha'**.
minother	Defines the minimum number of non-alphabetic characters that must be in a new password. The value is a decimal integer string. The default is a value of 0, indicating no minimum number. Range: 0 to 8
nofiles	Defines the soft limit for the number of file descriptors a user process may have open at one time. The **Value** parameter is an integer.
nofiles_hard	Defines the hard limit for the number of file descriptors a user process may have open at one time. The **Value** parameter is an integer. The default value is -1, which sets the limit to the maximum allowed by the system.

Item	Description
nproc	Defines the soft limit on the number of processes a user can have running at one time. The *Value* parameter is an integer equal to or greater than 1. The default value is -1, which sets the limit to the maximum allowed by the system.
nproc_hard	Defines the hard limit on the number of processes a user can have running at one time. The *Value* parameter is an integer equal to or greater than 1. The default value is -1, which sets the limit to the maximum allowed by the system.
pgrp	Identifies the primary group of the user. If the *domainlessgroups* attribute is set in the **/etc/secvars.cfg** file, the LDAP group can be assigned as a primary group to the local user and vice versa. For more information, see **/etc/secvars.cfg**. The *Value* parameter must contain a valid group name and cannot be a null value.
projects	Defines the list of projects to which the user's processes can be assigned. The value is a list of comma-separated project names and is evaluated from left to right. The project name should be a valid project name as defined in the system. If an invalid project name is found on the list, it will be reported as an error.
pwdchecks	Defines the password restriction methods enforced on new passwords. The value is a list of comma-separated method names and is evaluated from left to right. A method name is either an absolute path name or a path name relative to **/usr/lib** of an executable load module.
pwdwarntime	Defines the number of days before the system issues a warning that a password change is required. The value is a decimal integer string. A zero or negative value indicates that no message is issued. The value must be less than the difference of the **maxage** and **minage** attributes. Values greater than this difference are ignored and a message is issued when the **minage** value is reached.

Item	Description
rcmds	Controls the remote execution of the `r`-commands (`rsh`, `rexec`, and `rcp`). Possible values are as follows: **allow** Allows this user to perform remote command execution. This is the default value. **deny** Denies this user the ability to use remote command execution. **hostlogincontrol** Specifies that the ability of remote command execution is determined by the `hostsallowedlogin` and `hostsdeniedlogin` attributes. The user is only allowed to execute remote commands on a target system if the user (or target user) is allowed to log in the target system. This value is typically used for users defined in a centralized user database, such as LDAP, where the user might be allowed to log in to some systems but not others. **hostsallowedlogin** Allows the user to login to the specified hosts. **hostsdeniedlogin** The user is not allowed to login to the specified hosts. **Note:** The `rcmds` attribute controls only remote command execution. It does not control `r`-command functionality to open a remote shell. Login functions such as this are controlled by the `rlogin`, `hostsallowedlogin`, and `hostsdeniedlogin` attributes. Although the deprecated `ttys` attribute value `!rsh`, which is effectively the same as setting the `rcmds` attribute to `deny`, is still supported for purposes of backward compatibility, the `rcmds` attribute should be used instead to control the execution of `r`-commands.
rlogin	Permits access to the account from a remote location with the **telnet** or**rlogin** commands. Possible values are: **true** The user account can be accessed remotely. This is the default **rlogin** value. **false** The user cannot be accessed remotely.
roles	Defines the administrative roles for this user. The *Value* parameter is a list of role names, separated by commas.
rss	The soft limit for the largest amount of physical memory a user's process can allocate. The *Value* parameter is a decimal integer string specified in units of 512-byte blocks. This value is not currently enforced by the system.
rss_hard	The largest amount of physical memory a user's process can allocate. The Value parameter is a decimal integer string specified in units of 512-byte blocks. This value is not currently enforced by the system.
shell	Defines the program run for the user at session initiation. The *Value* parameter is a full path name.

Item	Description
stack	Specifies the soft limit for the largest process stack segment for a user's process. The *Value* parameter is an integer representing the number of 512-byte blocks to allot. The minimum allowable value for this attribute is 49.
stack_hard	Specifies the largest process stack segment of a user's process. The *Value* parameter is an integer representing the number of 512-byte blocks to allot. The minimum allowable value for this attribute is 49. The largest allowable value for this parameter is 2147483647.
su	Indicates whether another user can switch to the specified user account with the **su** command. Possible values are: **true** Another user can switch to the specified account. This is the default. **false** Another user cannot switch to the specified account.
sugroups	Defines the groups that can use the **su** command to switch to the specified user account. The *Value* parameter is a comma-separated list of group names, or a value of **ALL** that indicates all groups. An exclamation point (!) in front of a group name excludes that group. If this attribute is not specified, all groups can switch to this user account by using the **su** command. If the *domainlessgroups* attribute is set in the **/etc/secvars.cfg** file, the LDAP group can be assigned to the local user and vice versa. For more information, see **/etc/secvars.cfg** file. **Note:** If a user belongs to multiple groups and any of the groups is specified with the exclamation point (!), then user cannot use the **su** command to access the specified user account.
sysenv	Identifies the system-state (protected) environment. The *Value* parameter is a set of comma-separated *Attribute*=*Value* pairs as specified in the **/etc/security/environ** file.
threads	Specifies the soft limit for the largest number of threads that a user process can create. The *Value* parameter is an integer equal to or greater than 1, representing the number of threads each user process can create. This limit is enforced by both the kernel and the user space **pthread** library.
threads_hard	Specifies the largest possible number of threads that a user process can create. The *Value* parameter is an integer equal to or greater than 1, representing the number of threads each user process can create. This limit is enforced by both the kernel and the user space **pthread** library.

Item	Description
tpath	Indicates the user's trusted path status. The possible values are: **always** The user can only execute trusted processes. This implies that the user's initial program is in the trusted shell or some other trusted process. **no tsh** The user cannot invoke the trusted shell on a trusted path. If the user enters the secure attention key (SAK) after logging in, the login session ends. **nosak** The secure attention key (SAK) is disabled for all processes run by the user. Use this value if the user transfers binary data that may contain the SAK sequence. This is the default value. **on** The user has normal trusted path characteristics and can invoke a trusted path (enter a trusted shell) with the secure attention key (SAK).
ttys	Defines the terminals that can access the account specified by the *Name* parameter. The *Value* parameter is a comma-separated list of full path names, or a value of **ALL** to indicate all terminals. An ! (exclamation point) in front of a terminal name excludes that terminal. If this attribute is not specified, all terminals can access the user account.
umask	Determines file permissions. This value, along with the permissions of the creating process, determines a file's permissions when the file is created. The default is 022.
usrenv	Defines the user-state (unprotected) environment. The *Value* parameter is a set of comma-separated *Attribute***=***Value* pairs as specified in the **/etc/security/environ** file.
efs_keystore_access	Specifies the database type of the user keystore. You can specify the following values: **file** Creates the **/var/efs/users/*usrname*/keystore** keystore file associated with the user. **none** Keystore is not created. All the other keystore attributes have no effect. The default value is **file**. **Restriction**: The attribute is valid only when the system is EFS-enabled.
efs_adminks_access	Represents the database type for the **efs_admin** keystore. The only valid value is **file**. **Restriction**: The attribute is valid only when the system is EFS-enabled.

Item	Description
efs_initialks_mode	Specifies the initial mode of the user keystore. You can specify the following values: **admin** Root or other security privileged system users can open the keystore using the admin key and reset the keystore password. **guard** Root users cannot open the keystore using the admin key or reset the keystore password. The default value is **admin**. The attribute specifies the initial mode of the user keystore. You can use the attribute with the **mkuser** command. After the keystore has been created, changing the attribute value with the **chuser**, **chgroup**, or **chsec** command, or manual editing does not change the mode of the keystore unless the keystore is deleted and a new one is created. To change the keystore mode, use the **efskeymgr** command. **Restriction**: The attribute is valid only when the system is EFS-enabled.
efs_allowksmodechangebyuser	Specifies whether the mode can be changed. You can specify the following values: • **yes** • **no** The default value is **yes**. **Restriction**: The attribute is valid only when the system is EFS-enabled.
efs_keystore_algo	Specifies the algorithm that is used to generate the private key of the user during the keystore creation. You can specify the following values: • **RSA_1024** • **RSA_2048** • **RSA_4096** The default value is **RSA_1024**. You can use the attribute with the **mkuser** command. After the keystore has been created, changing the value of this attribute with the **chuser**, **chgroup**, or **chsec** command, or manual editing does not regenerate the private key unless the keystore is deleted and a new one is created. To change the algorithm for the keys, use the **efskeymgr** command. **Restriction**: The attribute is valid only when the system is EFS-enabled.

Item	Description
efs_file_algo	Specifies the encryption algorithm for user files. You can specify the following values: • **AES_128_CBC** • **AES_128_ECB** • **AES_192_CBC** • **AES_192_ECB** • **AES_256_CBC** • **AES_256_ECB** The default value is **AES_128_CBC**. **Restriction**: The attribute is valid only when the system is EFS-enabled.
minsl	Defines the minimum sensitivity-clearance level that the user can have. **Note:** This attribute is valid only for Trusted AIX. The valid values are defined in the "Clearances" section of the **/etc/security/enc/LabelEncodings** file for the system. The value must be defined in quotation marks if it has white spaces. The **minsl** value must be dominated by the **defsl** value for the user.
maxsl	Defines the maximum sensitivity-clearance level that the user can have. **Note:** This attribute is valid only for Trusted AIX. The valid values are defined in the "Clearances" section of the **/etc/security/enc/LabelEncodings** file. The value must be defined in quotation marks if it has white spaces. The **maxsl** value must dominate the **defsl** value for the user.
defsl	Defines the default sensitivity level that the user is assigned during login. **Note:** This attribute is valid only for Trusted AIX. The valid values are defined in the "Clearances" section of the **/etc/security/enc/LabelEncodings** file. The value must be defined in quotation marks if it has white spaces. The **defsl** value must dominate the **minsl** value and be dominated by the **maxsl** value.
mintl	Defines the minimum integrity clearance level that the user can have. **Note:** This attribute is valid only for Trusted AIX. The valid values are defined in the "Sensitivity labels" section of the **/etc/security/enc/LabelEncodings** file . If the optional "Integrity labels" section is defined in the **/etc/security/enc/LabelEncodings** file, the value must be from this section. The value must be defined in quotation marks if it contains white spaces. The **mintl** value must be dominated by the **deftl** value for the user.

Item	Description
maxtl	Defines the maximum integrity clearance level that the user can have. **Note:** This attribute is valid only for Trusted AIX. The valid values are defined in the "Sensitivity labels" section of the **/etc/security/enc/LabelEncodings** file . If the optional "Integrity labels" section is defined in the **/etc/security/enc/LabelEncodings** file, the value must be from this section. The value must be defined in quotation marks if it contains white spaces. The **maxtl** value must dominate the **deftl** value for the user.
deftl	Defines the default integrity clearance level that the user is assigned during login. **Note:** This attribute is valid only for Trusted AIX. The valid values are defined in the "Sensitivity labels" section of the **/etc/security/enc/LabelEncodings** file . If the optional "Integrity labels" section is defined in the **/etc/security/enc/LabelEncodings** file, the value must be from this section. The value must be defined in quotation marks if it contains white spaces. The **deftl** value must dominate the **mintl** value and be dominated by the **maxtl** value.
minloweralpha	Defines the minimum number of lower case alphabetic characters that must be in a new password. The value is a decimal integer string. The default is a value of 0, indicating no minimum number. Range: 0 to PW_PASSLEN.
minupperalpha	Defines the minimum number of upper case alphabetic characters that must be in a new password. The value is a decimal integer string. The default is a value of 0,indicating no minimum number. Range: 0 to PW_PASSLEN.
mindigit	Defines the minimum number of digits that must be in a new password. The value is a decimal integer string. The default is a value of 0, indicating no minimum number. Range: 0 to PW_PASSLEN.
minspecialchar	Defines the minimum number of special characters that must be in a new password. The value is a decimal integer string. The default is a value of 0, indicating no minimum number. Range: 0 to PW_PASSLEN.

Security

Access Control

This command must grant execute (x) access only to the root user and the security group. This command must be installed as a program in the trusted computing base (TCB). The command must be owned by the root user with the **setuid** (SUID) bit set.

On a Trusted AIX system, only users with the aix.mls.clear.write authorization can modify the attributes **minsl, maxsl, defsl, mintl, maxtl** and **deftl**.

Auditing Events

Event	Information
USER_Change	user, attributes

Files Accessed

Mode	File
rw	**/etc/passwd**
rw	**/etc/security/user**
rw	**/etc/security/user.roles**
rw	**/etc/security/limits**
rw	**/etc/security/environ**
rw	**/etc/security/audit/config**
rw	**/etc/group**
rw	**/etc/security/group**
r	**/etc/security/enc/LabelEncodings**
r	**/etc/security/domains**

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand. To get the full functionality of the command, besides the **accessauths**, the role should also have the following authorizations:

- **aix.security.user.audit**
- **aix.security.role.assign**
- **aix.security.group.change**

Limitations

Changing a user's attributes may not be supported by all loadable I&A modules. If the loadable I&A module does not support changing a user's attributes, an error is reported.

Examples

1. To enable user `smith` to access this system remotely, type:

   ```
   chuser rlogin=true smith
   ```

2. To change the expiration date for the `davis` user account to 8 a.m., 1 May, 1995, type:

   ```
   chuser expires=0501080095 davis
   ```

3. To add `davis` to the groups `finance` and `accounting`, type:

   ```
   chuser groups=finance,accounting davis
   ```

4. To change the user `davis`, who was created with the LDAP load module, to not be allowed remote access, type:

   ```
   chuser -R LDAP rlogin=false davis
   ```

5. To change the domains of the user `davis`, type:

   ```
   chuser domains=INTRANET,APPLICATION davis
   ```

6. To unset the roles of the user `davis`, type:

```
chuser roles=" " davis
```

Files

Item	Description
/usr/bin/chuser	Contains the **chuser** command.
/etc/passwd	Contains the basic attributes of users.
/etc/group	Contains the basic attributes of groups.
/etc/security/group	Contains the extended attributes of groups.
/etc/security/user	Contains the extended attributes of users.
/etc/security/user.roles	Contains the administrative role attributes of users.
/etc/security/lastlog	Contains the last login attributes of users.
/etc/security/limits	Defines resource quotas and limits for each user.
/etc/security/audit/config	Contains audit configuration information.
/etc/security/environ	Contains the environment attributes of users.
/etc/security/enc/LabelEncodings	Contains the label definitions for the Trusted AIX system.
/etc/security/domains	Contains the valid domain definitions for the system.

chusil Command

Purpose

Changes an attribute of an existing user-specified installation location (USIL) instance.

Syntax

chusil **-R** *RelocatePath* **-c** *NewComments* [**-X**]

Description

The **chusil** command changes an attribute of an existing USIL instance.

Flags

Item	Description
-c *NewComments*	Specifies the new comments to include in the USIL definition (visible with the **lsusil** command).
-R *RelocatePath*	Specifies the path to an existing USIL location.
-X	Expands the space needed automatically.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/usr/sbin/chusil	Contains the **chusil** command.

chvfs Command

Purpose

Changes entries in the **/etc/vfs** file.

Syntax

chvfs *VFSEntry*

Description

The **chvfs** command changes **/etc/vfs** file entries by specifying the following fields within the *VFSEntry* parameter. The *VFSEntry* parameter is composed of the following fields: *VFSName***:***VFSNumber***:***MountHelper***:***FileSystemHelper* **.**

Any of the entries in the *VFSEntry* can be null (empty), with the exception of the *VFSName* field, and the corresponding value will not be changed. If all of the arguments are satisfactory, the entry in the **/etc/vfs** file is changed.

Parameters

Item	Description
VFSEntry	A string in the following format: *VFSName***:***VFSNumber***:***MountHelper***:***FileSystemHelper*
VFSName	Specifies the name of a virtual file system type.
VFSNumber	Specifies the virtual file system type's internal number as known by the kernel.
MountHelper	Specifies the name of the backend used to mount a file system of this type.
FileSystemHelper	Specifies the name of the backend used by certain file system specific commands to perform operations on a file system of this type.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To change the *FileSystemHelper* for the vfs entry named `newvfs`, enter:

```
chvfs "newvfs:::/etc/helper/testhelper"
```

The missing parameters are left unchanged.

Files

Item	Description
/etc/vfs	Contains descriptions of virtual file system types.

chvg Command

Purpose

Sets the characteristics of a volume group.

Syntax

chvg [**-s** *Sync* { **y** | **n** }] [**-h** *Hotspare* {**y** | **Y** | **n** | **r** }] [**-a** *AutoOn* { **y** | **n** }] [**-c** | **-l**] [**-L** *LTGSize*] [**-Q** { **y** | **n** }][**-X** { **none** | **SSD** }] [**-u**] [**-r** { **y** | **n** }] [**-x** { **y** | **n** }] [**-S** | **-R**] [**-t** [*factor*]] [**-B** | **-G**] [**-P**] [**-v**][**-C**] [**-f**] [**-g**] [**-b** { **y** | **n** }] [**-I**] [**-O** { **y** | **n** }] [**-M** { **y** | **n** | **s** }] [**-N o**|**n**] [**-j** { **y** | **n** }] [**-e** *y*|*n*] [**-k y** | **n**] *VolumeGroup*

Description

The **chvg** command changes the characteristics of a volume group. You can also use the System Management Interface Tool (SMIT) **smit chvg** fast path to run this command.

Flags

Note:

1. Only the **-a**, **-R**, **-S**, **-u**, and **-h** options are allowed on the volume group that has a snapshot volume group.
2. Only the **-a**, **-R**, **-S**, and **-u** options are allowed on the snapshot volume group.
3. Changing a VG to a Big VG format **(-B flag)** or to a Scalable VG format **(-G flag)** and specifying the data encryption option **(-k flag)** cannot be combined with any other change operation.
4. Bad block relocation policy is not supported on a volume group that is created with 4 KB block physical volumes.

Item	Description
-a *AutoOn*	Determines if the volume group is automatically activated during system startup. The *AutoOn* variable can be either of the following: **y** The volume group is automatically activated during system startup. **n** The volume group is not automatically activated during system startup.
-b	Sets the bad-block relocation policy of a volume group. The default value is yes. **y** Will turn on the bad-block relocation policy of a volume group. **n** Turns off the bad block relocation policy of a volume group.

Item	Description
-B	Changes the volume group to Big VG format. This can accommodate up to 128 physical volumes and 512 logical volumes. **Note:** 1. The **-B** flag cannot be used if there are any stale physical partitions. 2. Once the volume group is converted, it cannot be imported into AIX Version 6.1 or lower versions. 3. The **-B** flag cannot be used if the volume group is varied on in concurrent mode. 4. There must be enough free partitions available on each physical volume for the VGDA expansion for this operation to be successful. 5. Because the VGDA resides on the edge of the disk and it requires contiguous space for expansion, the free partitions are required on the edge of the disk. If those partitions are allocated for user usage, they will be migrated to other free partitions on the same disk. The rest of the physical partitions will be renumbered to reflect the loss of the partitions for VGDA usage. This will change the mappings of the logical to physical partitions in all the PVs of this VG. If you have saved the mappings of the LVs for a potential recovery operation, you should generate the maps again after the completion of the conversion operation. Also, if the backup of the VG is taken with the map option and you plan to restore using those maps, the restore operation may fail since the partition number may no longer exist (due to reduction). It is recommended that backup is taken before the conversion, and right after the conversion if the map option is utilized. 6. Because the VGDA space has been increased substantially, every VGDA update operation (creating a logical volume, changing a logical volume, adding a physical volume, and so on) may take considerably longer to run.
-c	Same as **-C** flag. In AIX Version 6.1 and later only Enhanced Concurrent Capable volume groups will be created.
-C	Changes the volume group into an Enhanced Concurrent Capable volume group. Changes the volume group varied on in non-concurrent mode to Enhanced Concurrent Capable. This requires that the volume group be re-imported on all other nodes prior to activation in Enhanced Concurrent mode. Changes the volume group varied on in Concurrent mode to an Enhanced Concurrent mode volume group. Only use the **-C** flag with the PowerHA SystemMirror ES. It has no effect on volume groups and systems not using the HACMP ES product. Enhanced Concurrent volume groups use Group Services. Group Services ships with PowerHA SystemMirror ES and must be configured prior to activating a volume group in this mode. Use this flag to change a volume group into an Enhanced Concurrent Capable volume group. **Note:** 1. Enhanced Concurrent volume groups use Group Services. Group Services ships with HACMP ES and must be configured prior to activating a volume group in this mode. 2. Only Enhanced Concurrent Capable volume groups are supported when running with a 64-bit kernel. Concurrent Capable volume groups are not supported when running with a 64-bit kernel. 3. Enhanced Concurrent Capable volume groups always have multinode varyon protection enabled. See the **-N** flag for details about multinode varyon protection.

Item	Description
-e *y*|*n*	Enables the `Critical PVs` option of the volume group. This flag is available in IBM AIX 7.2 with Technology Level 1, or later. **y** Enables the `Critical PVs` option of the volume group. If write request failures occur in the mirrored logical volume, the PV is marked as missing and it stops sending I/O requests to the failed mirrored logical volume. If the `Critical PVs` option is enabled in a volume group, you can import only the volume group into IBM AIX 7.2 with Technology Level 1, or later. **n** The `Critical PVs` option is not used. This is the default value.
-f	Forces the volume group to be created on the specified physical volume unless the physical volume is part of another volume group in the Device Configuration Database or a volume group that is active.
-g	Will examine all the disks in the volume group to see if they have grown in size. If any disks have grown in size attempt to add additional PPs to PV. If necessary will determine proper 1016 multiplier and conversion to big vg. **Note:** The user might be required to execute **varyoffvg** and then **varyonvg** on the volume group for LVM to see the size change on the disks.
-G	Changes the volume group to Scalable VG format. This can accommodate up to 1024 physical volumes and 4096 logical volumes. **Notes:** 1. The **-G** flag cannot be used if there are any stale physical partitions. 2. Once the volume group is converted, it cannot be imported into AIX Version 6.1 or lower versions. 3. The **-G** flag cannot be used if the volume group is varied on. 4. There must be enough free partitions available on each physical volume for the VGDA expansion for this operation to be successful. 5. Since the VGDA resides on the edge of the disk and it requires contiguous space for expansion, the free partitions are required on the edge of the disk. If those partitions are allocated for user usage, they will be migrated to other free partitions on the same disk. The rest of the physical partitions will be renumbered to reflect the loss of the partitions for VGDA usage. This will change the mappings of the logical to physical partitions in all the PVs of this VG. If you have saved the mappings of the LVs for a potential recovery operation, you should generate the maps again after the completion of the conversion operation. Also, if the backup of the VG is taken with the map option and you plan to restore using those maps, the restore operation may fail since the partition number may no longer exist (due to reduction). It is recommended that backup is taken before the conversion, and right after the conversion if the map option is utilized. 6. Because the VGDA space has been increased substantially, every VGDA update operation (creating a logical volume, changing a logical volume, adding a physical volume, and so on) may take considerably longer to run. 7. Changing an existing volume group to Scalable VG format will change the device subtype (reported by the IOCINFO ioctl() call) for all associated LVs to DS_LVZ, regardless of the previous subtype. This alteration does not change any behavior of the LV's beyond the reported subtype.

Item	Description
-h *Hotspare*	Sets the sparing characteristics for the volume group specified by the *VolumeGroup* parameter. Either allows (**y**) the automatic migration of failed disks, or prohibits (**n**) the automatic migration of failed disks. This flag has no meaning for non-mirrored logical volumes **y** Enhances the automatic migration of failed disks by permitting one for one migration of partitions from one failed disk to one spare disk. The smallest disk in the volume group spare pool that is big enough for one to one migration will be used. **Y** Permits the automatic migration of failed disks and allows migration to the entire pool of spare disks, as opposed to a one for one migration of partitions to a spare. **n** Prohibits the automatic migration of a failed disk. This is the default value for a volume group. **r** Removes all disks from the *Hotspare* pool for the volume group. **Note:** This flag is not supported for the concurrent capable volume groups.
-I	Modifies the volume group so that it can be imported to AIX Version 6.1. The *LTGSize* will behave as if the volume group had been created prior to AIX Version 6.1. This operation might fail if the volume group contains striped logical volumes whose strip size (a strip size multiplied by the number of disks in an array equals the stripe size) is larger than the supported strip size on AIX Version 6.1. If logical volumes are later created with a strip size that is larger than the supported strip size on AIX Version 6.1, then attempting to import the volume group back to AIX Version 6.1 is not supported.
-j y | n	If the Enhanced Journaled File System (JFS2) is mounted, the resync operation of the logical volume manager (LVM) resynchronizes the blocks that are allocated only by the JFS2. You can specify the following values for this flag: **y** Resynchronizes the blocks that are allocated only by the JFS2. **n** Resynchronizes all of the blocks regardless of the JFS2 block allocations. This is the default value.

Item	Description
-k y | n	Changes the data encryption option in the volume group. The **-k** flag is available in IBM AIX 7.2 with Technology Level 5, or later. You can specify the following values for this flag: **y** Changes the data encryption option in the volume group. If the data encryption option is enabled in a volume group, you can import the volume group into an AIX LPAR that is running AIX 7 with 7200-05, or later. **n** Disables the data encryption option in the volume group. **Note:** • The **-k** flag cannot be used if the volume group is varied on in the concurrent mode. • The **-k** flag cannot be used on rootvg. • Each physical volume must have sufficient free partitions for the encryption metadata for a successful completion of this operation. • The encryption metadata resides at the end of the disk sectors and requires contiguous space for expansion. Therefore, free partitions are required at the end of the disk sectors. If the partitions are in use or allocated to a logical volume, the logical partitions are migrated to other free partitions on the same disk. The remaining physical partitions on the physical volume are renumbered after the loss of the partitions for encryption metadata usage. This renumbering might change the mappings of the logical partition to physical partitions in all the physical volumes of the VG. If you had saved the mappings of the LVs for a recovery operation, you must recreate the maps after the data encryption option is modified. Also, if a backup of the VG is saved with the map option, and you plan to restore the VG by using those maps, the restore operation might fail because the partition number might not exist due to the renumbering of the physical partition. As a best practice, you must back up the VG before and after you change the data encryption option if the map option is used. When you enable data encryption in a volume group, some disk space is reserved at end sector of the disk. You must reclaim some physical partitions (PPs) at the end sector of the disks to have sufficient free space for the encryption metadata. If the required PPs are in use, the **chvg** command returns an error. When you disable data encryption in a volume group, the validity of the operation is checked. If the volume group contains any encrypted logical volumes, the **chvg** command returns an error.
-l	Changes the volume group into a Non-Concurrent Capable volume group. The volume group must be varied on in non-concurrent mode for this command to take effect.
-L	For volume groups created on AIX Version 6.1, the **-L** flag is ignored. When the volume group is varied on, the logical track group size will be set to the common max transfer size of the disks. For volume groups created prior to AIX Version 6.1, the **-L** flag changes the logical track group size, in number of kilobytes, of the volume group. The value of the *LTGSize* parameter must be 0, 128, 256, 512, or 1024. In addition, it should be less than or equal to the maximum transfer size of all disks in the volume group. The default size is 128 kilobytes. An *LTGSize* of 0 will cause the next varyonvg to set the logical track group size to the common max transfer size of the disks.

Item	Description
-M	Changes the mirror pool strictness for the volume group. **y** Each logical volume copy created in the volume group must be assigned to a mirror pool. **n** No restrictions are placed on the user of mirror pool. This is the default value. **s** Super-strict mirror pools are enforced on the volume group. **Note:** 1. Local and remote physical volumes cannot belong to the same mirror pool. 2. A maximum of three mirror pools can be in a volume group. 3. Each mirror pool must contain at least one copy of each logical volume in the volume group.
-N o|n	**o** Changes the volume group that is allowed to varyon in the non-concurrent mode in more than one node at the same time. **n** Changes the VG that is not allowed to varyon in non-concurrent mode in more than one node at the same time. **Note:** • This VG can no longer be imported on a version of AIX that does not support this flag. • This option is not available for volume groups varied on in the concurrent mode.
-O y | n	Changes the infinite retry option of the volume group. **y** Enables the infinite retry option of the volume group. The failed I/O request is retried until it is successful. **n** Disables the infinite retry option of the volume group. The failing I/O on the volume group is not retried. It does not affect the logical volume infinite retry option. **Note:** Infinite retry is not supported in a GLVM environment.
-P *PhysicalPartitions*	Increases the number of physical partitions a volume group can accommodate. Where the *PhysicalPartitions* variable is represented in units of 1024 partitions. Valid values are 64, 128, 256, 512 768, 1024 and 2048. The value should be larger than the current value or no action is taken. This option is only valid with Scalable-type volume groups.
-Q	Determines if the volume group is automatically varied off after losing its quorum of physical volumes. The default value is yes. The change becomes effective immediately. **n** The volume group stays active until it loses all of its physical volumes. **y** The volume group is automatically varied off after losing its quorum of physical volumes.

Item	Description
-X *none* | *SSD*	Sets or changes a PV type restriction on the VG. Once a PV restriction is turned on, the VG can no longer be imported on a version of AIX that does not support PV type restrictions. The use of the **-I** flag on a PV restricted VG is prohibited. **none** Removes a PV type restriction on the VG. This flag has no effect if the VG was not previously PV restricted. **SSD** Places a PV type restriction on the VG if all the underlying disks are of type SSD. Displays an error message if one or more of the existing PV's in the VG does not meet the restriction.
-r y | n	Changes the critical volume group (VG) option of the volume group. **n** Disables the critical VG option of the volume group. **y** Enables the critical VG option of the volume group. If the volume group is set to the critical VG, any I/O request failure starts the Logical Volume Manager (LVM) metadata write operation to check the state of the disk before returning the I/O failure. If the critical VG option is set to rootvg and if the volume group losses access to quorum set of disks (or all disks if quorum is disabled), instead of moving the VG to an offline state, the node is crashed and a message is displayed on the console. **Notes:** • The critical VG can no longer be imported into a version of AIX that does not support the **-r** flag. • The critical VG option is not available for the volume groups that are varied on in concurrent mode.
-R	Resumes normal I/O operations for a volume group.
-s *Sync*	Sets the synchronization characteristics for the volume group specified by the *VolumeGroup* parameter. Either permits (**y**) the automatic synchronization of stale partitions or prohibits (**n**) the automatic synchronization of stale partitions. This flag has no meaning for non-mirrored logical volumes. Automatic synchronization is a recovery mechanism that will only be attempted after the LVM device driver logs **LVM_SA_STALEPP** in the **errpt**. A partition that becomes stale through any other path (for example, **mklvcopy**) will not be automatically resynced. **y** Attempts to automatically synchronize stale partitions. **n** Prohibits automatic synchronization of stale partitions. This is the default for a volume group. **Note:** This flag is not supported for the concurrent capable volume groups.
-S	Drains I/O's for this volume group and suspends future I/O's.

Item	Description
-t [*factor*]	Changes the limit of the number of physical partitions per physical volume, specified by *factor*. *factor* should be between 1 and 16 for 32 disk volume groups and 1 and 64 for 128 disk volume groups. If *factor* is not supplied, it is set to the lowest value such that the number of physical partitions of the largest disk in volume group is less than *factor* x 1016. If *factor* is specified, the maximum number of physical partitions per physical volume for this volume group changes to *factor* x 1016. **Note:** 1. This option is ignored for Scalable-type volume groups. 2. *factor* cannot be changed if there are any stale physical partitions in the volume group. 3. This flag cannot be used if the volume group is varied on in concurrent mode. 4. The maximum number of physical volumes that can be included in this volume group will be reduced to (MAXPVS/factor). 5. Changing an existing volume group to Scalable VG format will change the device subtype (reported by the IOCINFO ioctl() call) for all associated LVs to DS_LVZ, regardless of the previous subtype. This alteration does not change any behavior of the LV's beyond the reported subtype.
-u	Unlocks the volume group. This option is provided if the volume group is left in a locked state by abnormal termination of another LVM operation (such as the command core dumping, or the system crashing). **Note:** Before using the **-u** flag, make sure that the volume group is not being used by another LVM command.
-v *LogicalVolumes*	Increases the number of logical volumes that can be created. Valid values are 512, 1024, 2048 and 4096. The value should be larger than the current value or no action is taken. This option is only valid with Scalable-type volume groups.
-x	Changes the mode which the Concurrent Capable volume group is varied on. The volume group must be varied on in non-concurrent mode for this command to take effect. **Note:** There is no auto on support for Enhanced Concurrent Capable volume groups. On AIX Version 6.1 and later only Enhanced Concurrent Capable volume groups will be created. **y** autovaryon the volume group in concurrent mode. **n** autovaryon the volume group in non-concurrent mode. **Note:** If the volume group is not created Concurrent Capable, this command has no effect on the volume group. In order for this auto-varyon into concurrency of the volume group to take effect, you must enter the following line into the **/etc/inittab** file: `rc_clvmv:2:wait:/usr/sbin/clvm_cfg 2>&1`  **Attention:** This entry must be added after the entry used to initiate **srcmstr**.

Examples

1. To cause volume group vg03 to be automatically activated during system startup, type:

   ```
   chvg  -a y vg03
   ```

2. To change the volume group vg03 to a supported state if it is in violation of 1016 physical partitions per physical volume limit, type:

   ```
   chvg  -t  vg03
   ```

3. To change the maximum number of physical partitions per physical volume to 2032 and maximum number of physical volumes in volume group vg03 to 16, type:

   ```
   chvg  -t 2 vg03
   ```

Files

Item	Description
/**usr**/**sbin**	Directory where the **chvg** command resides.

chvirprt Command

Purpose

Changes the attribute values of a virtual printer.

Syntax

chvirprt **-d** *QueueDeviceName* **-q** *PrintQueueName* [**-a** *Attribute*=*Value* ...]

Description

The **chvirprt** command changes attribute values for the virtual printer assigned to *PrintQueueName* and *QueueDeviceName*.

> **Note:** Attribute names for default values of the **qprt**command line flags can be specified by entering the flag letters. For example, to change the default value for the **-w** flag (page width) to 132, enter w=132. All other attribute names must be 2 characters long.

You could also use the System Management Interface Tool (SMIT) **smit chvirprt** fast path to run this command.

Flags

Item	Description
-a *Attribute*=*Value*	Replaces the value for *Attribute* with *Value*. If *Value* contains one or more spaces, it must be surrounded by quotes ('*Value*'). be the last flag when entering the **chvirprt** command on the command line.
-d *QueueDeviceName*	Specifies the name of the queue device to which the virtual printer is assigned.
-q *PrintQueueName*	Specifies the name of the print queue to which the virtual printer is assigned.

Examples

To change the default page width to 132 characters (the **w** attribute) and specify that user `mary` receives the "intervention required" messages (the **si** attribute) for the virtual printer associated with the `proq` print queue and the `mypro` queue device, enter:

```
chvirprt  -q proq  -d mypro  -a si=mary w=132
```

Files

Item	Description
/etc/qconfig	Configuration file
/usr/sbin/chvirprt	Contains the **chvirprt** command.
/var/spool/lpd/pio/@local/custom/*	Virtual printer attribute files
/var/spool/lpd/pio/@local/ddi/*	Digested virtual printer attribute files.

chvmode Command

Purpose

Changes the current output device and viewport size of the X server.

Note: This command is usable only while the X server is running.

Syntax

chvmode [{ **+** | **-** } **l**] [{ **+** | **-** } **c**] [**-vsize** *Width***x***Height* [@ VSync]

Description

The **chvmode** command changes the current output device and viewport size used by the X server.

Viewport size specification is usable only for a CRT display and its resolution has panning option.

You could also use the System Management Interface Tool (SMIT) to run this command.

Flags

Item	Description
+/-c	Enables or disables CRT output.
+/-l	Enables or disables LCD output.
-vsize *Width***x***Height*[@VSync]	Specifies viewport size of CRT display and the vertical synchronization (refresh rate in Hz). If **@***VSync* is not specified, the current vertical synchronization frequency is used.

Security

Access Control: Any User

Auditing Events: None

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To disable the LCD panel and enable the CRT display, enter:

```
chvmode -l +c
```

2. To change the current CRT viewport to be 1024x768, enter:

```
chvmode -vsize 1024x768
```

3. To specify VGA mode with high refresh rate of 75Hz, enter:

```
chvmode -vsize 640x480@75
```

Files

Item	Description
/usr/bin/X11/chvmode	Contains the **chvmode** command.

chwpar Command

Purpose

Changes the characteristics of a workload partition.

Syntax

/usr/sbin/chwpar [**-a**] [**-b** *bootset id* [,bootset id, ...] [**-B** [*attribute=value* ...]]... [**-A**] [**-c**] [**-d** *directory*] [**-D** *attribute=value* ...] ... [**-F**] [**-h** *hostname*] [**-i**] [**-I** *attribute=value* ...] ... [**-n** *newname*] [**-H** [**-M** *attribute=value* ...][**-N** *attribute=value* ...] ... [**-P**] [**-R** *attribute=value* ...] [**-S** *attribute*[+|-]=*value*...] [**-u** *userscript*] [**-x**][**-U [uuid]]** [**-v**] [**-X attribute=value ...**]wparname

/usr/sbin/chwpar -K [**-A**] [**-b**][**-B** *bootset=bootset id*] [**-c**] [**-D devname**=*devicepathname*] ... [**-F**] [**-i**] [**-I rtdest**=*destination* **rtgateway**=*gateway* [*attribute=value* ...]] ... [**-M** *attribute=value* ...][**-N address**=*A.B.C.D*] ... [**-R** [*attribute* ...]] [**-S**] [**-u**] [**-x**] [**-v**] [**-X kext=value ...**]*wparname*

Note: Regardless of locale, only ASCII characters are allowed as arguments to **mkwpar**, **chwpar**, or **wparexec**

In addition to this, there are more restrictions for a WPAR's name:

- May not be more than 25 bytes.
- May not contain whitespace or any of the following symbols

```
= : / ! ; ` ' " < > ~ & ( ) * + [ ] , . ^ 0 { } | \
```

- May not start with **'-'** or **'0'**.

Description

The **chwpar** command modifies the configuration options of the workload partition specified by the *wparname* parameter. You can change most options whether the workload partition is running. Some

changes to the running workload partitions are detected and disallowed (see the **-d** and **-n** options). Other changes, such as unexporting a busy device or removing a mounted file system, might generate errors on a running workload partition, but you can make these changes.

Use the **-K** flag to remove characteristics from the configuration of a workload partition. For an attribute with a default option, removing the value for the attribute restores the default setting for the option.

WPAR does not support all types of CD ROM devices. It supports only CSI CD ROM devices using FCP (the subclass type). However, the Integrated Drive Electronics (IDE), Serial ATA (SATA), and the virtual devices (exported from a Virtual I/O Server) are not supported.

Flags

-a
Automatically resolves conflicting static settings if required. Settings that can be resolved are hostname and network configuration.

-A
Configures the workload partition to be started at system boot through the **/etc/rc.wpars** command by setting the **auto** attribute value of the workload partition to **yes**. When you specify the **-A** flag with the **-K** flag, the **auto** attribute value is set to **no**. The **-A** flag takes effect the next time the global system boots. The **-A** flag is not valid for application workload partitions.

-b bootset id[,bootset id, ...]
Configures a bootlist for the workload partition. The bootlist determines which bootset is used when the workload partition is started next time. At least one bootset ID must be specified. If a bootlist is not configured, the workload partition is started by using the same bootset that was used previously. If multiple bootset IDs are configured and if starting from the first bootset fails, the second listed bootset is used to start the workload partition, and so on.

-B {{devname=device name | devid=device identifier} [bootset=bootset id] | [bootset=bootset id] [vg=volume group name]}
Creates an alternate bootset for a workload partition. An alternate bootset is a clone of the current rootvg for a RootVG WPAR or of WPAR file systems for a non-RootVG WPAR. The valid attributes for a RootVG workload partition are: devname or devid, and bootset. The bootset and VG attributes are valid for non-RootVG workload partition.

devname=device name
Specifies the logical device short name of the storage device to clone the current rootvg.

devid=device identifier
Specifies the unique device identifier of a disk type device to clone the current rootvg.

bootset=bootset id
Specifies an integer identifier that is assigned to the new alternate bootset. The valid values are in the range 0 - 8. If the bootset ID is not specified, the next available ID is used.

vg=volume group name
Specifies the name of a volume group to create the alternate bootset file systems on a non-RootVG WPAR. If the volume group is not specified, the file systems are created on the same volume group where the currently active WPAR file systems reside.

-c
The workload partition is enabled for checkpoint.

Note: The capability to enable a workload partition for checkpoint depends on additional software.

-d *directory*
Changes the base directory for the workload partition. The **-d** flag can not be used on a running workload partition. This flag is not valid for application workload partitions. The base directory of a workload partition cannot be changed if it has alternate bootsets defined.

-D {devname=*device name* | devid=*device identifier*}[rootvg=*yes* | *no*] [devtype=[*clone* | *pseudo* | *adapter* |*disk* | *cdrom* | *tape*]]

Configures exporting or virtualization of a global device into the workload partition every time the system starts. You can specify more than one **-D** flag to allocate multiple devices. Separate the attribute=value by blank spaces. You can specify the following attributes for the **-D** flag:

devname=device name
Specifies the device name to allocate to the workload partition. For pseudo and clone type devices, this is the full path to the device (i.e. /dev/pty10). For storage type devices, it is the logical device short name.

devid=device identifier
Specifies the unique device identifier of a disk type device to allocate to the workload partition. This attribute only applies to disk, cdrom, or tape type devices.

rootvg= [yes | no]
Used to indicate if the specified disk device is to be used as a rootvg workload partition device. If the **rootvg** attribute is not specified, the command will take the default of no.

devtype=[clone | pseudo | adapter | disk | cdrom | tape]
Specifies the device type of the device to allocate to the workload partition.

-F
Suppresses failures due to settings that are not valid.

-h *hostname*
Modifies the kernel host name of the workload partition.

-H *architecture*
Changes or removes the architecture of a workload partition. The valid architecture values are: [pwr4 |ppc970|pwr5|pwr6|pwr7|pwr8]. The special value, none disables the compatibility of the workload partition architecture. The **-H** flag can not be used on a running workload partition.

Note: The **-H** flag is valid along with the **-K** flag.

-i
Enables WPAR-specific routing for the workload partition. When WPAR-specific routing is enabled on a running workload partition, any explicit routing table entries that were configured using the **-I** flag with the **mkwpar**, **wparexec**, or **chwpar** command are added to the routing table of the workload partition. Running the **chwpar -i** command on a workload partition with enabled WPAR-specific routing refreshes the routing table of the workload partition. You can use the **-i** flag, for example, to restore the routing table after a global route flush. You can use the **-i** flag with the **-K** flag to disable WPAR-specific routing for the workload partition. For more information about the **-i** flag, see the description of the **-i** flag of the **mkwpar** command.

-I *attribute=value ...*
Modifies explicit routing table entries. Entries are matched based on the combination of the **rtdest**, **rtgateway**, and **rtinterface** (if specified) attributes. If a matching entry is found, the remaining attributes are used to update that routing table entry. If no match is found, a new entry is created in the workload partition routing table. For more information about the **-I** flag, see the description of the **-i** flag and the **-I** flag of the **mkwpar** command. However, unlike the **mkwpar** command or the **wparexec** command, using the **-I** flag with the **chwpar** command does not change whether WPAR-specific routing is enabled or disabled. Use the **-i** flag (with or without the **-K** flag) to disable or enable WPAR-specific routing.

You can specify the following attributes with the **-I** flag:

rtdest=*destination*
(Required) Identifies the host or network to which you are directing the route. You can specify the value using either a symbolic name or a numeric address. You can use the keyword **default** to specify a default route. For more information about the **rtdest** attribute, see the *Destination* parameter of the **route** command.

rtgateway=*gateway*
(Required) Identifies the gateway to which packets are addressed. You can specify the value using either a symbolic name or a numeric address.

rtnetmask=*A.B.C.D*
Specifies the network mask to the destination address.

rtprefixlen=*n*
Specifies the length of a destination prefix, which is the number of bits in the netmask. The value must be a positive integer.

rttype={*net*|*host*}
Forces the **rtdest** attribute to be interpreted as the specified type.

rtinterface=*if*
Specifies the interface, for example, en0, to associate with the route so that packets are sent using the interface when the route is chosen.

rtfamily={*inet*|*inet6*}
Specifies the address family. For information about the parameters of the **rtfamily** flag, see the parameter section of the ../r_commands/route.html **route** command.

-M dev=devicepath directory=dir vfs=type [mountopts=mountopts]
Specifies a **namefs** (*vfs=namefs*) mount, which can be accessed from the workload partition. You can specify more than one **M** flag. The only workload partition mount form allowed here is: **namefs**.

Specifies that the global directory that is specified by the **dev** attribute is mounted over the directory that is specified by the **directory** attribute in the file system structure of the workload partition. The only other attribute that is applicable to a **namefs** mount is *mountopts*. B using the **-M** flag in the **chwpar** command, the existing directories in the workload partition cannot be mapped. The **namefs** mount can also be used with the **rootvg** workload partition. In this case, the content of the mount will not be saved by the **savewpar** command. You can use the **M** flag with the **K** flag to remove a **namefs** mount from the workload partition, but the **/, /var, /opt, /usr, /tmp, /proc or /etc/objrepos/wboot** file system of a workload partition cannot be removed.

-K
Deletes the specified attributes from the configuration of the workload partition. You can use the **-K** flag with the following flags:

-A
Changes the general **auto** option value of the workload partition to **no**, causing the workload partition not to be started when the **/etc/rc.wpars** command is running. This flag is not valid for application workload partitions.

-b
Deletes the currently configured bootlist.

-B bootset=bootset id
Deletes the specified alternate bootset from the workload partition.

-c
The workload partition is disabled for checkpoint.

-D [devname=*device name* | devid=*device identifier*]
Removes an explicit entry concerning an exported device, causing either a device that is not exported previously to be exported, or a previously exported device to be removed. This flag is not valid for application workload partitions.

You can specify the following attributes for the **-D** flag:

devname=device name
Specifies the device name to allocate to the workload partition. For pseudo and clone type devices, this is the full path to the device (i.e. /dev/pty10). For storage type devices, it is the logical device short name.

devid=device identifier
Specifies the unique device identifier of a disk type device to allocate to the workload partition. This attribute only applies to disk, cdrom, or tape type devices. When the **devid** attribute is used, the **devtype** attribute must also be specified.

-X [kext=/path/to/extension|ALL]
Removes an explicit entry for an exported kernel extension. Removing a kernel extension will prevent it from being loaded inside a workload partition. If the kernel extension is loaded inside a workload partition, the kernel extension will not be unloaded. A restart of the workload partition will be required to completely unexport the kernel extension from the workload partition. This flag is not valid for application workload partitions. The following attribute must be specified:

kext=/path/to/extension|ALL
Specify the kernel extension to remove. This must match a value inside the workload partition's configuration file. This must either be a fully qualified path or ***ALL*** if the **kext**=***ALL*** had previously been used.

Deletes the specified attributes from the configuration of the workload partition. You can use the **-K** flag with the following flags:

-i
Disables WPAR-specific routing for the workload partition. Any explicit routing table directives that are supplied using the **-I** flag with the **mkwpar**, **wparexec**, or **chwpar** command are maintained (but inactive) in the configuration of the workload partition. The explicit entries are created automatically the next time WPAR-specific routing is enabled.

-I rtdest=*destination* rtgateway=*gateway* [*attribute*=*value* ...]
Removes an explicit entry from the routing table of the workload partition. You must specify at least the **rtdest** attribute and the **rtgateway** attribute to identify the entry to be deleted.

-N address=*A.B.C.D*
Removes the specified IPv4 address from the configuration of the workload partition.

-N address6=*S*:*T*:*U*:*V*:*W*:*X*:*Y*:*Z*
Removes the specified IPv6 address from the configuration of the workload partition.

-R [*attribute* ...]
Removes specific fields from the resource control configuration of the workload partition. The **-R** flag can restore each field to its default state. For fields such as **totalProcesses**, the default state is **unlimited**. The following attributes can be restored to the default handling:

- **rset**
- **shares_CPU**
- **CPU**
- **shares_memory**
- **memory**
- **procVirtMem**
- **totalVirtMem**
- **totalProcesses**
- **totalThreads**
- **totalPTYs**
- **totalLargePages**
- **pct_msgIDs**
- **pct_semIDs**
- **pct_shmIDs**
- **pct_pinMem**

When no attributes are specified, the **-K** and **-R** flags restore the resource control profile of the workload partition to its default settings.

-S
Restores the security settings for the workload partition to default values.

-u
Disables the callout to the user script on administration events. (It not delete the script itself.)

-x
Disallows access to the cross-WPAR semaphores and shared memory segments.

M *directory=dir*
Removes the **namefs** mount specified by the directory attribute from the workload partition.

Note: The **/**, **/var**, **/opt** ,**/usr** , **/tmp**, **/proc**, or **/etc/objrepos/wboot** file system of a workload partition cannot be removed.

-n *newname*
The new name for the workload partition. Do not specify the **-n** flag for a running workload partition.

-N *attribute=value ...*
Modifies the network configuration attributes. Entries are matched based on the **address** or **address6** attribute. Each entry must be specified per **-N** flag. You can specify more than one **-N** flags to reconfigure multiple IP addresses. You can modify the following network configuration attributes:

- **interface**=*if* or **interface**=*namemappedif*
- **address**=*A.B.C.D*
- **netmask**=*A.B.C.D*
- **broadcast**=*A.B.C.D*
- **address6**=*S*:*T*:*U*:*V*:*W*:*X*:*Y*:*Z*
- **prefixlen**=*n*

The value of the **prefixlen** attribute ranges from 0 through 128.

The name-mapped interface is in the **/etc/wpars/devmap** file. You can specify the mapping between the name-mapped interface and the system interface as follows:

```
# The comments start with '#'
# Each line contains a pair of name-mapped interface
# and real interface separated by tab or blank spaces.
foo en0
goo en1
soo en2
```

-P
Interactively sets the password for the root user in the workload partition. This flag is not valid for application workload partitions.

-R *attribute=value ...*
Allows modification of resource control attributes. Most resource controls are similar to those used by Workload Manager. You can specify the following attributes:

active=yes|no
If you specify **yes**, this attribute allows resource control definitions to be retained, but they are rendered inactive. If you specify **no**, performance metrics such as processor and memory usage might not be available through commands such as the **topas** and **wlmstat** commands, both inside and outside of the workload partition.

rset=*rset*
Configures the workload partition to use a resource set that is created by the **mkrset** command.

shares_CPU=*n*
Specifies the number of processor shares that are available to the workload partition. See Workload Manager shares File.

CPU=*m*%-*SM*%,*HM*%
Specifies the percentage processor limits for the processes of the workload partition. See Workload Manager limits File.

shares_memory=*n*
Specifies the number of memory shares that are available to the workload partition. See Workload Manager shares File.

memory=*m*%-*SM*%,*HM*%
Specifies the percentage memory limits for the processes of the workload partition. See Workload Manager limits File.

procVirtMem=*n*[M|MB|G|GB|T|TB]
Specifies the maximum amount of virtual memory that a single process can consume. Processes that exceed the specified limit are terminated. The valid units are megabytes (M or MB), gigabytes (G or GB), and terabytes (T or TB). The minimum limit allowed is 1MB. The maximum limit that can be specified is 8796093022207M, 8589934591G, or 8388607T. If you set the value to -1 (no units), the limit is disabled. See workload partition limits File.

totalVirtMem=*n*[M|MB|G|GB|T|TB]
Specifies the maximum amount of virtual memory that can be consumed by the WPAR as a whole. Processes that cause the specified limit to be exceeded will be terminated. The valid range and units are the same as for procVirtMem. If the value is set to -1 (no units), the limit is disabled. See workload partition limits File.

totalProcesses=*n*
Specifies the total number of processes that are allowed in the workload partition. See workload partition limits File.

totalPTYs=*n*
Specifies the total number of pseudo terminals that are allowed in the workload partition. See **pty Special File**.

totalLargePages=*n*
Specifies the number of large pages that are allowed for the workload partition. See **Large Pages**.

pct_msgIDs=*n*%
Specifies the percentage of the maximum number of message queue IDs of the system that are allowed in the workload partition. See **Message Queue Kernel Services**.

pct_semIDs=*n*%
Specifies the percentage of the maximum number of semaphore IDs of the system that are allowed in the workload partition.

pct_shmIDs=*n*%
Specifies the percentage of the maximum number of shared memory IDs of the system that are allowed in the workload partition. See **Shared Memory**.

pct_pinMem=*n*%
Specifies the percentage of the maximum pinned memory of the system that can be allocated to the workload partition. See **Support for pinned memory**.

totalThreads=*n*
Specifies the total number of threads that are allowed in the workload partition.

-S *attribute*[+|-]=*value*...
Modifies the security settings for the workload partition. You can specify only one of the following forms of security changes:

secfile=*secAttrsFile*
Sets the privileges for the workload partition to the privileges listed in the specified file.

privs=*priv,priv,...*
Sets the privileges for the workload partition to the specified list of privileges.

privs+=*priv,priv,...*
Adds the specified list of privileges to the privilege set for the workload partition.

privs-=*priv,priv,...*
Removes the specified list of privileges from the privilege set for the workload partition.

Important: Do not change the security settings when a workload partition is active.

-u *userscript*
Changes the path to the user script to be run on workload partition administration events. If no user script was configured, the specified script is added to the configuration. An RBAC user cannot run this flag for a WPAR that others own.

-U [Workload Partition UUID]
Changes the Workload Partition UUID. If not given, a new UUID is automatically generated.

-v
Enables verbose output.

-x
Allows access to the cross-WPAR semaphores and shared memory segments.

-X [exportfile=/path/to/file | [kext=[/path/to/extension|ALL]] [local=yes|no] [major=yes|no]
Configures exporting kernel extensions that will be allowed to load inside a workload partition. You can specify more than one **-X** flag to allocate multiple kernel extensions. Separate the **attribute**=value by blank spaces. This flag is not valid for application workload partitions. You can specify the following attributes for the**-X** flag:

exportfile=/path/to/file
Specify a file containing valid extension stanza that will be exported. An extension stanza should contain at least the **kext** attribute. The **local** and **major** attribute can also be specified in the stanza which are described below. The **exportfile** attribute is mutually exclusive with the **kext** attribute. It is also mutually exclusive with the local and major attribute because these can be specified for each extension stanza in the **exportfile**. This is a file that can be created by a user to use with **exportfile**=/path/to/file for **mkwpar** and **chwpar**. It can contain multiple extension stanzas. The **kext** attribute is required for each extension stanza. The local and major are optional as they both have default values of **no**. The **exportfile** will look similar to the following:

```
extension:
        major = "yes"
        local = "no"
        kext = "/usr/lib/drivers/ldterm"
```

kext=/path/to/extension
Specify a kernel extension that will be exported. This is a kernel extension located in the global system's file system. The keyword ***ALL*** can also be specified. This will allow a workload partition to load any extension. When ALL is specified, the **local** and **major** attributes are restricted to **local**=yes and **major**=no. Additional **-X** flags can be specified to override the restricted **local** and **major** values. The **kext** attribute is mutually exclusive with the **exportfile** attribute.

local=yes|no
Specifying **local**=yes will make an instance of the kernel extension accessible to only the workload partition that is loading it. Specifying **local**=no will share the instance of the kernel extension loaded in the global system. By default, **local**=no.

major=yes|no
This attribute should only be used for kernel extensions that have an associated device major. By default, **major**=no.

recalc=yes
This attribute can be used to recalculate the checksum of the kernel extension.

Parameters

Item	Description
wparname	The name of the system or application workload partition to be changed. The *wparname* parameter must be the last parameter on a command line.

Security

Access Control

Only the root user can run the command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To modify the host name of the workload partition called `roy`, enter the following command:

   ```
   chwpar -h roy.com roy
   ```

2. To remove a network address from the workload partition called `dale`, enter the following command:

   ```
   chwpar -K -N address=219.81.45.65 dale
   ```

3. To disable resource controls in the workload partition called `wayne` while retaining the settings for future use, enter the following command:

   ```
   chwpar -R active=no wayne
   ```

4. To modify the **bootlist** attribute of a workload partition to the ordered list of bootset 1, bootset 2, and bootset 3, enter the following command:

   ```
   chwpar -b bootlist=1,2,3 <wpar name>
   ```

5. To create a **bootset** that consists of hdisk3 and hdisk4 to a RootVG workload partition, enter the following command:

   ```
   chwpar -B devname=hdisk3 -B devname=hdisk4 <wpar name>
   ```

6. To create a **bootset** that consists of hdisk3 and hdisk4 with the bootset ID 3 for a RootVG workload partition, enter the following command:

   ```
   chwpar -B devname=hdisk3 bootset=3 -B devname=hdisk4 bootset=3 <wpar name>
   ```

7. To create a **bootset** to a non RootVG workload partition, enter the following command:

   ```
   chwpar -B <wpar name>
   ```

8. To create a **bootset** on a certain volume group for a non RootVG workload partition, enter the following command:

   ```
   chwpar -B vg=<volume group> <wpar name>
   ```

9. To create a **bootset** on a certain volume group with the bootset ID 5 for a non RootVG workload partition, enter the following command:

   ```
   chwpar -B bootset=5 vg=<volume group> <wpar name>
   ```

10. To remove a **bootset** whose bootset ID is 3 from a workload partition, enter the following command:

    ```
    chwpar -K -B bootset=3 <wpar name>
    ```

11. To unexport a device from a workload partition, enter the following command:

    ```
    chwpar -K -D devname=hdisk1 <wpar name>
    ```

12. To export a device, enter the following command:

    ```
    chwpar -D devname=hdisk1 devtype=disk <wpar name>
    ```

13. To rename the workload partition from `moore` to `hart`, enter the following command:

```
chwpar -n hart moore
```

14. To add an adapter, fcs2, to a workload partition named 'roy', enter the following command:

```
chwpar -D devname=fcs2 roy
```

15. To remove an adapter, fcs2, from a workload partition named 'roy', enter the following command:

```
chwpar -K -D devname=fcs2 roy
```

Files

Item	Description
/etc/wpars/devexports	Default device export control file for workload partitions.

chypdom Command

Purpose

Changes the current domain name of the system.

Syntax

/usr/sbin/chypdom [**-I** | **-B** | **-N**] *DomainName*

Description

The **chypdom** command will change the domain name of the system. The *DomainName* parameter specifies the new domain name for the system.

You could also use the System Management Interface Tool (SMIT) **smit chypdom** fast path to run this command.

Flags

Item	Description
-I	Specifies that the domain name should be changed in the **/etc/rc.nfs** file. With this flag, the domain name will be changed on the next system restart.
-B	Specifies that the domain name should be changed now and the **/etc/rc.nfs** file should be updated to reflect the change.
-N	Specifies that the domain name should be changed now. No change is made to the **/etc/rc.nfs** file. The **domainname** command is executed to change the domain name of the system.

Examples

To modify the **/etc/rc.nfs** file to set the domain name to `mydomain` on the next system restart, enter:

```
chypdom -I mydomain
```

Files

Item	Description
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.

ckauth Command

Purpose

Checks the current user session for an authorization.

Syntax

ckauth [**-A**] { *AuthName* [,*AuthName*] ... }

Description

The **ckauth** command determines whether the process that the **ckauth** command is invoked in has the authorizations specified by the *AuthName* parameter. The command is used in shell scripts that need to check for authorizations. With the **ckauth** command, you can specify a single authorization or multiple authorizations through a comma-separated list. The **ckauth** command returns 0 when the calling process has any of the listed authorizations. If you specify the **-A** option, the **ckauth** command returns 0 when the calling process has all of the listed authorizations. A nonzero value is returned for failures.

Flags

Item	Description
-A	Checks whether the calling process has all of the listed authorizations.

Examples

1. To determine whether the existing user session has the `aix.fs.manage` authorization, use the following command:

```
$ ckauth aix.fs.manage
$ echo $?
0
```

2. To determine whether the existing user session has both the `aix.security.user` and `aix.security.group` authorizations, use the following command:

```
$ ckauth -A aix.security.user,aix.security.group
$ echo $?
0
```

ckfilt Command

Purpose

Checks the syntax of filter rules.

Syntax

`ckfilt` [`-O`] [`-v` 4 | 6]

Description

The `ckfilt` command checks the syntax of the filter rules. IPsec stateful filter rules allow for actions such as IF, ELSE and ENDIF. Thus it is possible to have syntax errors in the rules set, such as IF with out and ENDIF, or an ELSE or ENDIF with out a preceding IF. The `ckfilt` command checks for such errors. Nesting of IF rules is permitted. The `ckfilt` command displays the filter rules, indenting the rules within IF statements in a scoping fashion. If the `-0` flag is used, filter rules and all of their attributes are displayed in a scoped fashion. IPsec filter rules for this command can be configured using the `genfilt` command, IPsec smit (IP version 4 or IP version 6) in the Virtual Private Network submenu.

Flags

Item	Description
`-0`	Displays filter rule attributes.
`-v 4 | 6`	Specifies IPv4 or IPv6.

Exit Status

This command returns the following exit values:

Item	Description
`0`	The command completed successfully.
`non-zero`	An error occurred.

Security

This command is only executable by root.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To create a set of nested if-else-endif filter rules, use the `genfilt` command as follows:

```
genfilt -v4 -a I -s 192.168.100.101
-d 192.168.100.102 -c tcp -O eq -P 21 -D "IF ftp-cmd being used"

genfilt -v4 -a I -s 192.168.100.101
-d 192.168.100.102 -c tcp -O eq -P 1525 -D "IF 1525 port starts being used"

genfilt -v4 -a D -s 192.168.100.101
-d 192.168.100.102 -c tcp -O eq -P 37 -D "if scope: deny time"

genfilt -v4 -a L -s 192.168.100.101
-d 192.168.100.102 -c tcp -D "ELSE"

genfilt -v4 -a D -s 192.168.100.101
-d 192.168.100.102 -c tcp -O eq -P 13 -D "else scope: deny date"

genfilt -v4 -a E -s 192.168.100.101
-d 192.168.100.102 -c tcp -D "ENDIF"

genfilt -v4 -a L -s 192.168.100.101
-d 192.168.100.102 -c tcp -D "ELSE"

genfilt -v4 -a D -s 192.168.100.101
-d 192.168.100.102 -c tcp -O eq -P 20 -D "else scope: deny ftp-data"

genfilt -v4 -a E -s 192.168.100.101
-d 192.168.100.102 -c tcp -D "ENDIF"
```

   The output of the `lsfilt` command will look similar to the following:

```
%lsfilt -v4 -O
1|permit|0.0.0.0|0.0.0.0|0.0.0.0|0.0.0.0|no|udp|eq|4001|
eq|4001|both|both|no|all packets|0|all|0|||Default Rule

2|*** Dynamic filter placement rule for IKE tunnels ***|no

3|if|192.168.100.101|255.255.255.255|192.168.100.102|
```

```
255.255.255.255|yes|tcp|any|0|eq|21|both|both|no|all packets|0|all|0|||IF ftp-cmd being used

4|if|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|1525|both|both|no|all packets|0|all|0|||IF 1525 port starts being used

5|deny|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|37|both|both|no|all packets|0|all|0|||if scope: de ny time

6|else|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|any|0|both|both|no|all packets|0|all|0|||ELSE

7|deny|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|13|both|both|no|all packets|0|all|0|||else scope: deny date

8|endif|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|any|0|both|both|no|all packets|0|all|0|||ENDIF

9|else|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|any|0|both|both|no|all packets|0|all|0|||ELSE

10|deny|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|20|both|both|no|all packets|0|all|0|||else scope: deny ftp-data

11|endif|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|any|0|both|both|no|all packets|0|all|0|||ENDIF

0|permit|0.0.0.0|0.0.0.0|0.0.0.0|0.0.0.0|yes|all|any|0|
any|0|both|both|no|all packets|0|all|0|||Default Rule
```

The output of the `ckfilt` command will look similar to the following:

```
%ckfilt -v4
Beginning of IPv4 filter rules.
Rule 2
IF Rule 3
|    IF Rule 4
|    |    Rule 5
|    ELSE Rule 6
|    |    Rule 7
|    ENDIF Rule 8
ELSE Rule 9
|    Rule 10
ENDIF Rule 11
Rule 0
```

OR

```
%ckfilt -v4 -O
Beginning of IPv4 filter rules.
2|*** Dynamic filter placement rule for IKE tunnels ***|no
IF 3|if|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|21|both|both|no|all packets|0|all|0|||IF ftp-cmd being used

|    IF 4|if|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|1525|both|both|no|all packets|0|all|0|||IF 1525 port starts being used

|    |    5|deny|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|37|both|both|no|all packets|0|all|0|||if scope: deny time

|    ELSE 6|else|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|any|0|both|both|no|all packets|0|all|0|||ELSE

|    |    7|deny|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|13|both|both|no|all packets|0|all|0|||else scope: deny date

|    ENDIF 8|endif|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|any|0|both|both|no|all packets|0|all|0||| ENDIF

ELSE 9|else|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|any|0|both|both|no|all packets|0|all|0|||ELSE

|    10|deny|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|20|both|both|no|all packets|0|all|0|||else scope: deny ftp-data

ENDIF 11|endif|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|any|0|both|both|no|all packets|0|all|0|||ENDIF

0|all packets|0.0.0.0|0.0.0.0|0.0.0.0|0.0.0.0|no|0|???|0|???|0|?????|????????|no|???????|0||0|||
```

2. If incorrect if-else-endif rules are created, the `ckfilt` command will find and report the error as follows:

```
%lsfilt -v4 -O

1|permit|0.0.0.0|0.0.0.0|0.0.0.0|0.0.0.0|no|udp|eq|4001|eq|4001|both|both|no|all packets|0|all|0|||Default Rule

2|*** Dynamic filter placement rule for IKE tunnels ***|no
3|if|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|21|both|both|no|all packets|0|all|0|||IF ftp-cmd being used

4|deny|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|37|both|both|no|all packets|0|all|0|||if scope: deny time

5|else|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|any|0|both|both|no|all packets|0|all|0|||ELSE

6|deny|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|13|both|both|no|all packets|0|all|0|||else scope: deny date

7|endif|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|any|0|both|both|no|all packets|0|all|0|||ENDIF

8|else|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|any|0|both|both|no|all packets|0|all|0|||ELSE

9|deny|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|20|both|both|no|all packets|0|all|0|||else scope: deny ftp-data

10|endif|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|any|0|both|both|no|all packets|0|all|0|||ENDIF

0|permit|0.0.0.0|0.0.0.0|0.0.0.0|0.0.0.0|yes|all|any|0|any|0|both|both|no|all packets|0|all|0|||Default Rule
```

```
%ckfilt -v4
Beginning of IPv4 filter rules.
Rule 2
IF Rule 3
|    Rule 4
ELSE Rule 5
|    Rule 6
ENDIF Rule 7
No preceeding IF statement for filter rule 8.
The filter rules failed the syntax check.

%ckfilt -v4 -O
Beginning of IPv4 filter rules.
2|*** Dynamic filter placement rule for IKE tunnels ***|no
IF 3|if|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|21|both|both|no|all packets|0|all|0|||IF ftp-cmd being used

|    4|deny|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|37|both|both|no|all packets|0|all|0|||if scope: deny time

ELSE 5|else|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|any|0|both|both|no|all packets|0|all|0|||ELSE

|    6|deny|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|eq|13|both|both|no|all packets|0|all|0|||else scope: deny date

ENDIF 7|endif|192.168.100.101|255.255.255.255|192.168.100.102|255.255.255.255|
yes|tcp|any|0|any|0|both|both|no|all packets|0|all|0|||ENDIF


No preceeding IF statement for filter rule 8.
The filter rules failed the syntax check.
```

Location

`/usr/sbin/ckfilt`

Files

Item	Description
`/etc/security/ipsec_filter`	This command reads the `/etc/security/ipsec_filter` ODM database. Rules are inserted and changed in this database using the `genfilt` and `chfilt` commands.

ckpacct Command

Purpose

Checks data file size for process accounting.

Syntax

/usr/sbin/acct/ckpacct [*BlockSize*]

Description

The **ckpacct** command checks the size of the active data file, **/var/adm/pacct**. Normally, the **cron** daemon runs this command. If the size of the active data file exceeds the number of blocks specified by the *BlockSize* parameter, the **ckpacct** command invokes the **turnacct switch** command to turn off process accounting. The default value for the *BlockSize* parameter is 1000.

If the number of free disk blocks in the **/var** file system falls below 500, the **ckpacct** command automatically turns off process accounting by invoking the **turnacct off** command. When 500 blocks are again available, accounting is reactivated. This feature is sensitive to how frequently the **ckpacct** command is run.

When the **MAILCOM** environment variable is set to **mail root adm**, a mail message is sent both to the **root** and **adm** groups if an error occurs.

Security

Access Control: This command should grant execute (x) access only to members of the **adm** group.

Examples

To automatically check the size of the **/var/adm/pacct** data file, add the following to the **/var/spool/cron/crontabs/root** file:

```
5 * * * * /usr/sbin/acct/ckpacct
```

This example shows the instructions the **cron** daemon reads and acts upon. The **ckpacct** command runs at 5 minutes past every hour (5 *) every day. This command is only one of the accounting instructions normally given to the **cron** daemon.

Files

Item	Description
/usr/sbin/acct	The path to the accounting commands
/var/adm/pacct	Current file for process accounting.

ckprereq Command

Purpose

Verifies that all prerequisite software is available and at the appropriate revision levels.

Syntax

ckprereq [**-v**] [**-O** { **r** | **u** | **s** }] [**-f** *PrereqFile* | a**-l** *FilesetName* [*Level*]]

Description

The **ckprereq** command determines whether the system level is compatible with the software product to be installed or updated.

The **ckprereq** command is designed to be used during the installation procedures of a software product.

When **ckprereq** is invoked with the **-f** flag, the *PrereqFile* parameter specifies a software prerequisite list file. Each record in this file contains information about a prerequisite fileset needed to complete the installation procedure.

When **ckprereq** is invoked with the **-l** flag, the prerequisite information is read from the *ProductName* information in the Software Vital Product Data (SWVPD) database.

If the *PrereqFile* parameter was given with the **-f** flag, then an output file is produced by the **ckprereq** command. The output file overwrites the input file and is a listing of the original input. Any failing lines are marked with a failure code in the first column. The **ckprereq** command ignores the failure codes if an output from a previous **ckprereq** call is used as input.

There are four possible requisite tests: **prereq**, **coreq**, **ifreq**, and **instreq**.

A **prereq** is a test to check that a fileset is installed and at a specified revision level. To be considered installed, the SWVPD entry for the software product must be in the APPLIED, APPLYING, COMMITTED, or COMMITTING state. A **prereq** requires that the fileset also be at the specified revision level before installing the independent fileset.

A **coreq** test is similar to a **prereq**, except that **coreq** tests can be installed in any order, but **prereq** tests require a specific order. If a corequisite software product is not yet installed, the test is ignored and the failure codes are not set because it is assumed that the software product will be installed. The **coreq** test is ignored by the **ckprereq** command. (It is not ignored by the installp command's requisite checking procedures.)

An **ifreq** test is identical to a **coreq**, except that it tests for the revision level only if the fileset is installed. If the fileset is not installed, the **ifreq** test is ignored.

An **instreq** test is treated like a **prereq** test by the **ckprereq** command. The special meaning of instreq is only used by the up-front requisite checks of the **installp** command.

The **installp** command checks corequisite and if-requisite file sets at the completion of an install set, and returns messages for any unsatisfied **coreq** or **ifreq** conditions. An if-requisite condition would be unsatisfied if the if-requisite product is installed, but does not match the revision level specified.

Flags

Item	Description
-f *PrereqFile*	Specifies the file name of a prerequisite list file.
-l *FilesetName*[*Level*]	Specifies the name of the fileset or fileset update under which to look for the prerequisite information from the SWVPD database.
-O {**r**|**u**|**s**}	Specifies the part of the file tree of the software product that is to be checked. If this flag is not specified, the **ckprereq** command uses the value of the **INUTREE** environment variable to determine which part to check. The **INUTREE** environment variable is set by the **installp** command. The **r** option indicates the / (root) part of the software product is checked. The **u** option indicates the **/usr** part of the software product is checked. The **s** option indicates the **/usr/share** part of the software product is checked. Only one part can be checked at a time.
-v	Displays a descriptive message to standard error for each failure in the prerequisite list file.

Return Values

The **ckprereq** command tests the current version, release, modification level, fix level, and fix ID found in the SWVPD and marks the first column in each failing line in the output file with one of the following codes if the test was unsuccessful:

Item	Description
f	The test for the fix (level) was unsuccessful.
m	The test for the modification level was unsuccessful.
n	The fileset is not installed or is set to **broken**.
p	The test for the fix ID was unsuccessful.
r	The test for the release was unsuccessful.
s	There is a syntax error in the *PrereqFile* parameter.
v	The test for the version was unsuccessful.

If a serious error occurs, such as an invalid command line or a syntax error in the prerequisite list file, the return code for the **ckprereq** command is 255. Otherwise, the return code is a number that represents the number of tests that failed.

Security

Access Control

You must have root authority to run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To check that the requisite specifications in the file **/tmp/prq.test**, that has the following contents:

   ```
   *prereq bos.rte 4.1.0.0
   *prereq X11.base.rte 4.1.0.0
   ```

   are satisfied, while reporting any failures, enter:

   ```
   ckprereq -vf /tmp/prq.test
   ```

2. To check all the requisite software listed in the **/usr/lpp/snaserv/prereq2** file for the root part, enter:

   ```
   ckprereq -f /usr/lpp/snaserv/prereq2 -Or
   ```

3. To check that the requisites of the installed fileset update bos.net.tcp.client at level 4.1.0.1 are met, enter:

   ```
   ckprereq -l bos.net.tcp.client 4.1.0.1
   ```

Files

Item	Description
/etc/objrepos/product	Database containing information about the software installed in the **/root** part of the file system.
/**usr/lib/objrepos/product**	Database containing information about the software installed in the **/usr** part of the file system.
/usr/share/lib/objrepos/product	Database containing information about the software installed in the **/usr/share** part of the file system.

cksum Command

Purpose

Displays the checksum and byte count of a file.

Syntax

cksum [*File* ...]

Description

The **cksum** command reads the files specified by the *File* parameter and calculates a 32-bit checksum Cyclic Redundancy Check (CRC) and the byte count for each file. If no files are specified, the **cksum** command reads standard input. The checksum, number of bytes, and file name are written to standard output. If standard input is used, the path name and leading space are omitted.

The **cksum** command can be used to compare a suspect file copied or communicated over noisy transmission lines against an exact copy of a trusted file. The comparison made by the **cksum** command may not be cryptographically secure. However, it is unlikely that an accidentally damaged file will produce the same checksum as the original file.

The **cksum** command uses a different algorithm to calculate the 32-bit checksum CRC than the **sum** command. The **cksum** command uses a CRC algorithm based on the Ethernet standard frame check.

> **Note:** The **cksum** command is POSIX 1003.2 compliant and the checksum produced is guaranteed to be calculated the same on all POSIX 1003.2 compliant systems.

The following generating polynomial defines CRC checksum encoding:

$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

The following procedure mathematically defines the CRC value corresponding to a given file:

1. The *n* bits to be evaluated are considered to be the coefficients of a mod 2 polynomial M(x) of degree *n*-1. These *n* bits are the bits from the file. The most significant bit is the most significant bit of the first octet of the file. The last bit is the least significant bit of the last octet, padded with zero bits (if necessary) to achieve an integral number of octets, followed by one or more octets representing the length of the file as a binary value, least significant octet first. The smallest number of octets capable of representing this integer is used.
2. M(x) is multiplied by x32 (that is, shifted left 32 bits) and divided by G(x) using mod 2 division, producing a remainder R(x) of degree 31.
3. The coefficients of R(x) are considered to be a 32-bit sequence.
4. The bit sequence is complemented, and the result is the CRC.

Exit Status

This command returns the following exit values:

Item	Description
0	All files were processed successfully.
>0	An error occurred.

Examples

To display the checksum and the size, in bytes, of `file1` and `file2`, enter:

```
cksum file1 file2
```

If the checksum of the `file1` file is 3995432187 and contains 1390 bytes, and the checksum of the `file2` file is 3266927833 and contains 20912 bytes, the **cksum** command displays:

```
3995432187       1390    file1
3266927833       20912   file2
```

Files

Item	Description
/usr/bin/cksum	Contains the **cksum** command.

clcmd Command

Purpose

Takes an AIX command and distributes it to a set of nodes that are members of a cluster.

Syntax

`clcmd` [`-n` *clustername*] [`-m` *nodename* [,...]] [*File*]

Description

The AIX operating system can operate in a single node or multinode configuration. A multinode configuration of the AIX operating system is a cluster configuration.

Using the AIX system management commands (such as the `mkuser` command, `mkvg` command, and `lslv` command), a system administrator can perform operations on the characteristics and functional definitions such as devices, file systems, and user management attributes. These system management commands can be run in a local sphere or in a cluster sphere.

In a cluster configuration, running an AIX command produces a distribution of the AIX command to all nodes participating in the cluster. Thus, an AIX system administrator can manage a group of nodes as a single object.

The enablement of AIX commands for cluster awareness has the following characteristics:

- Determines the target nodes for the AIX command
- Distributes the AIX command to the target nodes

Flags

Item	Description
-n *clustername*	Specifies the name of a cluster to send a command to. All nodes in the cluster receive the command.
-m *nodename*	Specifies the node names to send a command to. The nodes must be members of a cluster. This allows the distribution of the command to a subset of nodes in a cluster.

Examples

1. To send the `ps` command to the `oscar-test-dev1` and `oscar-test-dev2` nodes in the `clusterabc` cluster, enter the following command:

```
clcmd -n clusterabc -m oscar-test-dev1,oscar-test-dev2 -- /usr/bin/ps
```

Files

Item	Description
`/path/to/localcmd` *<localcmd_options>*	A qualified file specification used to specify the command to run. The *<localcmd_options>* list contains the options relevant to the command being run.

clctrl Command

Purpose

Provides a set of system administration functions for managing a cluster.

Syntax

clctrl <*subcommand*> options

where <*subcommand*> are {**-start** | **-stop** | **-tune** | **-sec** | **-commit**}

Subcommand Syntax

To take a node offline for maintenance or bring it back online:

clctrl [**-n** *clustername*]{**-start** | **-stop**} [**-n** *clustername*]{ **-m** *node[,...]* | **-a**}

To display or set cluster tunable attribute values:

clctrl - tune -h [*tunable*]

clctrl -tune [**-n** *name* | **-u** *uuid*] (**-a** | {**-L** | **-x**} [*tunable*] | {**-o** *tunable*})

clctrl -tune [**-n** *name* | **-u** *uuid*] (**-D** | {**-d** *tunable*} | {**-o** *tunable*=value}))

To display or set security tunable values:

clctrl -**sec** { **-l** *sec_level* **-s** *sec_alg* } [**-e**] [**-t** *certificate_type* [**-c** *certificate_file* **-f** *privkey_file*]]

To manually commit a new cluster level that is effective throughout the cluster:

clctrl [**-n** *clustername*] **-commit**

Description

The **clctrl** command provides a set of subcommands for managing a cluster.

The **-stop** subcommand is used to take one or more nodes offline for maintenance. Stopping a node causes the other nodes to consider it as down. A stopped node does not send or receive heartbeat messages, and it remains in the stopped state, even across reboot operation, until a **-start** subcommand causes it to rejoin the cluster. The **-stop** subcommand can also be issued while a node is powered off to prevent it from rejoining the cluster when it is rebooted.

The **-start** subcommand is used to bring one or more nodes back online after they have been offline for maintenance. Starting a node allows it to rejoin the cluster and have the other nodes consider it as up. The **-start** subcommand can also be issued while a node is powered off to allow it to rejoin the cluster when it is rebooted.

The **-tune** subcommand is used to display or set cluster tunable values. The following flags control the **-tune** subcommand:

Item	Description
-a	Displays values for all tunables, one per line.
-D	Resets all tunables to their default values.
-d *tunable*	Resets tunable to its default value.
-h	Displays help about the command and its arguments.
-h *tunable*	Displays help about a tunable.
-L *tunable*	Lists information about one or all tunables in a table format.
-n *name*	Specifies the name of the cluster or node entity to which the tunable belongs. The name must be unique. Otherwise, the **-u** *uuid* flag must be used to identify the entity.
-o *tunable*	Displays the current value of a tunable.
-o *tunable*=value	Sets tunable to the value.

Item	Description
-u *uuid*	Specifies the UUID of the cluster or node entity. If neither the **-u** nor the **-n** options are specified, the invoking node is assumed.
-x *tunables*	Lists information about one or all tunables in a comma-separated format

The **-sec** subcommand is used to display or set security tunable values. The following flags control the **-sec** subcommand:

Item	Description
-c	Specifies the path to the certificate file for the asymmetric key.
-e	Displays values for all security tunables, one per line.
-f	Specifies the path to the private key file for the asymmetric key.
-l	Sets the security level. A value of 0 disables security; a value of 1-3 enables security and sets the level to the value. The default security level is 2.
-s	Specifies the algorithm type used to generate the symmetric key. The value may be set to AES, DES, or 3DES. Setting a value of NULL disables security if it is enabled. The default value is AES.
-t	Specifies the certificate type for the asymmetric key. The value may be set to Self Signed Certificates, Open SSL Certificates, or SSH Certificates. The default value is Self Signed Certificates.

The **-commit** subcommand manually commits a new cluster level that is effective throughout the cluster, after upgrading the CAA software levels on all nodes. The CAA software automatically commits the new cluster level. However, a system administrator might need to manually commit the new cluster level if the automatic commitment of the new cluster level fails.

Examples

1. To take a node named *fileserver1* offline for maintenance:

   ```
   clctrl -stop -n clustername -m fileserver1
   ```

2. To bring the node back online after completing maintenance:

   ```
     clctrl -start -n clustername -m fileserver1
   ```

3. To take all the nodes offline for maintenance:

   ```
   clctrl -stop -n clustername -a
   ```

4. To bring all the nodes back online after completing maintenance:

   ```
   clctrl -start -n clustername -a
   ```

5. To display information about all cluster tunables in a table format:

   ```
   clctrl -tune -L
   ```

6. To display help about tunable repos_mode:

   ```
   clctrl -tune -h repos_mode
   ```

7. To set cluster tunables value:

   ```
   clctrl -tune -o repos_mode=e
   ```

8. To display the current value of all security tunables:

```
clctrl -sec -e
```

9. To set the security algorithm used to generate the symmetric key:

```
clctrl -sec -s DES
```

10. To manually commit a new cluster level that is effective throughout the cluster:

```
clctrl -commit
```

11. To set the cluster communication mode to the unicast mode:

```
clctrl -tune -o communication_mode=u
```

12. To set the cluster communication mode to the multicast mode:

```
clctrl -tune -o communication_mode=m
```

clear Command

Purpose

Clears the terminal screen.

Syntax

clear

Description

The **clear** command clears your screen, if possible. The **clear** command first checks the **TERM** environment variable for the terminal type. Next, the **/usr/share/lib/terminfo** directory, which contains terminal definition files, is checked to determine how to clear the screen. If the **TERM** environment variable is not set, the **clear** command exits without taking any action.

Examples

To clear your screen, enter:

```
clear
```

Files

Item	Description
/usr/share/lib/terminfo	Contains terminal information database.

clffdc command

Purpose

Collects *snap data* from every node in the cluster, and stores the snap data in a single convenient cluster snapshot (csnap) compressed tar file on the node that initiated this command. The *snap data* contains the configuration information that might be required to identify and resolve system problems.

Syntax

```
clffdc -c component [-l localCorrelator] [-p priority] [-v verbosity] [-f file]
[-n lineNumber] [-g correlator] [-s]
```

Description

The **clffdc** command captures snap data from all the nodes in a Cluster Aware AIX (CAA) cluster. A cluster-wide snap operation might be triggered automatically by the operating system when a severe problem is detected. You can use the **clffdc** command to simplify snap data collection across the cluster.

The cluster-wide *snap file* is created in a default directory. For a Virtual I/O Server (VIOS) environment, the cluster-wide snap files are located in the `/home/ios/logs/ssp_ffdc` directory. For a non-VIOS environment, the cluster-wide snap files are located in the `/var/adm/ras/cl_ffdc` directory.

Each node in the cluster creates a snap file. The snap files are collected from each node and merged into a single convenient `csnap` file on the node that initiated the cluster-wide snap operation. The `csnap` file name uses the following format:

```
csnap_date_time_by_component_priority_ccorrelator.tar.gz
```

The snap file name uses the following format:

```
snap_date_time_by_component_priority_ccorrelator.tar.gz
```

Only a single cluster-wide snap operation can occur at a time. If a previous cluster-wide snap operation is in progress, a new cluster-wide snap operation cannot be initiated until the previous operation times out. Each cluster-wide snap operation is associated with a correlator value on the CAA repository disk. This correlator value increments when a new cluster-wide snap operation occurs. If the repository disk is inaccessible when a cluster-wide snap operation is initiated, a `csnap` file is not generated. In this scenario, each node generates a snap file with a time stamp, but a correlator value is not specified.

If the node that initiated the cluster-wide snap operation goes offline while the cluster-wide snap operation is in progress, each node creates a snap file but a `csnap` file is not created. If a non-initiator node goes offline while the cluster-wide snap operation is in progress, the initiator node waits for a timeout period before it captures the `csnap` file from the available nodes.

A new initiator node can collect the snap files by running the **clffdc -g** command.

The **-c**, **-f**, and **-n** flags are used to identify the location in the code that requested the snap data if the snap file was created automatically by the AIX operating system. If you manually collect the snap data, you must specify the **-c** flag to identify the component that is responsible for calling any other associated peer components during a snap collection.

Each new cluster-wide snap operation deletes the old `csnap` files and old snap files that are located in the default directory.

Flags

-c *component*
: Specifies the component that requested the cluster-wide snap operation. The *component* attribute can have the following values:

- CAA (Cluster Aware AIX)
- RSCT (Reliable Scalable Cluster Technology)
- VIOS (Virtual I/O Server)
- POOL (Shared storage pool)
- PHA (PowerHA SystemMirror)
- FULL

Note: The FULL value indicates that full snap data is collected on each node by using the **snap -a** command. Any other value indicates that a miniature snap data is collected on each node. The miniature snap data starts with the specified component and includes all peer components that are associated with that component.

-f *file*
Specifies the source file name within the component that initiated the cluster-wide snap operation. If the file name is not specified, the `clffdc.c` file name is used by default.

-g *correlator*
Gathers the cluster-wide snap files. The gathering operation collects a series of snap files that have the specified *correlator* value on each node, and brings the snap files together to create a single `csnap` file on the initiator node. The *correlator* value is specified as a decimal value. This flag is useful when used with the **-s** flag, or when a previous cluster-wide snap operation was interrupted before a `csnap` file could be generated.

Each node generates a snap file that has the specified correlator value. You can use this flag to collect the individual snap files and create a `csnap` file that represents the snap data from the entire cluster.

-l *localCorrelator*
Requests snap operation on a local node. The *localCorrelator* value is the correlator value in decimal format that is used to name the resulting snap file.

-p *priority*
Specifies the priority for the cluster-wide snap operation. The priority attribute can have the following values:

- 1 (high priority)
- 2 (medium priority)
- 3 (low priority)

The priority is used as part of the name in the resulting snap file and `csnap` file.

-n *lineNumber*
Specifies the line number of the caller who requested the cluster-wide snap operation.

-s
Initiates a staged cluster-wide snap collection. A staged collection indicates that the snap files are created on each node, but not gathered into a `csnap` file on the initiator node. This flag is useful when used with the **-g** flag, which gathers the snap files into a single `csnap` file on the initiator node.

-v verbosity
Specifies the verbosity for the cluster-wide snap operation. Possible values that can be specified with the **-v** flag are `0` or 1. You can specify 1 to collect more information for certain components during the cluster-wide snap operation.

Exit status

0
The command completed successfully.

>0
A problem occurred.

Examples

1. To collect a cluster-wide snap data that is associated with the CAA component with medium priority, enter the following command:

```
clffdc -c CAA -p 2
```

Note: In a VIOS environment, the associated components are CAA, RSCT, POOL, and VIOS. In a PowerHA environment, the associated components are CAA, RSCT, and PHA. The specified component and each associated peer component collect snap data for the cluster-wide snap operation.

2. To collect a cluster-wide snap data that contains the full snap data (collected by the `snap -a` command) with low priority, enter the following command:

```
clffdc -c FULL -p 3
```

3. To initiate a staged cluster-wide snap operation that is associated with the PHA component (PowerHA SystemMirror) with high priority, enter the following command:

```
clffdc -c PHA -p 1 -s
```

4. To gather snap files on each node with the correlator value 77 into a single convenient `csnap` file on the initiator node, enter the following command:

```
clffdc -g 77
```

Files

/usr/sbin/clffdc
Contains the **clffdc** command.

/var/adm/ras/cl_ffdc
Contains the **clffdc** command output in a non-VIOS environment.

/home/ios/logs/ssp_ffdc
Contains the **clffdc** command output in a VIOS environment.

clogin Command

Purpose

Initiates a user session or runs a command within a workload partition.

Syntax

clogin WparName [**-l** *User*] [command [args]]

Description

The **clogin** command provides a mechanism for the root user to log in or run a command within a workload partition.

Note: When you run the **clogin** command, some programs might not function properly, especially if executing in multibyte locales. Use the **clogin** command only for emergency system maintenance.

When you specify the **-l** flag, a session is initiated as if the session was started by the user specified using the *User* parameter in the workload partition. If a subsequent command is specified, the command runs as if it was launched as a parameter to the login shell associated with the *User*. The **clogin** command performs similar operations as the **su** command, so all of the functions that are associated with the **su** command apply to the **clogin** command.

Note: The pseudo-terminal on which the session is initiated belongs to the global environment, but the login shell running in the terminal belongs to the workload partition.

Flags

Item	Description
WparName	The name of the workload partition in which to log in.

Item	Description
-l *User*	Specifies the user name to log in the workload partition. Default is root. If you specify the *command* parameter, you must specify both the **-l** flag and the *User* parameter.
command	Specifies the command running within the workload partition. The command runs as a parameter to the login shell that is associated with the user.
args	Specifies optional parameters to use when the command that is specified by the *command* parameter runs.

Security

Item	Description
Access Control	Only the root user can run the command.

Examples

1. To log in to a workload partition named **bucko** as user **dan**, enter the following command:

```
clogin bucko -l dan
```

2. To run the **/usr/bin/ps** command with the **-T 1** option as user root in a workload partition named **howdy**, enter the following command:

```
clogin howdy -l root /usr/bin/ps -T 1
```

clusterconf Command

Purpose

clusterconf command is a service utility for administration of a CAA cluster configuration.

Syntax

clusterconf [**-r** *hdiskN*] [**-v**]

Description

The **clusterconf** command will allow administration of the CAA cluster configuration.

If a node in a CAA cluster configuration is showing as DOWN (can be viewed from issuing the command **"lscluster –m"**) or a node in a CAA cluster is not showing up in the CAA cluster configuration and you know the node is part of the CAA cluster configuration (can be viewed from another node in the CAA cluster with **"lscluster –m"**) the following flags will allow the node to search and read the repository disk and take self-correcting actions.

Note: The **clusterconf** command is required to create a CAA cluster.

In addition, the **inetd** daemon must be running, and the line configuring the *caa_cfg* service must be uncommented in the `/etc/inetd.conf` file. This must be true on all nodes.

Flags

If no flags are specified the **clusterconf** command will perform a refresh operation by retrieving the CAA cluster repository configuration and performing the necessary actions. Actions which may occur are for a CAA cluster node to join a CAA cluster that the node is a member of and for some reason had been disconnected from the CAA cluster (either via network or SAN problems), a CAA cluster node may perform

a resync with the CAA cluster repository configuration (again from some problems in the network or SAN) and the CAA cluster node may leave the CAA cluster configuration is the node had been removed from the CAA cluster repository configuration. The **clusterconf** command is a normal CAA cluster service and is automatically handled during normal operation.

Item	Description
-r *hdiskN*	If you know where the repository disk is (lspv and look for cvg) use this option to have the CAA cluster subsystem read the repository device and if this node is configured in the repository disk this command will cause the node to join the CAA cluster.
-v	Specify verbose mode.

Examples

1. To recover the CAA cluster configuration and start CAA cluster services:

```
clusterconf –r hdisk1
```

Files

Item	Description
/etc/inetd.conf	Contains configuration information for the **inetd** daemon.
/usr/sbin/clusterconf	Contains the **clusterconf** command.

clsnmp Command

Purpose

TheAIX **clsnmp** command provides the SNMP manager function from the AIX shell to query SNMP agents for network management information.

Syntax

clsnmp [**-d** *DebugLevel*] [**-h** *TargetHost*] [**-c** *Community*] [**-t** *TimeOutValue*] [**-r** *RetryNumber*] [**-n** *NonRepeaters*] [**-m** *MaxRepetitions*] [**-p** *PortNumber*] [**-v**] [**-f** *ConfigurationFile*] [**-?**] *Function* [*MIBVariable* [*VariableType*] [*Value*] [...]]

Description

Use the **clsnmp** command to issue SNMP requests to agents and to process SNMP responses returned by agents. The AIX **clsnmp** commands supports issuance of SNMPv1, SNMPv2c, and SNMPv3 requests.

SNMP request types

findname
Sends a request that a search be done to obtain the textual name, for a given *MIBVariable* input, whose internal ASN.1 value best matches the input ASN.1 value. The search first checks the **/etc/mib.defs** file, and if a matching textual name is not found, continue with the compiled MIB. Only one *MIBVariable* is allowed per **clsnmp findname** invocation.

get
Sends a request to an SNMP agent for a specific management information base (MIB) variable. **clsnmp** then waits for a response or times out.

getbulk
Obtains the value of the variables in the MIB tree specified by the OID or MIB variable name. A single **getbulk** performs the same function as a series of **getnext**s with fewer data exchanges between the **clsnmp** command and the SNMP agent.

getnext
Sends a request to an SNMP agent for the next MIB variable that lexicographically follows the *MIBVariable* specified. **clsnmp** then waits for a response or times out.

set
Sends a request to an SNMP agent to set a specific MIB variable. **clsnmp** then waits for a response or times out.

trap
Listens for SNMP traps and displays **trap** information when they occur. Uses the default, well-known port 162 or the port number specified on the -p option. The **clsnmp trap** function continues to listen for traps until the process is killed or canceled.

walk
Issues a **getnext** request for a specified prefix, then continues to issue **getnext** requests for as long as there are variables that match the specified prefix. A prefix can be any leading portion of the complete object identifier.

Usage

The **set** operation is not supported on all MIB objects. The **set** operation may be rejected if the agent or subagents managing the MIB object does not support SET.

getbulk is an SNMPv2 function. If the target agent only supports SNMPv1, the target agent ignores your request. As a result, your request times out.

The function keywords are not case sensitive. The flags, variable names and values are case sensitive.

In order to listen to traps from NetView® SNMP and AIX **clsnmp** at the same time, use the **-p** *PortNumber* parameter on the **clsnmp** command. Only one management application at an IP address can listen on a port at a time. Specifying **-p** on the **clsnmp** trap command enables a port other than well-known port 162 to be used. Both ports must be configured as agent trap destinations.

An **clsnmp** command that is not authenticated (by using an acceptable community name or user name) will time out.

The **clsnmp** command uses two configuration files: **/etc/mib.defs** and **clsnmp.conf**. Sample files are shipped in the **/usr/samples/snmpdv3** directory.

The clsnmp command supports sending SNMPv1, SNMPv2c, and SNMPv3 requests. The file clsnmp uses to determine whether it should send an SNMPv1,SNMPv2c or SNMPv3 request is the **clsnmp.conf** file. If the target specified by way of the -h parameter matches a winSNMP name in the **clsnmp.conf** file, **clsnmp** sends the request using the parameters specified on the entry. If the **-h** parameter is not specified, then the request will be sent as an SNMPv1request.

Flags

Item	Description
-c *Community*	Specifies the community name used to access the specified variables at the destination SNMP agent. If you do not specify a community name, the default name is public. Community names are not required when using the user-based security model. **Note:** Community names are case sensitive.

Item	Description
-d *DebugLevel*	Specifies the debug level. The default level is 0, which means no debug. The higher the debug level, the greater the number of messages that are displayed. The debug levels are 0-4.
-f *ConfigurationFile*	Specifies the full path and file name of the configuration file.
-h *TargetHost*	Specifies the target host to which you want to send a request. The host can be an IPv4 address, an IPv6 address, a host name, or a winSNMP name in the **clsnmp.conf** configuration file. If you do not specify a host, the default is your local host.
-m *MaxRepetitions*	Only applies to **getbulk**. This is ignored if the function request is not a **getbulk**. Maximum repetitions is the number of lexicographic successors to be returned for each variable binding pair after the first "**-n** number" successors. For example, starting with successor "**-n** number"+1, return "**-m** number" of successors for each variable binding pair. The default is 10.
-n *NonRepeaters*	Only applies to **getbulk** requests. This is ignored if the function request is not a **getbulk**. *NonRepeaters* is the number of variable binding pairs (name/value), starting with the first, for which only a single successor is returned. The default is 0.
-p *PortNumber*	Specifies the number of the port that listens for traps. If a port number is not specified, the **clsnmp trap** function listens on the well-known port 162, the default port for **clsnmp** traps.
-r *RetryNumber*	Specifies the maximum number of times to retry the command if it timed out. The default is 2.
-t *TimeOutValue*	Specifies the amount of time (in seconds) that the **clsnmp** command waits for a reply from the SNMP agent. The default is 3.
-v	Specifies that the output from a request should be displayed using verbose output, for example, using the textual name instead of the MIB object identifier.
-?	Displays help information.

Parameters

Item	Description
Function	Specifies the SNMP function/operation to perform, which is one of the following: **get**, **getnext**, **getbulk**, **set**, **walk**, **trap**, **findname**.

Item	Description
MIBVariable	Specifies the Management Information Base (MIB) object, using its object descriptor (textual name), object identifier in ASN.1 notation, or a combination of the two. When used with **walk**, this is the MIB object prefix. A prefix can be any leading portion of the complete object identifier. When used with **findname**, this is the object identifier in ASN.1 notation.
Value	Specifies the value to be set by the SET function. If white space is needed in the value, you must enclose the value in double quotes ("). If you want to set a variable to a value that is also a type, you must specify the type.
VariableType	Specifies the type of value being set. To complete an SNMP SET request, the SMI_type must be known. If no type is specified, **clsnmp** searches first the **/etc/mib.defs** file and then the compiled MIB to determine the type. If the variable is not found, an error is returned. If a VariableType is specified, the VariableType takes precedence over any type that may be assigned in the MIB. The VariableType and value must be compatible. For example, if you specify a type of "number" and a value of "foo," an error is returned because "foo" is not a number. *VariableType* is not case sensitive. Valid variable types are: • bitstring • counter • counter32 • counter64 • display or displaystring • gauge • gauge32 • integer • integer32 • ipaddress • nsapaddress • null • objectidentifier or OID • octetstring • opaque • opaqueascii • timeticks • uinteger

Limitation

When the **snmpdv3** daemon encounters SMI-v2 data type MIB while processing a **SNMPv1** protocol request from the **clsnmp** manager, it skips the MIB until it finds a SMI-v1 data type MIB.

Work around

The **clsnmp** manager should be configured with **SNMPv2** type requests or **SNMPv3** type requests to dump all of the MIB variables with the **snmpdv3** daemon.

Examples

1. Getting the MIB variable.
   a. The following requests MIB object `sysName.0`:

      ```
      clsnmp get sysName.0
      ```

      The output from this command looks similar to:

      ```
      1.3.6.1.2.1.1.5.0 = hostname.austin.ibm.com
      ```

   b. The following requests MIB object `myName.0`, where `myName` is defined in the **/etc/mib.defs** file to be the same object identified by `sysName.0`:

      ```
      clsnmp get myName.0
      ```

      The output from this commands looks similar to:

      ```
      1.3.6.1.2.1.1.5.0 = myhostname.austin.ibm.com
      ```

   c. The following requests MIB object `sysName.0` through an IPv6 address:

      ```
      clsnmp -h 2000:1:1:1:209:6bff:feae:6d67 get sysName.0
      ```

      The output from this command looks similar to:

      ```
      1.3.6.1.2.1.1.5.0 = hostname.austin.ibm.com
      ```

2. Getting the next MIB variable.
   a. The following requests the next logical MIB object:

      ```
      clsnmp getnext udp
      ```

      The output from this command looks similar to:

      ```
      1.3.6.1.2.1.7.1.0 = 653
      ```

   b. The following requests the next logical object, using the **-v** option to have value displayed with textual name instead of object identifier:

      ```
      clsnmp -v getnext udp
      ```

      The output from this command looks similar to:

      ```
      udpInDatagrams.0 = 653
      ```

3. Setting the MIB variable.
   a. The following sets MIB object sysName.0 to a value of 'hostname.austin.ibm.com':

      ```
      clsnmp set sysName.0  "hostname.austin.ibm.com"
      ```

      This command produces output similar to:

```
1.3.6.1.2.1.1.5.0 = hostname.austin.ibm.com
```

b. The value of MIB object sysName.0 can also be set using the *VariableType* parameter to specify the type of value being set, as in the following example:

```
clsnmp set sysName.0 displayname "hostname.austin.ibm.com"
```

This command produces output similar to:

```
1.3.6.1.2.1.1.5.0 = hostname.austin.ibm.com
```

4. Walking the MIB tree.

The following returns by name all objects beginning with the same object identifier prefix, but with fewer data packages to be exchanged between the clsnmp command and the SNMP agent:

```
clsnmp -h loopback -v -m 10 bulkwalk udp
```

The output of this command looks similar to the following:

```
clsnmp -v walk udp
udpInDatagrams.0 = 653
udpNoPorts.0 = 22
udpInErrors.0 = 0
udpOutDatagrams.0 = 678
udpLocalAddress.0.0.0.0.7 = 0.0.0.0
udpLocalAddress.0.0.0.0.9 = 0.0.0.0
udpLocalAddress.0.0.0.0.13 = 0.0.0.0
udpLocalAddress.0.0.0.0.19 = 0.0.0.0
udpLocalAddress.0.0.0.0.37 = 0.0.0.0
udpLocalAddress.0.0.0.0.161 = 0.0.0.0
udpLocalAddress.0.0.0.0.5020 = 0.0.0.0
udpLocalPort.0.0.0.0.7 = 7
udpLocalPort.0.0.0.0.9 = 9
udpLocalPort.0.0.0.0.13 = 13
udpLocalPort.0.0.0.0.19 = 19
udpLocalPort.0.0.0.0.37 = 37
udpLocalPort.0.0.0.0.161 = 161
udpLocalPort.0.0.0.0.5020 = 5020
```

5. Getting multiple MIB variables.

The following requests multiple MIB objects using the **getbulk** request type. The **getbulk** request type returns the next logical object for one or more MIB objects listed on the command. In the following example, the **-n** option indicates that only one next logical object is requested for the first two variables (sysLocation and ifTable). For all the other objects in the list (tcp, udp, and icmp), the **-m** option indicates that 5 repetitions are requested.

Note: The **getbulk** request type is an SNMPv2 function. The **-h** parameter identifies a host, loopback, defined in the **clsnmp.conf** file as an agent that supports SNMPv2 or SNMPv3.

```
clsnmp -h loopback -v -n 2 -m 5 getbulk sysLocation ifTable tcp udp icmp
```

This command produces output similar to the following:

```
sysLocation.0 = Research Triangle Park, NC
ifIndex.1 = 1
tcpRtoAlgorithm.0 = 4
udpInDatagrams.0 = 782
icmpInMsgs.0 = 22
tcpRtoMin.0 = 0
udpNoPorts.0 = 22
icmpInErrors.0 = 0
tcpRtoMax.0 = 120
udpInErrors.0 = 0
icmpInDestUnreachs.0 = 22
tcpMaxConn.0 = -1
udpOutDatagrams.0 = 807
icmpInTimeExcds.0 = 0
tcpActiveOpens.0 = 1
```

```
udpLocalAddress.0.0.0.0.7 = 0.0.0.0
icmpInParmProbs.0 = 0
```

6. Finding the name of an ASN.1 variable.

   The following sends a request that a search be done to obtain the textual name, for a given *MIBVariable* input, whose internal ASN.1 value best matches the input ASN.1 value. The search begins with the **/etc/mib.defs** file and, if not found, continues with the compiled MIB. Only one *MIBVariable* is allowed per **clsnmp findname** invocation. For example, this can be done with a command similar to the following:

   ```
   clsnmp findname 1.3.6.1.2.1.6.13.1.2
   ```

   This command produces output similar to the following:

   ```
   1.3.6.1.2.1.6.13.1.2 found as: tcpConnLocalAddress
   ```

   A similar example is:

   ```
   clsnmp findname 1.3.6.1.2.1.6.13.1.2.0
   ```

   This command produces output similar to the following:

   ```
   1.3.6.1.2.1.6.13.1.2.0 found as: tcpConnLocalAddress.0
   ```

   Another similar example is:

   ```
   clsnmp findname 1.3.6.1.2.
   ```

   This command produces output similar to the following:

   ```
   1.3.6.1.2. found as: mgmt
   ```

7. Issuing an SNMPv3 request.

   a. If an winSnmpName entry is configured in **/etc/clsnmp.conf** file on the manager host with an entry like the following (all on one line):

      ```
      target1   9.3.149.26 snmpv3 u1 -          - AuthNoPriv   HMAC-SHA
      76784e5935acd6033a855df1fac42acb187aa867 -   -
      ```

      and on the snmpd agent machine 9.3.149.26, user u1 is properly configured, then we can issue command on the manager host:

      ```
      clsnmp -v -h target1 get sysName.0
      ```

      This command will produce output similar to:

      ```
      sysName.0 = somehostname.austin.ibm.com
      ```

   b. It is simple to issue a trap command, as follows:

      ```
      clsnmp trap
      ```

      Note: If the security model of the trap received is SNMPv3, make sure on the manage station where is listens to the trap has the /**etc/clsnmp.conf** file properly configured in order to receive the trap.

Files

Item	Description
/etc/clsnmp.conf	Configuration file for the **clsnmp** command.

Item	Description
/etc/mib.defs	Defines the Management Information Base (MIB) variables the SNMP agent and manager should recognize and handle.

clvupdate Command

Purpose

Cleans up the system if the Live Update operation fails and if the default cleanup attempt is not successful.

Syntax

clvupdate [*-e*] [*-o*] [*-u*] [*-a* **HMChostname** *-U* **HMCusername**] [*-f* **AltCfFile**]

clvupdate [*-e*] [*-n*] [*-u*] [*-a* **HMChostname** *-U* **HMCusername**] [*-f* **AltCfFile**]

clvupdate *-d*

clvupdate *-l*

clvupdate *-r*

clvupdate *-v*

Description

You must clean up the system before you attempt to run a Live Update operation if the initial one failed and the default cleanup operation is not successful. The **clvupdate** command analyzes the state of the system and restores the system to its original state. An unsuccessful attempt during a default cleanup operation indicates that some condition is blocking the cleanup operation. The **clvupdate** command identifies the conditions that are blocking the cleanup operation and prompts the user with the necessary actions. After the user resolves all of the blocking conditions, the **clvupdate** command completes the cleanup operation.

The **clvupdate** command depends on the content of the `/var/adm/ras/liveupdate/liveupdate.cf` file that is generated during the failed Live Update operation. If the `liveupdate.cf` file is lost or corrupted, the **clvupdate** command cannot restore the state of the logical partition (LPAR). You must specify the previous `liveupdate.cf` file by using the *-f* option if you attempted another Live Update operation after the first operation failed.

It is recommended that you run the **clvupdate** command under the guidance of technical support if a failed Live Update operation caused a system failure or an unusable system state.

Note: The cleanup of a failed Live Update operation depends on additional operations on the HMC or PowerVC server that manages the LPAR. You must ensure a valid authentication session with the managing server before a Live Update operation. For instructions about how to use an HMC or PowerVC to create a valid authentication session, refer to the hmcauth and pvcauth commands.

Parameters

HMChostname
Specifies the host name or the IP address of an alternative HMC for authentication.

HMCusername
Specifies the HMC user name of the alternative HMC for authentication. You must have access to all objects and have appropriate task authority on the HMC.

AltCfFile
Specifies the path of an alternative `liveupdate.cf` file that was generated in a previous Live Update operation.

Flags

-d
Remove the surrogate boot disks only

-e
Ignore the Live Update state

-l
Unlock the Live Update lock only

-n
Run the LVUP_ERR0R phase scripts

-o
Force original shutdown operation

-r
Reset the Live Update state and status only

-u
Ignore the Live Update status

-a
Specify an alternative HMC host name

-U
Specify an alternative HMC user name

-f
Specify an alternative `liveupdate.cf` file

-v
Remove the volume group from the previous Live Update operation

Examples

1. If Live Update process fails and the Live Update lock is taken, run the **clvupdate** command with the *-l* option to remove the lock before running the **clvupdate** operation again. The following example shows the usage of the **clvupdate** command with the *-l* option on a display:

   ```
   clvupdate
   The Live Update lock is taken, indicating that a Live Update operation might be in progress.
   Use -l to forcefully remove the lock and proceed with cleanup anyway.
   clvupdate -l.
   clvupdate
   Cleanup action succeeded.
   ```

   Note: The Live Update operation keeps track of its state during the operation in the kernel. The Live Update operation uses the *Live Update kernel state* and *Live Update kernel status* kernel variables for coordinating operations that require strict synchronization. The **clvupdate** command checks these kernel variables before it proceeds with the cleanup operation. By checking these kernel variables, the **clvupdate** identifies situations in which a cleanup operation is not necessary. In some cases, you must specify additional options to bypass these checks, such as when the partition is rebooted after a failed Live Update operation.
2. In some cases, the Live Update kernel state might be clean but requires a cleanup. If the partition has been rebooted after the failed Live Update attempt, you must run the **clvupdate** command with the *-e* option.

   ```
   clvupdate
   The Live Update kernel state is INIT, which normally indicates that the system is in a clean state.
   Use -e to proceed with cleanup anyway.
   clvupdate -e
   Cleanup action succeeded.
   ```

3. If the Live Update operation has aborted unexpectedly, the Live Update kernel status might be in a state that prevents the **clvupdate** command from running. To bypass this check, use the **clvupdate** command with the-*u* option.

```
clvupdate
The Live Update kernel status indicates that some processes from live update might still be running.
Please run cleanup tool later or, if this is not the case, use -u to proceed with cleanup anyway.
clvupdate -u
Cleanup action succeeded.
```

4. If an administrator performed a manual cleanup operation but did not reset the Live Update kernel variables, you can run the **clvupdate** command with the *-r* option to reset the Live Update kernel variables to perform another Live Update operation. An example **clvupdate** command with the *-r* option follows:

```
clvupdate -r
```

Note: The Live Update operation uses the `liveupdate.cf` file to record important configuration information about the operation. The **clvupdate** command reads the `liveupdate.cf` file and identifies the operations that are needed to restore the partition to its original state. The `liveupdate.cf` file is located in the `/var/adm/ras/liveupdate` directory. The old `liveupdate.cf` file is renamed by appending a time stamp to its file name each time the Live Update operation is run. If the **clvupdate** command is run without specifying the *-f* option, it reads the default `liveupdate.cf` file.

5. If you want to perform a cleanup action to undo a change that was caused by a previous failed Live Update operation, use an alternative `liveupdate.cf` file instead of the default file. An example **clvupdate** command with the *-f* option follows:

```
clvupdate -f liveupdate.cf.yyyy-mm-dd_HH:MM:SS.xxx
Cleanup action succeeded.
```

6. An error can occur during a Live Update operation after the blackout time. When an error occurs past that point, the workload is moved to the surrogate partition, and the original partition is moved to an active state. You must run the **clvupdate** command with the *-o* option for the cleanup operation to proceed. An example **clvupdate** command with the *-o* option follows:

```
clvupdate
The clean up process has been aborted because the original partition is still active.
Turn off partition manually or use '-o' option to force shutdown first.
clvupdate -o
Cleanup action succeeded.
```

7. The Live Update operation has a framework that notifies other system components that are affected by the operation. The Live Update notification feature allows other system components to run scripts that coordinate with the Live Update operation at different phases. During one of the phases, you can run scripts to perform a cleanup operation that is specific to system components during a Live Update operation failure. The **clvupdate** command provides the option that runs only the scripts of other system components in certain scenarios. An example **clvupdate** command with the *-n* option on a display follows:

```
clvupdate -n
Cleanup action succeeded.
```

8. You can use an alternative HMC if the HMC that was used during the Live Update is unresponsive. You must authenticate to the HMC first. The following example shows the authentication process on a display:

```
hmcauth -a hmc1 -u user1 -p password
clvupdate -a hmc1 -U user1
Cleanup action succeeded.
```

Note: Surrogate boot disks are imported for debugging purposes if the Live Update operation in a PowerVC mode fails. In this case, the following message is displayed:

```
Surrogate boot disk(s) have been imported as hdiskx for debugging purposes.
Use clvupdate -d to remove them when done.
```

If the disk is no longer needed, it can be deleted by running the **clvupdate** command with the *-d* option.

9. To delete the surrogate boot disk, enter the following command:

```
clvupdate -d
Cleanup action succeeded.
```

10. `lvup_rootvg` is the name of the volume group that the surrogate partition uses to start. The `lvup_rootvg` volume group is not needed after the partition is rebooted after a Live Update operation. The `lvup_rootvg` volume group is not automatically removed after a PowerVC mode Live Update operation. The **clvupdate** command provides an option to remove this volume group. You must authentic the partition with the PowerVC server to perform this operation because a PowerVC operation must be performed. The system displays an output that is similar to the following example:

```
clvupdate -v
1430-159 FAILED: No valid PowerVC session token. Run pvcauth.
pvcauth -a pvchost -u pvcuser -u pvcpasswd
clvupdate -v
The temporary Live Update storage was successfully removed. Cleanup action succeeded.
```

cmp Command

Purpose

Compares the contents of two files and reports the first character that differs.

Syntax

cmp [**-l** | **-s**] *File1 File2*

Description

The **cmp** command compares files designated by the *File1* and *File2* parameters and writes the results to standard output. If you specify a **-** (minus sign) for either the *File1* or *File2* parameter, the **cmp** command reads standard input for that file. Only one file can be read from standard input. Under default conditions, the **cmp** command displays nothing if the files are the same. If they differ, the **cmp** command displays the byte and line number at which the first difference occurs. If the **-l** flag is specified and if one file is an initial subsequence of the other (that is, if the **cmp** command reads an end-of-file character in one file before finding any differences), the **cmp** command notes this. Normally, use the **cmp** command to compare non-text files and the **diff** command to compare text files.

Flags

Item	Description
-l	(Lowercase L) Displays, for each difference, the byte number in decimal and the differing bytes in octal.
-s	Returns only an exit value. A value of 0 indicates identical files; value of 1 indicates different files; a value of 2 indicates inaccessible file or a missing option.

Exit Status

This command returns the following exit values:

Item	Description
0	The files are identical.

Item	Description
1	The files are different. This value is given even if one file is an initial subsequence of the other (one file is identical to the first part of the other).
>1	An error occurred.

Examples

1. To determine whether two files are identical, enter:

   ```
   cmp prog.o.bak prog.o
   ```

   This compares `prog.o.bak` and `prog.o`. If the files are identical, then a message is not displayed. If the files differ, then the location of the first difference is displayed; for example:

   ```
   prog.o.bak prog.o differ: char 4, line 1
   ```

   If the message `cmp: EOF on prog.o.bak` is displayed, then the first part of `prog.o` is identical to `prog.o.bak`, but there is additional data in `prog.o`.
2. To display each pair of bytes that differ, enter:

   ```
   cmp -l prog.o.bak prog.o
   ```

   This compares the files, and then displays the byte number (in decimal) and the differing bytes (in octal) for each difference. For example, if the fifth byte is octal 101 in `prog.o.bak` and 141 in `prog.o`, the **cmp** command displays:

   ```
   5 101 141
   ```
3. To compare two files without writing any messages, enter:

   ```
   cmp -s prog.c.bak prog.c
   ```

   This gives an exit value of `0` if the files are identical, a value of `1` if different, or a value of 2 if an error occurs. This form of the command is normally used in shell procedures. For example:

   ```
   if cmp -s prog.c.bak prog.c
   then
    echo No change
   fi
   ```

   This partial shell procedure displays `No change` if the two files are identical.

Files

Item	Description
/usr/bin/cmp	Contains the **cmp** command.

col Command

Purpose

Filters for standard output text having reverse line feeds and forward/reverse half-line-feeds.

Syntax

col [**-b**] [**-f**] [**-p**] [**-x**] [**-T** *Name*] [**-l** *Number*]

Description

The **col** command reads a text file from standard input and writes to standard output. It performs the line overlays implied by the **flr** commands (reverse line feeds), as well as by the **hlf** and **hlr** commands (forward and reverse half-line-feed, respectively). The **nterm** file format document gives a description of these line-feed commands. Use the **col** command for filtering multicolumn output produced by the **nroff** command, the **.rt** request, and by output from the **tbl** command.

Use the **col** command as an **nroff** backend filter for devices that cannot handle reverse line motions (such as most impact printers). To print correctly, use the **col** command to process outputs from the **tbl** command, the **neqn** command, or explicit reverse motion request files (such as the **.sp -10V** file), or files with 2-column output. Do not process the **nroff** output targeted for the following devices with the **col** command:

- **hplj**
- **ibm4019**
- **ibm5577**
- **ibm5575**

Unless the **-x** flag is given, whenever possible, the **col** command converts white spaces to tabs upon output wherever possible to shorten printing time.

The **col** command, used with the **-T37** file, assumes the ASCII control characters, SO (\017) and SI (\016), begin and end text in an alternate character set. The **col** command remembers the character set each input character belongs to and upon output, generates SI and SO characters as appropriate to ensure that each character is printed in the correct character set.

Upon input, the **col** command accepts only the control characters for the Space, Backspace, Tab, and Return keys; the new-line character; the SI, SO (with the **-T37** file), and VT control characters; and the reverse line feed, forward half-line-feed and reverse half-line-feed characters. The VT control character (\013) is an alternate form of full reverse line feed, included for compatibility with some earlier programs of this type. The **col** command ignores all other nonprinting characters.

Normally, the **col** command ignores any escape sequences that are unknown to it and found in the input. However, the **-p** option can be used to cause the **col** command to output these sequences as regular characters, subject to overprinting from reverse line motions. The use of this option is highly discouraged unless the user is fully aware of the textual position of the escape sequences.

Notes:

1. If the output is being sent to a device that can interpret half-line motions, enter:

   ```
   nroff -Tppds File... | col -f -Tppds
   ```

   Otherwise, for example, enter:

   ```
   nroff -Tlp File... | col -Tlp
   ```

2. The maximum number of lines that can be backed up is 128.
3. No more than 800 characters, including backspaces, are allowed on a line.
4. Local vertical motions that would result in backing up over the first line are ignored. As a result, the first line must not contain any superscripts.

Flags

Item	Description
-b	Assumes that the output device in use is not capable of backspacing. In this case, if two or more characters are to be displayed in the same position, only the last one that is read is displayed in the output.

Item	Description
-f	Suppresses the default treatment of half-line motions in the input. Normally, the **col** command does not emit half-line motions on output, although it does accept them in its input. With this flag, output can contain forward half-line-feeds (**hlf**) but not reverse line feeds (**flr** or **hlr**).
-p	Displays unknown escape sequences as characters, subject to overprinting from reverse line motions. Normally, the **col** command ignores them.
-x	Converts tabs to white space.
-T*Name*	Uses the workstation specification indicated by the *Name* variable. *Name* variables for "Terminal Names for Typewriter-like Devices and Line Printers" are discussed in the **nroff** command **-T** *Name* flag. The default is **37**.
-l *Number*	(lowercase L) Sends the specified number lines of text in memory to a buffer during processing.

Exit Status

The following exit values are returned:

Item	Description
0	Indicates successful completion.
>0	Indicates an error occurred.

colcrt Command

Purpose

Filters **nroff** command output for cathode ray tube (CRT) previewing.

Syntax

colcrt [**-**] [**-2**] [*File* ...]

Description

The **colcrt** command filters output from the **nroff** command so that the output can be previewed on a CRT. The **colcrt** command provides virtual half-line-feed and reverse line-feed sequences for terminals without these capabilities. The **colcrt** command changes underline characters to dashes and places these characters and the half-line characters on new lines between the normal output lines.

Notes:

1. Use this command with the **37** viewing device
2. The **-** (minus sign) flag removes all underlining; therefore, a true underline character does not show with the **-** (minus sign) flag.
3. It is not possible to back up more than 102 lines.
4. General overstriking is lost. As a special case, the | (vertical bar) overstruck with the - (dash) or the _ (underline) characters becomes the + (plus sign).
5. Lines are truncated to up to 132 characters.

Parameters

Item	Description
File	Specifies a file processed by the **nroff** command for viewing on a CRT.

Flags

Item	Description
-	Suppresses underlining. This flag is useful for previewing boxed tables from the **tbl** command.
-2	Causes all half-lines to be printed, effectively double-spacing the output. This is useful when printing output with subscripts and superscripts on a line printer, where half-lines normally are not displayed.

Examples

A typical use of the **colcrt** command is:

```
tbl exum2.n | nroff -ms -T37 | colcrt - | pg
```

colrm Command

Purpose

Extracts columns from a file.

Syntax

colrm *First* [*Last*]

Description

The **colrm** command removes selected columns from a file. Input is taken from standard input. Output is sent to standard output.

If called with one parameter, the columns of each line are removed starting with the specified column. If called with two parameters, the columns from the first column to the last column are removed.

Column numbering starts with column 1.

Examples

To remove columns from the `text.fil` file, enter:

```
colrm 6 < text.fil
```

If `text.fil` contains:

```
123456789
```

then the **colrm** command displays:

```
12345
```

Files

Item	Description
/usr/bin/colrm	Contains the **colrm** command.

comb Command (SCCS)

Purpose

Combines SCCS deltas.

Syntax

comb [**-o**] [**-s**] [**-c** *List* | **-p** *SID*] *File*

Description

The **comb** command writes to standard output a shell procedure that can combine specified SCCS deltas (SIDs) or all deltas into one delta. You can reduce the size of your Source Code Control System (SCCS) file by running the resulting procedure on the file. To see how much the file will be reduced, run the **comb** program with the **-s** flag. If you specify a directory for the *File* value, the **comb** command performs the requested actions on all SCCS files (that is, those having an **s.** prefix). If you specify a **-** (minus) for the *File* value, the **comb** command reads standard input and interprets each line as the name of an SCCS file. The **comb** command continues to take input until it reads an end-of-file character.

If you do not specify any flags, the **comb** command preserves only leaf deltas and the minimal number of ancestors needed to preserve the tree.

> **Note:** The **comb** command may rearrange the shape of the tree deltas. It may not save any space. In fact, it is possible for the reconstructed file to actually be larger than the original.

Flags

> **Note:** Each flag or group of flags applies independently to each named file.

Item	Description
-c *List*	Specifies a list of deltas (*SID*s) that the shell procedure will preserve (see the **get** command **-i** *List* flag). The procedure combines all other deltas.
-o	Accesses the reconstructed file at the release of the delta to be created for each **get** command **-e** flag generated; otherwise, accesses the reconstructed file at the most recent ancestor. Using the **-o** flag may decrease the size of the reconstructed SCCS file. It may also alter the shape of the delta tree of the original file.
-p *SID*	Specifies the *SID* of the oldest delta for the resulting procedure to preserve. All older deltas are combined in the reconstructed file.
-s	Causes the **comb** command to generate a shell procedure that produces a report for each file listing: the file name, size (in blocks) after combining, original size (also in blocks), and percentage change computed by the formula: `100 * (original - combined) / original` You should run the **comb** program using this flag and then run its procedure before combining SCCS files in order to judge how much space will actually be saved by the combining process.

Examples

1. To generate a report on how much space will be saved by combining all deltas older than SID `1.4` for sccs file `s.test.c`, enter:

   ```
   comb -p1.4 -s s.test.c
   ```

   Run the report by piping the output of the above command to the **sh** command.
2. To actually perform the combination, enter:

   ```
   comb -p1.4 s.test.c
   ```

Files

Item	Description
s.COMB	The name of the reconstructed SCCS file.
comb*	Temporary files.

comm Command

Purpose

Selects or rejects lines common to two sorted files.

Syntax

comm [**-1** **-2** **-3**] *File1 File2*

Description

> **Note:** If you specify - (minus) for one of the file names, the **comm** command reads standard input.

The **comm** command reads the *File1* and *File2* parameters and writes, by default, a three-column output to standard output. The columns consist of:

- Lines that are only in *File1*
- Lines that are only in *File2*
- Lines that are in both *File1* and *File2*.

Both *File1* and *File2* should be sorted according to the collating sequence specified by the current National Language environment.

Flags

Item	Description
-1	Suppresses the display of the first column (lines in *File1*).
-2	Suppresses the display of the second column (lines in *File2*).
-3	Suppresses the display of the third column (lines common to *File1* and *File2*).

Exit Status

This command returns the following exit values:

Item	Description
0	All input files were output successfully.
>0	An error occurred.

Examples

1. To display the lines unique to each file and common to both, enter:

```
comm things.to.do things.done
```

   If the files `things.to.do` and `things.done` contain the following lists:

```
things.to.do

buy soap
groceries
luncheon
meeting at 3
system update
tech. review

things.done

2nd revision
interview
luncheon
system update
tech. review
weekly report
```

   then the **comm** command displays:

```
       2nd revision
buy soap
groceries
       interview
                luncheon
meeting at 3
                system update
                tech. review
       weekly report
```

   The first column contains the lines found only in `things.to.do`. The second column, indented with a tab character, lists the lines found only in `things.done`. The third column, indented with two tabs, lists the lines common to both.

2. To display the lines that appear in only one file, enter:

```
comm -23 things.to.do things.done
```

   This suppresses the second and third columns of the **comm** command listing. If the files are the same as in Example 1, then the following is displayed:

```
buy soap
groceries
meeting at 3
```

File

Item	Description
/usr/bin/comm	Contains the **comm** command.

command Command

Purpose

Executes a simple command.

Syntax

```
command [-p] command_name [argument...]
```

```
command [-p] [-v|-V ] command_name
```

Description

The **command** command causes the shell to treat the specified command and arguments as a simple command, suppressing shell function lookup.

Normally, when a / (slash) does not precede a command (indicating a specific path), the shell locates a command by searching the following categories:

1. special shell built-ins
2. shell functions
3. regular shell built-ins
4. **PATH** environment variable

For example, if there is a function with the same name as a regular built-in, the system uses the function. The **command** command allows you to call a command that has the same name as a function and get the simple command.

The **command -v** and **command -V** commands write to standard output what path name will be used by the shell and how the shell interprets the command type (built-in, function, alias, and so forth). Since the **-v** and **-V** flags produce output in relation to the current shell environment, the **command** command is provided as a Korn shell or POSIX shell regular built-in command. The **/usr/bin/command** command might not produce correct results, because it is called in a subshell or separate command execution environment,. In the following example the shell is unable to identify aliases, subroutines, or special shell commands:

```
(PATH=foo command -v)
nohup command -v
```

Flags

Item	Description
-p	Performs the command search using a default value for the **PATH** environment variable that is finds all of the standard commands.
-v	Writes to standard output the path name used by the current shell to invoke the specified command, according to the following conventions: • Commands, regular built-in commands, commands including a / (slash), and any implementation-provided functions found by the **PATH** environment variable are written as absolute path names. • Shell functions, special built-in commands, regular built-in commands not associated with a **PATH** environment variable search, and shell reserved words are written as just their names. • Aliases are identified as such, and their definitions are included in the string. If the specified command name cannot be found, no output is written and the exit status returns a **>0** value.

Item	Description
-V	Writes to standard output the command name that will be interpreted by the current shell environment. Although the format of this output is unspecified, The output indicates in which of the following categories the command falls: • Commands, regular shell commands, and any implementation-provided subroutines found using the **PATH** environment variable are identified as such and written as absolute path names. • Other shell functions are identified as functions. • Aliases are identified as such, and their definitions are included in the string. • Special built-in commands are identified as such. • Regular built-in commands not associated with the **PATH** environment variable search are identified as such. • Shell reserved words are identified as such.

Exit Status

When the **-v** or **-V** flag is specified, the following exit values are returned:

Item	Description
0	Successful completion.
>0	The command specified with the *command_name* parameter could not be found or an error occurred.

When the **-v** or **-V** flag is not specified, the following exit values are returned:

Item	Description
126	The command specified by the *command_name* parameter was found but could not be invoked.
127	An error occurred in the **command** command, or the command specified by the *command_name* parameter could not be found.

Otherwise, the **command** command returns the exit status associated with the command specified by the *command_name* parameter.

Examples

1. To make a version of the **cd** command that prints out the new working directory whenever you change directories, enter:

```
cd () {
        command cd "$@" >/dev/null
        pwd
}
```

2. To start off a secure shell script, one in which the script avoids being spoofed by its parent, enter:

```
IFS='
'
#       The preceding value should be <space><tab><newline>.
#       Set IFS to its default value

\unalias -a
#       Unset all possible aliases.
#       Note that unalias is escaped to prevent an alias
#       being used for unalias.

unset -f command
#       Ensure command is not a user function.

PATH="$(command -p getconf _CS_PATH):$PATH"
#       Put on a reliable PATH prefix.

# ...
```

   At this point, given correct permissions on the directories called by the **PATH** environment variable, the script has the ability to ensure that any command it calls is the intended one.

Files

Item	Description
/usr/bin/ksh	Contains the Korn shell **command** built-in command.
/usr/bin/command	Contains the **command** command.

comp Command

Purpose

Composes a message.

Syntax

comp [**+***Folder*] [**-draftfolder +***Folder* | **-nodraftfolder** *Folder*] [*Message* | **-draftmessage** *Message*] [**-file** *File*] [**-editor** *Editor* | **-noedit**] [**-form** *FormFile*] [**-use** | **-nouse**] [**-nowhatnowproc** | **-whatnowproc** *Program*]

Description

The **comp** command starts an editor that assists you in creating and modifying messages. The **comp** command provides a header template, the **/etc/mh/components** file. By default, the specified editor creates a *UserMhDirectory***/draft** file. If a **draft** file already exists, the **comp** command prompts you for permission to replace or use the existing file. To edit an existing **draft** file without being prompted for permission, specify the **-use** flag.

Once started, the editor prompts you to enter values for each of the message header fields. The **comp** command uses the definitions in the *UserMhDirectory***/components** file for the header fields. If the file does not exist, the **/etc/mh/components** file is used. You can use the **-form** or **+***Folder* flag to specify an alternative header format.

To exit the editor, use the Ctrl-D sequence. When you exit the editor, the **comp** command responds with a `What now?` prompt. From this prompt, you can specify any of the **whatnow** subcommands. To see a list of the available subcommands, press Enter. You can use the subcommands to continue composing the message, direct the disposition of the message, or end the processing of the **comp** command.

> **Note:** A line of dashes or a blank line must be left between the header and the body of the message for the message to be identified when it is sent.

The **-file**, **-draftfolder**, and **-draftmessage** flags are used to specify existing draft messages. If the **-draftfolder +***Folder* flag is followed by a *Message* parameter, it is the same as specifying the **-draftmessage** flag. If you wish, you can define a default `Draft-Folder:` entry in your Message Handler (MH) **$HOME/.mh_profile** file.

Flags

Item	Description
-draftfolder +*Folder*	Identifies the folder containing the draft message. If a message is not specified with this flag, the default message is **new**.
-draftmessage *Message*	Identifies the draft message. Specifying a *Message* variable after the **-draftfolder +***Folder* flag is the same as specifying the **-draftmessage** flag.
-editor *Editor*	Specifies the initial editor for composing the message. If you do not specify the **-editor** flag, the **comp** command selects the default editor specified by the `Editor:` entry in your **$HOME/.mh_profile** file.
-file *File*	Places the draft message in the specified file. If you do not specify the absolute path name for the *File* variable, the **comp** command places the file in the user's MH directory. If a file is specified, the **comp** command prompts you for the disposition of the draft.

Item	Description
+*Folder Message*	Uses the header information from a file in the specified folder. If you specify a folder but no message, the **comp** command uses the current message as the default.
-form *FormFile*	Uses the header fields specified by the *FormFile* variable. The **comp** command treats each line in *FormFile* as a format string.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
Message	Specifies a message. Use the following references to specify a message: ***Number*** Number of the message. **cur or . (period)** Current message. This is the default. **first** First message in a folder. **last** Last message in a folder. **next** Message following the current message. **prev** Message preceding the current message.
-nodraftfolder	Places the draft in the *UserMhDirectory***/draft** file. This is the default.
-noedit	Suppresses the initial edit. When you specify this flag, you receive the `What now?` prompt.
-nouse	Creates a new message.
-nowhatnowproc	Prevents interaction with the editor and the `What now?` prompt.
-use	Continues composing an existing draft of a message.
-whatnowproc *Program*	Starts the specified program to guide you through the composing tasks. If you specify the **whatnow** command as the value for the *Program* variable, the **comp** command starts an internal **whatnow** procedure, instead of a program with the file name **whatnow**.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Draft-Folder:`	Sets the default folder for drafts.
`Editor:`	Sets the default initial editor.
`fileproc:`	Specifies the program used to refile messages.
`Msg-Protect:`	Sets the protection level for the new message files.
`Path:`	Specifies a user's MH directory.
`whatnowproc:`	Specifies the program used to prompt `What now?` questions.

Examples

1. To compose a new message, enter:

   ```
   comp
   ```

   The system prompts you to enter the information for the message fields. To bypass a field, press the Enter key. When the header information is complete, enter the text for the body of the message.

   To finish composing a message and exit the editor, press the Ctrl-D sequence. The following prompt is displayed on your screen:

   ```
   What now?
   ```

   Pressing the Enter key displays a list of the **whatnow** subcommands. If you want to send the message, enter the **send** subcommand after the `What now?` prompt.
2. To compose a new message using the `vi` editor, enter:

   ```
   comp  -editor vi
   ```
3. To compose a message using message 8 in the `schedules` folder, enter:

   ```
   comp  +schedules 8  -use
   ```
4. To compose a message using a message draft in the `/home/mike/parts` file, enter:

   ```
   comp  -file /home/mike/parts
   ```

   The system prompts you for the disposition of the file. Press the Enter key for a list of options. Select the appropriate option.

Files

Item	Description
UserMhDirectory/**components**	Specifies the user's default message format. (If it exists, it overrides the system default message format.)
UserMhDirectory/**draft**	Contains the current draft message.
$HOME/.mh_profile	Specifies the user's MH profile.
/etc/mh/components	Identifies the system default message format.
/usr/bin/comp	Contains the **comp** command.

compare_report Command

Purpose

Compares fileset levels to those available and generates a report of filesets needed.

Syntax

To compare filesets installed on a system to filesets contained in a fix repository:

compare_report -s -i *FixDir* { [**-l**] [**-h**] [**-m**] [**-n**] } [[[**-t** *ReportDir*] [**-Z**]] | **-v**]

To compare filesets installed on a system to filesets available from the support Web site:

compare_report -s -r *ServiceReport* { [**-l**] [**-h**] } [[[**-t** *ReportDir*] [**-Z**]] | **-v**]

To compare filesets contained in a fix repository to filesets available from the support Web site:

compare_report -i *FixDir* **-r** *ServiceReport* [[[**-t** *ReportDir*] [**-Z**]] | **-v**]

To compare a list of installed software on a base system to another system:

compare_report -b *BaseList* **-o** *OtherList* { [**-l**] [**-h**] [**-m**] [**-n**] } [[[**-t** *ReportDir*] [**-Z**]] | **-v**]

To compare a list of installed software to filesets contained in a fix repository:

compare_report -b *BaseList* **-i** *FixDir* { [**-l**] [**-h**] [**-m**] [**-n**] } [[[**-t** *ReportDir*] [**-Z**]] | **-v**]

To compare a list of installed software to filesets available from the support Web site:

compare_report -b *BaseList* **-r** *ServiceReport* { [**-l**] [**-h**] } [[[**-t** *ReportDir*] [**-Z**]] | **-v**]

Description

The **compare_report** command compares the filesets installed on a system with the contents of a fix directory or a service report that contains a list of the latest available fixes. The comparison reports produced provide assistance in assuring a system is running a certain level of software. The fix directory can be an image repository, such as an **lpp_source**. The service report is a list of both the latest level fixes and the fixes contained in the latest technology level and can be downloaded from the IBM System p Support for AIX 5L and Linux servers site (http://www.ibm.com/servers/eserver/support/unixservers/index.html). Some of the generated reports can be used as input to request fixes from the IBM System p Support for AIX 5L and Linux servers site.

The `lslpp` command and the **compare_report** command both show information about interim fixes installed on the system. The `lslpp -L` or `lslpp -Lc` command must be run by root, and any interim fix information returned is used by the `compare_report` command. The information includes an interim fix label and a level value. The interim fix label is the equivalent of a fileset name, and its level is based on the time (*YY.MM.DD.HHMMSS*, where *YY* is the year, *MM* is the month, *DD* is the day, *HH* is the hour, *MM* is the minute, and *SS* is the second) in which the interim fix was packaged.

Flags

Item	Description
-b *BaseList*	The name of the file containing the software installed on the base system (generated with **lslpp -Lc**)
-h	Indicates that the higher level fileset reports should be generated. This will generate one or all of the reports **higherlevel.rpt**, **higherthanmaint.rpt**, or **basehigher.rpt**, depending on which comparisons are performed. This flag is only valid when used either with the **-s** or with both the **-b** and the **-o** flags.
-i *FixDir*	Specifies the name of the fix repository directory. The fileset levels of the images contained in this directory will be used in the comparison.

Item	Description
-l	Indicates that the lower level fileset reports should be generated. This will generate one or all of the reports **lowerlevel.rpt**, **lowerthanlatest1.rpt**, **lowerthanmaint.rpt**, **lowerthanlatest2.rpt**, or **baselower.rpt**, depending on which of the comparisons are performed. This flag is only valid when used either with the **-s** or with both the **-b** and the **-o** flags.
-m	Indicates that a fileset report should be generated that lists either the filesets installed on the system that are not in the image repository, or the filesets installed on the base system that are not installed on the other system. This will generate either the **no_update_found.rpt** or the **baseonly.rpt** report file. This flag is only valid when both the **-s** and **-i** flags are specified or when both the **-b** and **-o** flags are specified.
-n	Indicates that a fileset report should be generated that lists either the filesets in the image repository that are not installed on the system or the filesets installed on the other system that are not installed on the base system. This will generate either the **notinstalled.rpt** or the **otheronly.rpt** report file. This flag is only valid when both the **-s** and **-i** flags are specified or when both the **-b** and **-o** flags are specified.
-o *OtherList*	The name of the file containing the software installed on another system that will be compared to a base system (generated with the command **lslpp -Lc**).
-r *ServiceReport*	Specifies a file that contains the list of available updates. This file can be obtained from the support Web site.
-s	Specifies that the comparison should involve a list of the fileset levels that are installed on this system.
-t *ReportDir*	Specifies the target directory where the comparison reports will be stored. If the **-t** flag is not specified, the reports will be stored in the **/tmp** directory. If report files already exist in the specified directory, they will be removed and recreated. This flag is not valid with the **-v** flag.
-v	Specifies that no report files should be saved to disk. This flag is not valid with the **-t** or **-Z** flags.
-Z	Suppresses displaying the report output to stdout. This flag is not valid with the **-v** flag.

Exit Status

0

The command completed successfully.

>0
An error occurred.

Examples

1. To compare filesets installed on the system to filesets contained in an image repository, type:

   ```
   compare_report -s -i /tmp/imagedir -l -n
   ```

   This command will create reports listing filesets on the system that are at a lower level and filesets in the image repository that are not installed on the system. If all reports (**-l**, **-h**, **-m**, **-n**) are requested for this type of comparison, the following reports will be generated:

   - **lowerlevel.rpt** (generated with **-l**)
   - **higherlevel.rpt** (generated with **-h**)
   - **no_update_found.rpt** (generated with **-m**)
   - **notinstalled.rpt** (generated with **-n**)

2. To compare filesets installed on the system to filesets available from the support Web site, type:

   ```
   compare_report -s -r /tmp/LatestFixData -l -Z
   ```

   This command will create reports listing filesets on the system that are at a lower level from the latest levels, and those that are at a lower level than the last technology level. The reports will be saved to disk but not displayed to stdout. If all reports (**-l**, **-h**) are requested for this type of comparison, the following reports will be generated:

   - **lowerthanlatest1.rpt** (generated with **-l**)
   - **lowerthanmaint.rpt** (generated with **-l**)
   - **higherthanmaint.rpt** (generated with **-h**)

3. To compare filesets contained in an image repository to filesets available from the support Web site, type:

   ```
   compare_report -i /tmp/imagedir -r /tmp/LatestFixData
   ```

   This command creates a report listing filesets in the image repository that are at lower levels than the latest levels available from the support Web site. The **lowerthanlatest2.rpt** report is the only report generated from this type of comparison.

4. To compare a list of installed software on a base system to a list of installed software on another system, type:

   ```
   compare_report -b /tmp/base.lslpp.out -o /tmp/other.lslpp.out -l -h -m -n
   ```

   This command will create reports listing the following:

   - filesets on the base system that are at a lower level than the other system
   - filesets on the base system that are at a higher level than the other system
   - filesets installed on the base system that are not installed on the other system
   - filesets installed on the other system that are not installed on the base system

   If all reports (**-l**, **-h**, **-m**, and **-n**) are requested for this type of comparison, the following reports will be generated respectively:

   - **baselower.rpt** (generated with **-l**)
   - **basehigher.rpt** (generated with **-h**)
   - **baseonly.rpt** (generated with **-m**)
   - **otheronly.rpt** (generated with **-n**)

Files

Item	Description
/usr/sbin/compare_report	Contains the **compare_report** command.

compress Command

Purpose

Compresses data.

Syntax

compress [**-c**] [**-C**] [**-d**] [**-F**] [**-f**] [**-n**] [**-q**] [**-v**] [**-V**] [**-b** *Bits*] [*File* ...]

Description

The **compress** command compresses data, using adaptive Lempel-Zev coding to reduce the size of files. Each original file specified by the *File* parameter is replaced when possible by a compressed file with a **.Z** appended to its name. If the invoking process has appropriate privileges, the compressed file retains the same ownership, modes, and modification time of the original file. If the path of the file specified is more than 1023 bytes the command does not work. If no files are specified, the standard input is compressed to the standard output. If compression does not reduce the size of a file, a message is written to standard error and the original file is not replaced.

Note: Files must have correct permissions to be replaced.

The amount of compression depends on the size of the input, the number of bits per code specified by the *Bits* variable, and the distribution of common substrings. Typically, source code or English text is reduced by 50 to 60%. The compression of the **compress** command is usually more compact and takes less time to compute than the compression achieved by Huffman coding (as used in the **pack** command) or adaptive Huffman coding.

Flags

Item	Description
-b *Bits*	Specifies the maximum number of bits to use to replace common substrings in the file. The value of the *Bits* variable must be in the range from 9 bits through 16 bits, with the default being 16 bits. When compressing data, the algorithm first uses all of the 9-bit codes (257 through 512) to replace as many substrings as possible. Then it uses all 10-bit codes, and so on, continuing until the limit specified by the **-b** flag is reached.
-c	Writes to standard output. No files are changed.
-C	Produces output compatible with the Berkeley Software Distribution (BSD) Revision 2.0.
-d	Causes the **compress** command to function exactly like the **uncompress** command.
-f or **-F**	Forces compression. The **-f** and **-F** flags are interchangeable. Overwrites the *File***.Z** file if it already exists. After the value of the *Bits* variable is attained, the **compress** command periodically checks the compression ratio. If it is increasing, the **compress** command continues to use the existing code dictionary. However, if the compression ratio decreases, the **compress** command discards the table of substrings and rebuilds it. Rebuilding the table allows the algorithm to adapt to the next block of the file. When the **.Z** file already exist, if the **-f** flag is not given, and the process is not running in the background, it prompts to verify whether to overwrite the existing **.Z** file.

Item	Description
-n	Omits the compressed file header from the compressed file. **Note:** If this option is used, the **-n** flag should also be used when using the **uncompress** command to uncompress the file.
-q	Suppresses the display of compression statistics generated by the **-v** flag. If several **-v** and **-q** flags are on the same command line, the last one specified controls the display of the statistics.
-v	Writes the percentage of compression.
-V	Writes the current version and compile options to standard error.

Parameters

Item	Description
File	Specifies the file to compress.

Return Values

If an error occurs, the exit status is 1. If the **compress** command exits without compressing a file, it exits with a status of 2. Otherwise, the **compress** command exits with a status of 0.

The **compress** command detects an error and exits with a status of 1 if any of the following events occur:

- An input file is not a regular file.
- An input file name is too long to append the **.Z** extension.
- An input file cannot be read or an output file cannot be written.

Exit Status

Item	Description
0	Successful completion.
1	An error occurred.
2	One or more files were not compressed because they would have increased in size (and the **-f** flag was not specified).
>2	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

To compress the `foo` file and write the percentage of compression to standard error, enter:

```
compress -v foo
```

The `foo` file is compressed and renamed `foo.Z`.

comsat Daemon

Purpose

Notifies users of incoming mail.

Syntax

/usr/sbin/comsat [**-d** *Directory*]

Description

The **comsat** daemon is the server that receives reports of incoming mail and notifies users if they have enabled this service with the **biff** command. Started by the **inetd** daemon, the **comsat** daemon is not meant to be used at the command line. The **comsat** daemon receives messages on a datagram port associated with the **biff** service specification. The one-line messages are of the form:

```
user@mailbox-offset
```

If the user specified is logged in to the system and has run the **biff y** command, the first 7 lines or 560 characters of the message are displayed on the user's login terminal. Lines that appear to be part of a message header other than the `From:` or `Subject:` lines are not included in the displayed message.

Flags

Item	Description
-d *Directory*	Specifies the name of the directory to use as the system mail directory. If the **-d** flag is not specified, the **comsat** daemon uses the **/var/spool/mail** directory as the default system mail directory.

Files

Item	Description
/etc/utmp	Contains a list of users who are logged in, including their terminals.
/etc/services	Contains a list of Internet network services and the well-known ports where the servers accept connections.

configassist Command

Purpose

Displays the Configuration Assistant wizard.

Syntax

/usr/cfgassist/bin/configassist

Description

The Configuration Assistant wizard displays automatically after the operating system is installed and is used to assist with configuration tasks. It can also be run at any time to complete additional configuration.

Note: The full pathname of this command, **/usr/cfgassist/bin/configassist**, must be specified.

Flags

None

Examples

N/A

conflict Command

Purpose

Searches for alias and password conflicts.

Syntax

conflict [**-mail** *User*] [**-search** *Directory ...*] [*File ...*]

Description

The **conflict** command finds invalid mail drops and alias conflicts. The **conflict** command is not started by the user. The **conflict** command is called by the **cron** daemon and other programs used for system accounting. However, root user authority and the full path name of the command, **/usr/lib/mh/conflict**, are required to invoke the program.

The **conflict** command searches specified mail drop directories for mailbox files with names that do not correspond to valid users in the **/etc/passwd** file. In addition, the program searches alias files specified by the *File* parameter for duplicate names that do not resolve to the same address. By default, the **conflict** command searches the **/etc/mh/MailAliases** file.

The **conflict** command also searches entries in the group file (**/etc/group**) for invalid user names and users who do not have a valid group number.

Command output is to the monitor unless you specify the **-mail** flag. The **-mail** flag sends the command output to the specified user.

Flags

Item	Description
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.
-mail *User*	Sends the results of the **conflict** command to the user specified by the *User* variable.
-search *Directory*	Searches the directory indicated by the *Directory* variable for mailboxes that are not valid. You can specify any number of **-search** flags. The default mailbox directory is **/var/spool/mail**.

Files

Item	Description
/etc/mh/MailAliases	Contains the default mail alias file.
/etc/passwd	Contains a list of users.

Item	Description
/etc/group	Contains a list of groups.
/var/spool/$USER	The mail drop for the user **$USER.**
/$HOME/.mh_profile	Contains the MH user profile.
/etc/mh/mtstailor	Contains MH command definitions.

confsetcntrl Command

Purpose

Manage a set of time-based Workload Manager (WLM) configurations.

Syntax

confsetcntrl **-C** *ConfigurationSet DefaultConfig*

confsetcntrl { **-D** | **-R** } *ConfigurationSet*

confsetcntrl [**-d** *ConfigurationSet*] { **-a** | **-r** } *Configuration TimeRange*

confsetcntrl [**-d** *ConfigurationSet*] [**-l** | **-c**]

Description

The **confsetcntrl** command supports the following operations:

- Create a new configuration set with its initial default regular configuration.
- Delete an existing configuration set (this includes the configuration set directory and its **.times** and **description** files, but does not affect the regular configurations of the set).
- Add or remove from a configuration set a configuration and its associated time range.
- Remove from a configuration set all configurations and associated time ranges.
- Check the configuration set file.
- List all the configurations contained in a set with their associated time ranges.

Note: Only the root user can create, delete, or change configuration sets, but any user can list or check them.

Time Ranges

Time ranges are used to indicate at which day of the week and which times of the day the associated configuration will be used by the WLM for classifying processes, for accounting, and regulation.

A time range is represented by a range of days, with 0 representing Sunday and 6 representing Saturday, and a range of time, in 24 hour format with hours and minutes specified. These two ranges are separated with a comma. In each range, values are separated with a minus sign, and values may wrap (the first value may be greater than the second one).

The range of days may be omitted, which means every day of the week. Both ends of this range are included. It may then also consist in only one day: 1 is valid and stands for 1-1.

The range of time may be omitted, which means the whole day. Elsewhere, start and end times must be specified. Hours and minutes are separated with a colon or a dot. The end time is not part of the range, so 24:00 is a valid end time but 12:00-12:00 is empty and not valid.

At least one of the day or time ranges must be present. A single minus sign is a valid time range and is a special case: It is called the default time range and means always outside the other defined time ranges if any. This is different from specifying all the time, for example with 0-6,00:00-24:00

For the WLM to be able to find which configuration must be activated, there must exist one and only one configuration applicable at any time of the week. The default time range, which is added when creating a set, is useful to avoid the possibility that no configuration would be applicable for some time. Additional time ranges must not overlap with each other.

Time range examples:

1-4,8:00-17:00
Monday to Thursday, from 8 am to 5 pm

5-0,22:00-6:00
Friday, Saturday and Sunday, from midnight to 6 am and from 10 pm to midnight

3
Wednesday

14:00-16:30
Every day from 2 pm to 4:30 pm

-
The default time range

Flags

Item	Description
-a *Configuration TimeRange*	Adds *Configuration* to the configuration set for the given *TimeRange*. *Configuration* must be an existing WLM regular configuration. It may appear several times in a set associated with different time ranges. **Note:** Even if time ranges become not coherent due to this operation, the changes are performed, but a warning is reported indicating that further changes are needed.
-c	Checks all the configuration/time range pairs of the set.
-C *ConfigurationSet DefaultConfig*	Creates configuration set *ConfigurationSet* with *DefaultConfig* initial configuration, having default time range. (The default time range means always outside any other explicit time range. Only one is allowed in a set.) *DefaultConfig* must be an existing WLM regular configuration.
-d *ConfigurationSet*	Specifies an alternate configuration set. If not given, current configuration set will be the target of the command.
-D *ConfigurationSet*	Deletes configuration set *ConfigurationSet*.
-l	Checks and lists all the configuration/time range pairs of the set. This is the default operation if no flag is given.
-r *Configuration TimeRange*	Removes the *Configuration* and *TimeRange* pair from the configuration set. This pair is supposed to exist in the set. **Note:** Even if time ranges become not coherent due to this operation, the changes are performed, but a warning is reported indicating that further changes are needed.

Item	Description
-R *ConfigurationSet*	Erases configuration set *ConfigurationSet* (removes from *ConfigurationSet* all configuration/time range pairs). This operation is not recommended as the resulting configuration set state is not consistent and requires additional changes.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

The following examples demonstrate how to display, change, and use WLM configurations using the **lswlmconf** command, the **confsetcntrl** command, the **wlmcheck** command, and the **wlmcntrl** command.

1. To find the WLM configurations, type:

   ```
   lswlmconf
   ```

   The output to this command might look similar to the following:

   ```
   standard
   template
   fvtrules
   fvtlimits
   fvtregul
   fvtdfct
   fvtsynt
   fvtthreads
   ```

2. To show the current WLM configuration, type:

   ```
   lswlmconf -c
   ```

   The output might look similar to the following:

   ```
   fvtlimits
   ```

3. To show configuration sets, use the **lswlmconf** with the **-s** flag, type:

   ```
   lswlmconf -s
   ```

   Since this example configuration contains no configuration sets, this command produces a message indicating that no matching configuration was found.

4. In order to create a configuration set using "standard" as the default configuration, type:

   ```
   confsetcntrl -C confset1 standard
   ```

5. To use the **lswlmconf** command to show the new configuration set, type:

   ```
   lswlmconf -s
   ```

   The command now produces the following output:

   ```
   confset1
   ```

6. In order to use the "fvtlimits" configuration for "confset1" on week days (Monday through Friday) by specifying a time range, type:

```
confsetcntrl -d confset1 -a fvtlimits 1-5
```

7. You might want this configuration only in the morning. You cannot change a time range. Instead you must remove the time range and then create a new time range.

   First, remove the old time range, as follows (**confsetcntrl** accepts day names, as reported by "**locale day**" or "**locale abday**" commands):

   ```
   confsetcntrl -d confset1 -r fvtlimits monday-friday
   ```

   Then create the new time range, as follows:

   ```
   confsetcntrl -d confset1 -a fvtlimits 1-5,8:00-12:00
   ```

8. In order to add another time range for using the "fvtregul" configuration on Sundays, type:

   ```
   confsetcntrl -d confset1 -a fvtregul 0
   ```

9. In order to display configuration set "confset1", type:

   ```
   confsetcntrl -d confset1
   ```

   In this example, this command produces the following output:

   ```
   fvtlimits:
           time = "1-5,8:00-12:00"

   fvtregul:
           time = "0"

   standard:
           time = "-"
   ```

10. In order to create a configuration set called "confset2" using "template" as the default configuration, type:

    ```
    confsetcntrl -C confset2 template
    ```

    In order change "confset2' so it will use the configuration "fvtsynt" every nigh, type:

    ```
    confsetcntrl -d confset2 -a fvtsynt 18:00-10:00
    ```

11. In order to display the list of regular configurations, type:

    ```
    lswlmconf -r
    ```

    In this example, this produces the following output, (which demonstrates that in this example the list of regular configurations has not changed):

    ```
    standard
    template
    fvtrules
    fvtlimits
    fvtregul
    fvtdfct
    fvtsynt
    fvtthreads
    ```

    However, as expected, the list of configurations sets in this example has changed, as shown by the following command:

    ```
    lswlmconf -s
    ```

    This command produces the following output in this example:

    ```
    confset1
    confset2
    ```

12. In order to show which configuration would be currently active when that the **date** command reports the current time as "Tue Jul 16 18:55:10 EET 2002" with configuration set "confset2", type:

```
lswlmconf -d confset2 -l
```

In this example, this command produces the following output:

```
confset2/fvtsynt
```

You can also show which configurations would be active at another time. To show which configurations would be active on Sunday at 9:00am, type:

```
lswlmconf -l -t 0,9:00
```

This command produces the following output in this example:

```
standard
template
fvtrules
fvtlimits
fvtregul
fvtdfct
fvtsynt
fvtthreads
confset1/fvtregul
confset2/fvtsynt
```

In order to display this information only for configuration sets, type:

```
lswlmconf -s -l -t 0,9:00
```

This produces the following output in this example:

```
confset1/fvtregul
confset2/fvtsynt
```

13. In order to remove configuration set "confset2", type:

```
confsetcntrl -D confset2
```

lswlmconf -s now produces the following output in this example:

```
confset1
```

14. In order to check configuration set "confset1", using the **wlmcheck** command, type:

```
wlmcheck -d confset1
```

In this example, this produces the following output:

```
WLM is not running.
Checking classes and rules for 'confset1' configuration...
fvtlimits/System
fvtlimits/Default
fvtlimits/Shared
fvtlimits/login
fvtregul/System
fvtregul/Default
fvtregul/Shared
standard/System
standard/Default
standard/Shared
```

15. In order to start using configuration set "confset1" used in this example, type:

```
wlmcntrl -a -d confset1
```

The command **lswlmconf -c** now produces the following output:

```
confset1
```

The command **lswlmconf -cl**, which shows the active regular configuration, now produces the following output:

```
confset1/standard
```

Files

The configuration set files reside in a subdirectory of **/etc/wlm** whose name is the set name.

Item	Description
.times	Contains the list of all the configuration/time range pairs of the set.
description	Contains an optional description text of the set.

confsrc Command

Purpose

The **confsrc** command configures a subsystem, a group of subsystems, or a subserver.

Syntax

confsrc [[**-R**] **-h** Host] [**-p** *SubsystemPID*] [**-q**] [**-Q**] [**-u** *UserID*] [**-U** *Password*] **-s** *Subsystem* **-a** *ConfigString*

Description

The **confsrc** command sends a request to the System Resource Controller (SRC) to configure a subsystem. When a request to configure the subserver is passed to the SRC and the subsystem to which the subserver belongs is not active, the SRC starts the subsystem and transmits the request to the subsystem.

Note: The configure subserver request is processed only when the subsystem supports the request.

Flags

Item	Description
-a *ConfigString*	Specifies a string containing the configuration information. This string is passed from the command line and appended to the command-line arguments from the subsystem object class. If the specified string exceeds 1200 characters, the command is unsuccessful. The command argument is passed by the SRC to the subsystem according to the rules used by the shell. Quoted strings are passed as a single argument, and blanks outside a quoted string delimit an argument. Single and double quotes can be used in the string.
-h *Host*	Specifies the foreign host on which the configure action is requested. The local user must be running as root. The remote system must be configured to accept remote SRC requests by starting the **srcmstr** daemon (**/etc/inittab**) with the **-r** flag and configuring the **/etc/hosts.equiv** or**rhosts** file to allow remote requests.

Item	Description
-p *SubsystemPID*	Specifies an instance of the subsystem to which the configure request is passed.
-q	Specifies the quiet mode of operation with minimum local output.
-Q	Specifies the quiet mode of operation with suppressed output.
-R	Uses TCP for remote connections. **Note:** This flag is active only when the **-h** flag is used.
-s *Subsystem*	Specifies the subsystem to be started. The specified *Subsystem* can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the specified *Subsystem* is not contained in the subsystem object class.
-u *UserID*	Specifies the user ID used on the remote host. **Note:** This flag is active only when the **-h** flag is used.
-U *Password*	Specifies the password for the user ID. **Note:** This flag is active only when the **-u** flag is used.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration Object Class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration Object Class.
/etc/services	Defines the sockets and protocols used for Internet services.
/dev/SRC	Specifies the **AF_UNIX** socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.

cp Command

Purpose

Copies files.

Syntax

To copy a file to another file, use the following syntax:

cp [**-d**] [**-e**] [**-E** {force|ignore|warn}] [**-f**] [**-h**] [**-i**] [**-p**] [**-I**] [**-S**] [**-U**] [**--**] *SourceFile TargetFile*

To copy a file to a directory, use the following syntax:

cp [**-d**] [**-e**] [**-E** {force|ignore|warn}] [**-f**] [**-h**] [**-i**] [**-p**] [[**-r** | **-R**] [**-H** | **-L** | **-P**]] [**-I**] [**-U**] [**--**] *SourceFile ... TargetDirectory*

To copy a directory to a directory, use the following syntax:

cp [**-d**] [**-e**] [**-E** {force|ignore|warn}] [**-f**] [**-h**] [**-i**] [**-p**] { **-r** | **-R** } [**-H** | **-L** | **-P**] [**-I**] [**-U**] [**--**] *SourceDirectory ... TargetDirectory*

Description

The **cp** command copies the source file specified by the *SourceFile* parameter to the destination file specified by the *TargetFile* parameter. If the target file exists, **cp** overwrites the contents, but the mode, owner, and group associated with it are not changed. The last access time of the *SourceFile* and the last modification time of the *TargetFile* are set to the time the copy was done. If the *TargetFile* does not exist, **cp** creates a new file named *TargetFile* that has the same mode as the source file except that the sticky bit is not set unless it was done by a superuser; the owner and group of the *TargetFile* is that of the user. When the *TargetFile* is a link to another file, **cp** overwrites the destination link with the content of the source file; the links from the *TargetFile* remains. Also, the **cp** command can copy the source files specified by the *SourceFile* parameter (or directories named by the *SourceDirectory* parameter) to the directory specified by the *TargetDirectory* parameter.

> **Note:** If one of the source parameters is a directory, you need to specify one of the **-r** or **-R** flags.

If any directories are created by the **cp** command during the copying process, the newly created directory will have the same mode as the corresponding source directory.

You can also copy special device files. The preferred option for accomplishing this is the **-R** flag. Specifying **-R** causes the special files to be re-created under the new path name. Specifying the **-r** flag causes the **cp** command to attempt to copy the special file to a regular file.

Note: The I/O buffer size for the read and write system calls generated by this command can be configured by using the *AIX_STDBUFSZ* environment variable.

Flags

Item	Description
-d	Specifies that the source file is stored in decrypted (clear-text) format on target.
-e	Specifies that the source file is stored in encrypted form, if the target file system is an Encrypted File System (EFS).
-E	The **-E** option requires one of the following arguments. If you omit the **-E** option, **warn** is the default behavior. **force** Fails the **cp** operation on a file if the fixed extent size or space reservation of the file cannot be preserved. **ignore** Ignores any errors in preserving extent attributes. **warn** Issues a warning if the space reservation or the fixed extent size of the file cannot be preserved.
-f	Specifies removal of the target file if it cannot be opened for write operations. The removal precedes any copying performed by the **cp** command.
-h	Forces the **cp** command to copy symbolic links. The default is to follow symbolic links, that is, to copy files to which symbolic links point.
-H	Take actions based on the type and contents of the file referenced by any symbolic link specified as a *SourceFile* operand.

Item	Description
-i	Prompts you with the name of a file to be overwritten. This occurs if the *TargetDirectory* or *TargetFile* parameter contains a file with the same name as a file specified in the *SourceFile* or *SourceDirectory* parameter. If you enter y or the locale's equivalent of y, the **cp** command continues. Any other answer prevents the **cp** command from overwriting the file.
-I	Suppresses the warning message during ACL conversion.
-L	Take actions based on the type and contents of the file referenced by any symbolic link specified as a *SourceFile* operand or any symbolic links encountered during traversal of a file hierarchy.
-p	Duplicates the following characteristics of each *SourceFile*/*SourceDirectory* in the corresponding *TargetFile* and/or *TargetDirectory*: • The time of the last data modification and the time of the last access. If this duplication fails for any reason, the **cp** command will write a diagnostic message to standard error. The **nanoseconds** field of the *SourceFile*/*SourceDirectory* is not duplicated for last modification time or last access time. • The user ID and group ID. If this duplication fails for any reason, the **cp** command may write a diagnostic message to standard error. • The file permission bits and the S_ISUID and S_ISGID bits. If this duplication fails for any reason, the **cp** command will write a diagnostic message to standard error. If the user ID or group ID cannot be duplicated, the file permission bits S_ISUID and S_ISGID are cleared. In order to preserve the owner ID and group ID, permission modes, modification and access times, user must have the appropriate file access permissions (user should be a superuser or have the same owner ID as the destination file) The target file will not be deleted if these characteristics cannot be preserved. Access control lists (ACLs) associated with the *SourceFile* are preserved if the target filesystem supports the same. If the source file contains NFS4 ACL and the target filesystem does not support NFS4 ACL, the NFS4 ACL is converted to AIXC. When ACL conversion succeeds, a warning message is printed out the stderr. If the source file is encrypted and the **-p** flag is specified, the **cp** command preserves the EFS information. Generally, the **-e** or **-d** flag takes precedence over the **-p** flag. If a user requests to convert a clear-text file to an encrypted format using the **-e** flag, then even if the user specifies the **-p** flag, the copy does not preserve attributes like the time of the last data modification, the time of the last access and so on. As long as the encryption or decryption status remains the same, the **-p** flag preserves the file attributes and EFS information.
-P	Take actions on any symbolic link specified as a *SourceFile* operand or any symbolic link encountered during traversal of a file hierarchy.
-r	Copies file hierarchies under the file or directory specified by the *SourceFile* or *SourceDirectory* parameter (recursive copy). The **-r** flag processes special files in the same manner as regular files.

Item	Description
-R	Copies file hierarchies under the regular files and directories from the directory specified by the *SourceFile* or *SourceDirectory* parameter to the directory specified by the *TargetDirectory* parameter. Special file types, such as first-in, first-out (FIFO) files and block and character device files, are re-created instead of copied. Symbolic links are followed unless the **-h** flag is specified. (The **-R** flag is preferred to the **-r** flag.) If none of the **-H**, **-L**, or **-P** options were specified, it is unspecified which of those options will be used as the default. Consider the following: • If the **-H** option was specified, the **cp** command will take action based on the type and contents of the file referenced by any symbolic link specified as a *SourceFile* operand. • If the **-L** option was specified, the **cp** command will take action based on the type and contents of the file referenced by any symbolic link specified as a *SourceFile* operand or any symbolic links encountered during traversal of a file hierarchy. • If the **-P** option was specified, the **cp** command will copy any symbolic link specified as a *SourceFile* operand and any symbolic links encountered during traversal of a file hierarchy and will not follow any symbolic links.
-S	Preserves the sparseness of the source file while you copy it to the destination file if the source file is sparse. A *sparse file* is a type of computer file that attempts to use the file system space more efficiently when the blocks that are allocated to the file are mostly empty.
-U	Copies Extended Attributes (EA), Access Control Lists (ACL) in the *SourceFile* to the *TargetFile*. If the EA is not supported on the target filesystem then it is ignored. If the source ACL type is not supported on the target filesystem then it is converted to the compatible ACL type supported by the target filesystem.
--	Indicates that parameters following the **--** (dash, dash) flag are to be interpreted as file names. This null flag allows the specification of file names that start with a **-** (minus sign).

The following table shows the encryption or decryption status of the target file under different conditions:

Explicit flag for the cp command	**Source file**	**Target file system**	**Result**
-e (encrypted)	Non-EFS	Non-EFS	Error
-e	Non-EFS	EFS	Encrypted file
-e	EFS	EFS	Encrypted file
-e	EFS	Non-EFS	Error
-d (decrypted)	Non-EFS	Non-EFS	Clear-text file
-d	Non-EFS	EFS	Clear-text file
-d	EFS	Non-EFS	Clear-text file
-d	EFS	EFS	Clear-text file
No explicit flag	Non-EFS	Non-EFS	Clear-text file
No explicit flag	Non-EFS	EFS	If the target directory is EFS inheritance enabled, the target file is an encrypted file. Otherwise the target file is a clear-text file.

Explicit flag for the cp command	Source file	Target file system	Result
No explicit flag	EFS	EFS	Encrypted file
No explicit flag	EFS	Non-EFS	Error

Note: It is not permitted to overwrite an encrypted file with a plain-text file and vice versa unless you specify the **-f** flag. The encryption status of the target depends on the **-e** or **-d** flag, the encryption inheritance if you do not specify the **-e** or **-d** flag with the **-f** flag, and the encryption status of the source file if the encryption inheritance is not active.

Exit Status

This command returns the following exit values:

Item	Description
0	All files were copied successfully.
>0	An error occurred.

Examples

1. To make a copy of a file in the current directory, enter:

   ```
   cp prog.c prog.bak
   ```

   This copies `prog.c` to `prog.bak`. If the `prog.bak` file does not already exist, the **cp** command creates it. If it does exist, the **cp** command replaces it with a copy of the `prog.c` file.
2. To copy a file in your current directory into another directory, enter:

   ```
   cp jones /home/nick/clients
   ```

   This copies the `jones` file to `/home/nick/clients/jones`.
3. To copy a file to a new file and preserve the modification date, time, and access control list associated with the source file, enter:

   ```
   cp -p smith smith.jr
   ```

   This copies the `smith` file to the `smith.jr` file. Instead of creating the file with the current date and time stamp, the system gives the `smith.jr` file the same date and time as the `smith` file. The `smith.jr` file also inherits the `smith` file's access control protection.
4. To copy all the files in a directory to a new directory, enter:

   ```
   cp /home/janet/clients/* /home/nick/customers
   ```

   This copies only the files in the `clients` directory to the `customers` directory.
5. To copy a directory, including all its files and subdirectories, to another directory, enter:

   ```
   cp /home/janet/clients/* /home/nick/customers
   ```

   Note: A directory cannot be copied into itself.

   This copies the `clients` directory, including all its files, subdirectories, and the files in those subdirectories, to the `customers/clients` directory.
6. To copy a specific set of files to another directory, enter:

   ```
   cp jones lewis smith /home/nick/clients
   ```

This copies the `jones`, `lewis`, and `smith` files in your current working directory to the `/home/nick/clients` directory.

7. To use pattern-matching characters to copy files, enter:

```
cp programs/*.c .
```

This copies the files in the `programs` directory that end with `.c` to the current directory, signified by the single `.` (dot). You must type a space between the c and the final dot.

8. To copy a file to a new file and preserve the ACL and EA associated with the source file, enter:

```
cp -U smith smith.jr
```

9. To preserve the sparseness of the source file while you copy, enter:

```
cp -S file.c sparse_file.c
```

Files

/usr/bin/cp
Contains the **cp** command.

cp_bos_updates Command

Purpose

Restores the root files from the `bos.rte*` software updates to the system.

Syntax

cp_bos_updates -d <*device*>

Description

The **cp_bos_updates** command creates and populates directories for the `bos.rte*` software updates root part files (`inst_root` paths). The directories are created and populated only for the updates at the same version.release.modification.fix (VRMF) level as that of the software during the time of the original operating system installation. During installation of the AIX Version 6 with the 7100-02 Technology Level or AIX Version 6 with the 6100-08 Technology Level, the command is called and the directories are created automatically. A log file containing the **cp_bos_updates** output from the operating system installation is saved in the `/var/adm/ras/cp_bos_updates.log` file. If the system is base installed before this support and then upgraded to a level that supports the **cp_bos_updates** command, the command can be run manually to create and populate these directories for the user. The resultant directories are only needed if you are upgrading a WPAR that is copied or restored (by using the **restwpar** command) from a different system that has a different level of the base operating system.

Flag

Item	Description
-d *device*	The device can be a directory or an optical device, such as `/dev/cd0`.

Files

Item	Description
`/usr/sbin/cp_bos_updates`	The **cp_bos_updates** command.

Examples

1. If the operating system was originally installed at AIX 6 with the 6100-06 Technology Level (run `lslpp -ah bos.rte.install` to get the original VRMF, which in this case will be 6.1.6.0), insert the Base Media from that level of AIX into the DVD drive, /dev/cd0, and type the following command:

   ```
   cp_bos_updates -d /dev/cd0
   ```

2. If the operating system is originally installed from a NIM lpp_source that was created from the AIX 6 with 6100-06 base media, and had no additional service packs added to the lpp_source, then mount that lpp_source onto the system at `/mnt/6100_06`, and type the following command:

   ```
   cp_bos_updates -d /mnt/6100_06
   ```

   Note: If a NIM lpp_source is created from the AIX 6 with 6100-06 base media and had subsequent service packs added to the lpp_source, and the **lppmgr** command is run against the lpp_source to eliminate unnecessary software images, some of the required updates at the base level VRMF are removed. You must either must find the AIX 6 with 6100-06 base media, or download the AIX Version 6 with 6100-06 Technology Level, for using the **cp_bos_updates** command.

cpcosi Command

Purpose

Clones a Common Operating System Image (COSI).

Syntax

cpcosi **-c** *COSI* [**-S** *Server*] [**-l** *Location*] [**-v**] *COSI*

Description

The `cpcosi` command clones a Common Operating System Image (COSI). A COSI is a repository that contains all the software necessary to bring up a system to a functional state. The `mkcosi` command creates the COSI.

The `cpcosi` command takes a common image and attempts to make a duplicate copy of it. The copied version is stored at the location specified with the `-l` flag. If the `-l` flag is not specified, the location of the originating common image is used instead. If the `-S` flag is specified, the clone common image is stored on that particular server. The `-S` flag must point to a machine that is managed by the caller of the `cpcosi` command. The naming convention for the clone is the original common image name suffixed with an _X{*count*}, where *count* is a number that is incremented every time a common image is cloned.

A common image must exist on the system before it can be cloned. Use the `mkcosi` command to create a common image. The `lscosi` command lists any common images that exist in the environment. The `lscosi` command depends on the `bos.sysmgt.nim.master` fileset being present on the system.

Flags

Item	Description
-c	Specifies the COSI to clone.
`-l` *Location*	Specifies the full path name to a location for storing the COSI.
`-S` *Server*	Specifies the name of the machine on which the COSI image will reside.

Item	Description
`-v`	Enables verbose debug output when the `cpcosi` command runs.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

Access Control: You must have root authority to run the `cpcosi` command.

Examples

1. To clone a COSI named `cosi2` from a COSI named `cosi1`, enter:

   ```
   cpcosi -c cosi1 cosi2
   ```

   Because no location path was specified in the preceding example, if `cosi1` was stored at `/export/cosi1`, the cloned COSI will be placed in `/export/cosi2`.

Location

`/usr/sbin/cpcosi`

Files

Item	Description
`/etc/niminfo`	Contains variables used by NIM.

cpio Command

Purpose

Copies files into and out of archive storage and directories. This document describes the AIX **cpio** command and the System V **cpio** command.

Syntax

cpio -o [**a**] [**c**] [**-E**{**force**|**ignore**|**warn**}] [**-g**] [**-H** *hdr*][-U] [**v**] [**B** | **C** *Value*] [**-Z**] <*FileName* >*Output*

cpio -i [**b**] [**c**] [**d**] [**-E**{**force**|**ignore**|**warn**}] [**f**] [**-H** *hdr*] [**m**] [**M**] [**r**] [**s**] [**t**] [-U] [**u**] [**v**] [**S**] [**6**] [**B** | **C** *Value*] [**-Z**] [*Pattern...]* <*Input*

cpio -p [**a**] [**d**] [**-E**{**force**|**ignore**|**warn**}] [**l**] [**m**] [**M**] [-U] [**u**] [**v**] [**-Z**] *Directory* <*FileName*

Description

Attention: If you redirect the output from the **cpio** command to a special file (device), you should redirect it to the raw device and not the block device. Because writing to a block device is done asynchronously, there is no way to know if the end of the device is reached.

Note:

1. The **cpio** command is not enabled for files greater than 2GB in size due to limitations imposed by XPG/4 and POSIX.2 standards.
2. The **cpio** command does not preserve the sparse nature of any file that is sparsely allocated. Any file that was originally sparse before the restoration will have all space allocated within the file system for the size of the file.
3. You cannot use the System V **cpio** command for Encrypted File Systems.
4. The I/O buffer size for the read and write system calls generated by this command can be configured by using the *AIX_STDBUFSZ* environment variable.

cpio -o Command

The **cpio -o** command reads file path names from standard input and copies these files to standard output, along with path names and status information. Avoid giving the **cpio** command path names made up of many uniquely linked files, as it may not have enough memory to keep track of them and would lose linking information.

cpio -i Command

The **cpio -i** command reads from standard input an archive file created by the **cpio -o** command and copies from it the files with names that match the *Pattern* parameter. These files are copied into the current directory tree. You can list more than one *Pattern* parameter, using the file name notation described in the **ksh** command. Note that in this application the special characters * (asterisk), ? (question mark), and [...] (brackets and ellipses) match the / (slash) in path names, in addition to their use as described in the **ksh** command**.** The default for the *Pattern* parameter is an * (asterisk), selecting all files in the Input. In an expression such as [a-z], the minus sign means *through* according to the current collating sequence.

A collating sequence can define equivalence classes for use in character ranges.

cpio -p Command

The **cpio -p** command reads file path names from standard input and copies these files into the directory named by the *Directory* parameter. The specified directory must already exist. If these path names include directory names that do not already exist, you must use the **d** flag to cause the specified directory to be created.

Note: You can copy special files only if you have root user authority.

cpio -U command

For AIX 5.3, the `cpio` command will ignore extended attributes by default. The `-U` option informs `cpio` to archive or restore attributes, which includes ACLs.

A new record type is required for extended attribute entries in `cpio` archive files. A new record type is also required for ACL entries in `cpio` archive files.

Each object in the `cpio` archive contains a `cpio` header followed by the data for the specified object.

The following table describes the `cpio` header for default binary format and the **-c** format::

Name of field	Size (number of bytes)	Use
h_magic	2	Magic number for identifying header.
h_dev	2	Device that contains a directory entry for this file.
h_ino	2	Inode number that identifies the input file to the file system.
h_mode	2	Mode of the input file, as defined in the mode.h file. The POSIX standard has 0130000, 0150000 - 0170000 available for file types that are not to be transported to other systems.
h_uid	2	User ID of the owner of the input file.

Name of field	Size (number of bytes)	Use
h_gid	2	Group ID of the owner of the input file.
h_nlink	2	Number of links that are connected to the input file.
h_rdev	2	ID of the remote device from which the input file is taken.
h_mtime	4	Time when data was last modified.
h_namesize	2	Length of the pathname including the NULL.
h_filesize	4	Length of the file in bytes.
h_name	PATH_MAX	Null-terminated pathname.

Each file which has an ACL will have a <header,data> object immediately preceding the object itself which describes the ACL as follows:

Header for ACL
The h_mode field set to 0130000 indicates the header describes an ACL. Additionally, the h_mode bits are set to indicate who can write the ACL. All other fields in the cpio header are set as for the inode of the file owning the ACL.

Data
The data will be the ACL itself. The first 64-bits of the data will be the ACL type. It will be immediately followed by the ACL value.

Each extended attribute will have a single <header,data> object in the archive which completely describes the extended attribute as follows:

Header for EA
The h_mode field set to 0150000 indicates an extended attribute header. All fields in the cpio header are set as for the inode of the extended attribute. Except the h_name field is set to <NULL><*EAName*><NULL>

Data:
This is formatted to describe the owner of the extended attribute as well as the data for the extended attribute. There is a eaHeader followed by the pathname of the owner of the extended attribute, followed by the extended attribute data.

```
struct eaHeader {
    char    pathLen[12];
    char    dataLen[12];
};
```

Parameters

Item	Description
Directory	Specifies the directory.
<*FileName*	Specifies a list of file names for the **cpio** command to use as input.
>*Output*	Specifies the output device such as a diskette or file. For more information on using tape devices see the **rmt** special file.
<*Input*	Specifies the input device (where *Input* is the *Output* file created by the **cpio -o** command). For more information on using tape devices, see the **rmt** special file.
Pattern	Specifies the pattern (as described in the **ksh** command) to be used with the command. The default for the *Pattern*parameter is an * (asterisk), selecting all the files in the *Input*.

Flags

All flags must be listed together, without any blanks between them. Not all of the following flags can be used with each of the **-o**, **-i**, and **-p** flags.

Item	Description
a	Resets the access times of the source files to their previous times.
b	Swaps both bytes and halfwords. **Note:** If there is an odd number of bytes or halfwords in the file being processed, data can be lost.
B	Performs block input and output using 512 bytes to a record. **Note:** When using the **B** or **C** options to extract or create a tape archive, the blocking factor must be a multiple of the physical block size for that tape device. When using the **B** or **C** options to extract an archive from tape, the blocking factor should not be larger than the size of the archive as it exists on the tape. The **B** flag and the **C** flag are mutually exclusive. If you list both, the **cpio** command uses the last one it encounters in the flag list.
c	Reads and writes header information in ASCII character form. If a **cpio** archive was created using the **c** flag, it must be extracted with **c** flag.
C *Value*	Performs block input and output using the *Value* parameter times 512 bytes to a record. For instance, a **-C2** flag changes the block input and output sizes to 1024 bytes to a record.
d	Creates directories as needed.
-E	The **-E** option requires one of the following arguments. If you omit the **-E** option, **warn** is the default **cpio** behavior. **force** Fails the extract or copy operation on a file if the file's extent attributes cannot be preserved. **ignore** Ignores any errors in preserving extent attributes. **warn** Issues a warning if the space reservation or the fixed extent size of the file cannot be preserved. This is the default behavior.
f	Copies all files except those matching the *Pattern* parameter.
g	Allows the large UID or GID (> USHORT_MAX) values while archiving. **Note:** The environment variable can also be used for the same. **Usage** `Export CPIO_LARGE_UID=ON`

Item	Description
H	Reads or writes header information in *hdr* format. Either the **-H** or **-c** option can be used when the target and the destination computers are of different types. This option is mutually exclusive with the **-c** and **-6** options. This format allows system interoperability and portability. The **cpio** utility supports the archival of files larger than 2 GB in size when the CRC (-Hcrc) format is used. If a **cpio** archive is created by using the **H** flag, it must be extracted with the **H** flag. The valid values for the *hdr* variable are: **crc** Same as CRC. ASCII header with an additional per-file checksum. The crc file format handle files larger than 2 GB and maximum size supported is 4 GB. **odc** ASCII header with small fundamental types. **newc** The new ASCII portable format.
l	Links files rather than copying them, whenever possible. This flag can only be used with the **cpio -p** command.
m	Retains previous file modification time. This flag does not work when copying directories.
M	Retains previous file modification time even when directories are copied.
r	Renames files interactively. If you do not want to change the file name, enter a single period or press the <Enter> key. In the latter case, the **cpio** command does not copy the file.
s	Swaps bytes. This flag is used only with the **cpio -i** command. **Note:** If there is an odd number of bytes in the file being processed, data can be lost.
S	Swaps halfwords. This flag is usable only with the **cpio -i** command. **Note:** If there is an odd number of halfwords in the file being processed, data can be lost.
t	Creates a table of contents. This operation does not copy any files.
-U	Performs archival and extraction of ACL and Extended Attributes. Attributes include Access control list (ACL) also. If the ACL type is not supported on the *Target* filesystem then it is converted to the ACL type supported by the *Target* filesystem. If the EA is not supported on the filesystem then it is not copied.
u	Copies unconditionally. An older file now replaces a newer file with the same name.
v	Lists file names. If you use this with the **t** flag, the output looks similar to that of the **ls -l** command.
6	Processes an old file (for example, one written in UNIX Sixth Edition format). This flag is usable only with the **cpio -i** command.
-Z	Archives the Encrypted File System (EFS) information of encrypted files or directories. The EFS information is extracted. When you specify the **-t** and **-v** flags along with the **-Z** flag, an **e** indicator is displayed after the file mode for encrypted files and directories that were archived with the **-Z** flag, and a hyphen (-) is displayed for other files. **Note:** Archives created with the **-Z** flag can be restored only on AIX 6.1 or later releases.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To copy files onto diskette, enter:

   ```
   cpio -ov <filenames >/dev/rfd0
   ```

   This copies the files with path names listed in the `filenames` file in a compact form onto the diskette (`>/dev/rfd0`). The **v** flag causes the **cpio** command to display the name of each file as it is copied. This command is useful for making backup copies of files. The diskette must already be formatted, but it must not contain a file system or be mounted.

   > **Note:** Files with uid's and gid's greater than 65535 cannot be archived using the **cpio** command. In such instances, the user should use backup and restore.

2. To copy files in the current directory onto diskette, enter:

   ```
   ls *.c | cpio -ov >/dev/rfd0
   ```

   This copies all the files in the current directory whose names end with `.c`

3. To copy the current directory and all subdirectories onto diskette, enter:

   ```
   find . -print | cpio -ov >/dev/rfd0
   ```

   This saves the directory tree that starts with the current directory (`.`) and includes all of its subdirectories and files. Do this faster by entering:

   ```
   find . -cpio /dev/rfd0 -print
   ```

   The `-print` entry displays the name of each file as it is copied.

4. To list the files that have been saved onto a diskette with the **cpio** command, enter:

   ```
   cpio -itv </dev/rfd0
   ```

   This displays the table of contents of the data previously saved onto the `/dev/rfd0` file in the **cpio** command format. The listing is similar to the long directory listing produced by the **ls -l** command. To list only the file path names, use only the **-it** flags.

5. To copy the files previously saved with the **cpio** command from a diskette, enter:

   ```
   cpio -idmv </dev/rfd0
   ```

   This copies the files previously saved onto the `/dev/rfd0` file by the **cpio** command back into the file system (specify the **-i** flag). The **d** flag allows the **cpio** command to create the appropriate directories if a directory tree is saved. The **m** flag maintains the last modification time in effect when the files are saved. The **v** flag causes the **cpio** command to display the name of each file as it is copied.

6. To copy selected files from diskette, enter:

   ```
   cpio -i "*.c" "*.o" </dev/rfd0
   ```

This copies the files that end with `.c` or `.o` from diskette. Note that the patterns `"*.c"` and `"*.o"` must be enclosed in quotation marks to prevent the shell from treating the * (asterisk) as a pattern-matching character. This is a special case in which the **cpio** command itself decodes the pattern-matching characters.

7. To rename files as they are copied from diskette, enter:

```
cpio -ir </dev/rfd0
```

The **-r** flag causes the **cpio** command to ask you whether to rename each file before copying it from diskette. For example, the message:

```
Rename <prog.c>
```

asks whether to give the file saved as `prog.c` a new name as it is copied. To rename the file, type the new name and press the Enter key. To keep the same name, you must enter the name again. To avoid copying the file at all, press the Enter key.

8. To copy a directory and all of its subdirectories, enter:

```
mkdir /home/jim/newdir
find . -print | cpio -pdl /home/jim/newdir
```

This duplicates the current directory tree, including the current directory and all of its subdirectories and files. The duplicate is placed in the new `/home/jim/newdir` directory. The **l** flag causes the **cpio** command to link files instead of copying them, when possible.

> **Note:** The performance of **cpio** to the 9348 Magnetic Tape Unit Model 12 can be improved by changing the default block size. To change the block size, enter the following at the command line:
>
> ```
> chdev -1 <device_name> -a block_size=32k
> ```

9. To copy files in the current directory onto diskette and preserve the ACL and EA associated with the files, enter:

```
ls *.c | cpio -oUv >/dev/rfd0
```

Files

Item	Description
/usr/bin/cpio	Contains the **cpio** command.

System V cpio Command

Purpose (System V cpio Command)

Copies files into and out of archive storage and directories.

Syntax (System V cpio Command)

cpio -i [**-b**] [**-B**] [**-c**] [**-d**] [**-f**] [**-k**] [**-m**] [**-r**] [**-s**] [**-S**] [**-T**] [**-t**] [**-u**] [**-v**] [**-V**] [**–6**] [**-C** *bufsize*] [**-E** *file*] [**-H** *hdr*] [**-I** *file* [**-M** *message*]] [**-R** *ID*]] [*Patterns* ...]

cpio -o [**-a**] [**-A**] [**-B**] [**-c**] [**-L**] [**-v**] [**-V**] [**-C** *bufsize*] [**-H** *hdr*] [**-K** *mediasize*] [**-O** *file* [**-M** *message*]]

cpio -p [**-a**] [**-d**] [**-l**] [**-L**] [**-m**] [**-u**] [**-v**] [**-V**] [**-R** *ID*] *Directory*

Description (System V cpio Command)

The **cpio** command copies files into and out of an archive. The **-i**, **-o** and **-p** options select the action to be performed. The following list describes each of the actions. The **-o**, **-p** and **-i** options are mutually exclusive.

cpio -i (copy in)

cpio -i (copy in) extracts files from the standard input (only if **-I** is not specified), which is assumed to be the product of a previous **cpio -o**. Only files with names that match *Patterns* are selected. *Patterns* are regular expressions given in the filename generating notation of **ksh**. In *Patterns*, meta-characters "?", "*", and "[. . .]" match the slash ("/") character, and backslash ("\") is an escape character. A "!" meta-character means not. (For example, the "[!abc]" pattern would exclude all files that begin with either a, b or c.) Multiple patterns may be specified and if no patterns are specified, the default for *Patterns* is "*" (that is, select all files). Each pattern must be enclosed in double quotes; otherwise, the name of a file in the current directory might be used. Extracted files are conditionally created and copied into the current directory tree based on the options described below.

The **cpio -i** command reads the standard input of an archive file created that was using the **cpio -o** command, and copies the files with names that match the *pattern* parameter. The *pattern* parameter is a regular expression given with general notation of **ksh**. These files are copied into the current directory tree. More than one pattern parameter can be used, using the file name notation described in the **ksh** command. The patterns can be special characters * (asterisk), ? (question mark), and [...] (brackets and ellipses). The default for the pattern parameter is an * (asterisk), selecting all files in the input. In an expression such as [a-z], the minus sign means `through` according to the current collating sequence.

The permissions of the files will be those of the previous **cpio -o**. Owner and group permissions will be the same as the current user unless the current user is the root user. If this is true, owner and group permissions will be the same as those resulting from the previous **cpio -o**. Blocks are reported in 512-byte quantities.

If **cpio -i** tries to create a file that already exists and the existing file is the same age or younger (newer), **cpio** will output a warning message and not replace the file. On the other hand if the file being extracted is older than the one in the cpio archive then the existing file will be replaced without any warning from the command.

cpio -o (copy out)

cpio -o reads the standard input to obtain a list of path names and copies those files onto the standard output together with path name and status information.

cpio -p (copy pass)

cpio -p reads the standard input to obtain a list of path names of files and copies these files into the directory named by the *Directory* parameter. The specified directory must already exist. If these path names include directory names that do not already exist, you must use the d flag to cause the specified directory to be created. By default the Access Control List's (ACL) are transferred [copied] from source file to destination file with this option only.

Flags (System V cpio Command)

Item	Description
-a	Resets the access time of the source files to their previous times.
-A	Appends files to an archive. The **-A** option requires the **-O** option. The append option **-A** is not valid for the **rmt** special file and diskettes.
-b	Reverse the order of the bytes within each word. This option is valid only with the **-i** option.
-B	The default buffer size is 512 bytes when neither this nor the **-C** option is used. But when **-B** flag is used the buffer size is set to 5120 bytes block for the Input/Output operations.

Item	Description
-c	Read or write header information in ASCII character form for system interoperability and portability. The **-c** option is mutually exclusive with **-H** and **-6**. Either the **-c** or **-H** option can be used when the target and destination machines are different types.
-C *bufsize*	The block size for Input/Output operation is set to *bufsize*, where *bufsize* indicates the buffer size in positive integer. If used with **-K**, *bufsize* must be a multiple of 1K.
-d	Creates directories as needed.
-E *file*	Specify an input file (*file*) that contains a list of file names to be extracted from the archive with one file name per line.
-f	Copy in all files except those in *Pattern* parameter.
-H *hdr*	Read or write header information in hdr format. Either the **-h** or **-c** option can be used when the target and the destination machines are different types. This option is mutually exclusive with the **-c** and **-6** options. This format allows system interoperability and portability. The **cpio** utility supports the archival of files larger than 2 GB in size when using the ASCII (**-c**), CRC (**-Hcrc**), tar (**-Htar**), or ustar (**- Hustar**) formats. Valid values for *hdr* are: **crc** Same as **CRC**. ASCII header with an additional per-file checksum. The **crc** file format will handle files larger than 2 GB. **ustar** Same as **USTAR**. IEEE/P1003 Data Interchange Standard header and format. **tar** Same as **TAR**. Tar header and format. The **tar** format is provided for compatibility with the **tar** program. **odc** ASCII header with small fundamental types.
-I *file*	Read the contents of *file* as an input archive. If *file* is a character special device, and the current medium has been completely read, replace the medium and press the Enter key to continue to the next medium. This option is valid only with the **-i** option.
-k	Attempt to skip corrupted file headers and I/O errors that may be encountered. This option lets the user read only those files with good headers if files from a medium that is corrupted. This option is valid only with the **-i** option.
-K *mediasize*	Specify the media size as a multiple of 1K. If used with **-C** *bufsize*, then *bufsize* must be a multiple of 1K.
-l	Hard links files rather than copying them, whenever possible. If a file cannot be linked, then it will be copied. This option is valid only with **-p** option.
-L	This option assists in copying the files rather than linking. The content of the link file is copied with the links name. Without **-L** or **-l** option, the symbolic links will be maintained as is default with **-p**.
-m	Retain previous file modification time. The modification time and access time of a restored file is set to the modification time of the file when it was backed up. Modification time of directories is not retained.
-M *message*	Define a message to use when switching media. When the **-O** or **-I** options are given cpio on a special device, this option can be used to define the message that is printed when you reach the end of the medium. A **%d** can be placed in message to print the sequence number of the next medium needed to continue.

Item	Description
-O *file*	Direct the output of **cpio** to *file*. If *file* is a special device and the current medium is full, replace the medium and type Enter to continue to the next medium. This option is valid only with the **-o** option.
-r	Renames files interactively. To skip a file, type Enter. To retain the original path name, type . (period). This option is valid only with the **-i** option.
-R *ID*	Reassigns ownership and group information for each file to a valid user *ID*. This option is valid only for the root user.
-s	Swap bytes within each half word. **Note:** The **-s** and the **-S** flags are basically for byte sequencing.
-S	Swap half words within each word. **Note:** The **-s** and the **-S** flags are basically for byte sequencing.
-t	Creates a table of contents. This operation does not create any files. The **-t** flag and the **-V** flag are mutually exclusive.
-T	Truncates long file names to 14 characters. This option is valid only with the **-i** option.
-u	Copies unconditionally (normally, an older file will not replace a newer file with the same name).
-v	This is the verbose option that causes a list of file names to be printed. When used with the **-t** option, the table of contents looks like the output of an **ls -l** command.
-V	This is a special verbose option that allows to print a dot for each file read or written. Useful to assure the user that cpio is working without printing out all file names. Note that the **-V** and **-v** options are mutually exclusive and whichever occurs earlier in the command line will be processed accordingly ignoring the other.
-6	Processes a UNIX System Sixth Edition archive format file. This option is mutually exclusive with the **-c** and **-H** options.

Parameters (System V cpio Command)

Item	Description
Directory	Specifies the directory.
Patterns	Specifies one or more patterns (as described in the **ksh** command) to be used with the command. The default for the *Patterns* parameter is an * (asterisk), selecting all the files in the input.

Exit Status (System V cpio Command)

0
The command completed successfully.

>0
An error occurred.

Examples (System V cpio Command)

1. To copy all the files in the current directory onto tape device **/dev/rmt0**, enter:

```
find . | /usr/sysv/bin/cpio -oc >/dev/rmt0
```

The **-c** option ensures that the file is made portable to other machines. Instead of **find** you can also use **ls**, **cat**, **echo** and so on to pipe a list of names to **cpio**. The output could also be redirected to a regular **cpio** file instead of a device.

2. To extract an **cpio** archive file named "arfile" created by **cpio** command use the following:

```
/usr/sysv/bin/cpio -icdI arfile
```

   Here all the files are extracted from the **cpio** archive and the **-d** option ensures that the required directory paths are created as when required.

3. A **cpio** archive file can also be extracted as follows:

```
/usr/sysv/bin/cpio -icd < arfile
```

   The **-d** option ensures that all the required directories are created under the current directory. The standard input can be used only if **-I** flag is not specified.

4. To extract unconditionally all the files in "arfile" use the following:

```
/usr/sysv/bin/cpio -icduI arfile
```

5. To skip any files which corrupted headers, **cpio** can be used as follows:

```
/usr/sysv/bin/cpio -ickudI arfile
```

6. If the access time of the files archived needs to be reset when cpio is used to create an archive, use **cpio** in the following way:

```
ls | /usr/sysv/bin/cpio -oca > arfile
```

7. To extract only the files matching the pattern "a*" from the archive "ar", use the following:

```
cat ar | /usr/sysv/bin/cpio -ickud "a*"
```

   This command extracts all the files starting with letter "a".

8. To display the list of files archived, use **cpio** in the following way:

```
cat ar | /usr/sysv/bin/cpio -itv
```

   The verbose option (**-v**) ensures that the list given by **-t** option is listed in a very similar way as **ls -l** command.

9. The **cpio -p** command can be used to copy a directory tree to a new path, as follows:

```
find . -print | /usr/sysv/bin/cpio -pd /home/user1/newdir
```

   The entire directory tree from current directory is copied to **/home/user1/newdir**. The **-d** option ensures that directories are created as necessary.

10. To retain the modification time and access control list while copying the directory tree, use the **cpio** command as follows:

```
find . -name "*.o" -print | /usr/sysv/bin/cpio -pdlmv /home/user1/newdir
```

    In this example only the **.o** files under the directory tree are copied to **/home/user1/newdir**.

11. To append a list of files to a **cpio** archive matching a particular pattern, invoke a command similar to the following:

```
ls d* | /usr/sysv/bin/cpio -oAO /tmp/ar
```

    In this example, all files starting with "d" in the current directory will be appended to the **cpio** archive.

12. To extract only a list of files listed inside a regular file from an **cpio** archive, use the following command:

    ```
    cat ar | /usr/sysv/bin/cpio -i -E Efile
    ```

    In this example, **cpio** extracts only those files that are listed in the regular file "Efile", provided the specified file name exists in the archive.
13. To hard link all the files instead of copying them, invoke a command similar to the following:

    ```
    ls d* | /usr/sysv/bin/cpio -pdl /home/user2/newdir
    ```

    In this example, the **-l** flag ensures all the file names starting with the character "d" are hard linked to the **/home/user2/newdir**, the directory specified. Hard linking across file systems is not allowed, thus the **-l** option cannot be used when the destination directory is in any other filesystem.

Files (System V cpio Command)

Item	Description
/usr/sysv/bin/cpio	Contains the System V **cpio** command.

cpiscsi Command

Purpose

Copies the configuration settings of internet Small Computer System Interface (iSCSI) target devices from a device to another device.

Syntax

```
cpiscsi -l destination_device -s source_device
```

Description

When you use multiple iSCSI software initiator devices, you can use the **cpiscsi** command to copy the configuration settings of the iSCSI target devices from an initiator device to another initiator device. If two devices must access the same target devices, copying the configuration settings from the source device to the destination device is the easiest method to configure the destination device.

When you run the **cpiscsi** command, all the configuration settings of the iSCSI target devices from the source device are copied. The configuration settings include any specified Challenge Handshake Authentication Protocol (CHAP) user names and passwords. If the configuration settings for the destination device are similar but not the same as the source device, you can first use the **cpiscsi** command and then use the **chiscsi** command for minor changes. Existing configuration settings for the destination device are not changed.

Flags

-l *destination_device*
Specifies the name of the destination device to which the configuration settings are copied.

-s *source_device*
Specifies the name of the source device from which the configuration settings are copied.

Example

To copy the configuration settings of the iSCSI target devices from the `iscsi0` device to the `iscsi1` device, run the following command:

```
cpiscsi -l iscsi1 -s iscsi0
```

This command copies all the current configuration settings from the `iscsi0` device to the `iscsi1` device. Thus, the `iscsi1` device can access all of the target devices by using the same TCP/IP addresses and port numbers, as are accessed by the `iscsi0` device. When the `iscsi1` device accesses the specified target devices, it uses the initiator device name that is specified by the **`initiator_name`** attribute of the `iscsi1` device.

cplv Command

Purpose

Copies the contents of a logical volume to a new logical volume.

Syntax

To Copy to a New Logical Volume

cplv [**-v** *VolumeGroup*] [**-y** *NewLogicalVolume* | -Y *Prefix*] *SourceLogicalVolume*

To Copy to an Existing Logical Volume

cplv **-e** *DestinationLogicalVolume* [**-f**] *SourceLogicalVolume*

Description

Attention: Do not copy from a larger logical volume containing data to a smaller one. Doing so results in a corrupted file system because some data (including the superblock) is not copied. This command will fail if the **cplv** creates a new logical volume and the volume group is varied on in concurrent mode.

The **cplv** command **copies** the contents of *SourceLogicalVolume* to a new or existing *DestinationLogicalVolume*. The *SourceLogicalVolume* parameter can be a logical volume name or a logical volume ID. The **cplv** command creates a new logical volume with a system-generated name by using the default syntax. The system-generated name is displayed.

Note:

1. If you are copying a striped logical volume and the destination logical volume does not exist, an identical copy, including the striped block size and striping width of the source logical volume is created and then the data is copied.
2. If you are copying a striped logical volume and you have created the destination logical volume, with the **mklv** command using a different stripe block size and striping width, or the destination is not a striped logical volume, the new characteristics are maintained, and the data is copied from the source logical volume.
3. To use this command, you must either have root user authority or be a member of the **system** group.
4. The **cplv** command is not allowed on a snapshot volume group.
5. If the *SourceLogicalVolume* is a jfs or jfs2 type, the file system must be successfully unmounted and **fsck** must be run successfully on the newly created file system before the **cplv** command can be run. If you run the **fsck** command before mounting the new file system, errors are returned because the log device contained in the superblock would still refer to the original file system. Mount the file system before running **fsck** so that a new log device is created.

6. If you copy an encrypted logical volume and the destination logical volume does not exist, a logical volume is created with data encryption option disabled. The source encrypted logical volume must be unlocked to copy the encrypted logical volume.

You could also use the System Management Interface Tool (SMIT) **smit cplv** fast path to run this command.

Flags

Item	Description
-e	Specifies that the *DestinationLogicalVolume* exists and that a new logical volume should not be created. If the *DestinationLogicalVolume* is smaller than the *SourceLogicalVolume*, the extra logical partitions are not copied. When you use this flag, any data already in the *DestinationLogicalVolume* is destroyed. For this reason, user confirmation is required, unless the **-f** flag is added. The *Type* characteristic of the *DestinationLogicalVolume* must be **copy** to prevent inadvertently overwriting data. To change the *Type* characteristic, use the **chlv** command.
-f	Copies to an existing logical volume without requesting user confirmation.
-v *VolumeGroup*	Specifies the volume group where the new logical volume resides. If this is not specified, the new logical volume resides in the same volume group as the *SourceLogicalVolume*.
-y *NewLogicalVolume*	Specifies the name to use, in place of a system-generated name, for the new logical volume. Logical volume names must be unique systemwide names, and can range from 1 to 15 characters.
-Y *Prefix*	Specifies a prefix to use in building a system-generated name for the new logical volume. The prefix must be less than or equal to 13 characters. A name cannot begin with a prefix already defined in the PdDv class in the Device Configuration Database for other devices, or a name already used by another device.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To copy the contents of logical volume `fslv03` to a new logical volume, type:

   ```
   cplv fslv03
   ```

   The new logical volume is created, placed in the same volume group as `fslv03`, and named by the system.

2. To copy the contents of logical volume `fslv03` to a new logical volume in volume group `vg02`, type:

   `cplv -v vg02 fslv03` where fslv03 is source logical volume name. It is mandatory field.

   The new logical volume is created, named, and added to volume group `vg02`.

3. To copy the contents of logical volume `lv02` to a smaller, existing logical volume, `lvtest`, without requiring user confirmation, type:

```
cplv  -e lvtest  -f lv02
```

Files

Item	Description
/usr/sbin	Directory where the **cplv** command resides.

cpp Command

Purpose

Performs file inclusion and macro substitution on C language source files.

Syntax

/usr/ccs/lib/cpp [**-C**] [**-P**] [**-q***DBCS*] [**-I***Directory*] [**-U***Name*] [**-D***Name* [=*Defin ition*]] [**-qlanglvl=***Language*] [*InFile*] [*OutFile*]

Description

The **cpp** command performs file inclusion and macro substitution on C language source files. It reads *InFile* and writes to *OutFile* (standard input and standard output by default).

The **cpp** command is designed to conform to the preprocessing directives and instructions for the C language as defined by the document "Draft American National Standard for Information Systems - Programming Language C" (X3J11/88-159).

The **cpp** program recognizes the following special names:

Item	Description
__LINE__	The current line number.
__DATE__	The date of translation of the source file.
__TIME__	The time of translation of the source file.
__STDC__	Indicates a conforming implementation.
__FILE__	The current file name.
__STR__	Indicates the compiler will generate inline code for certain string functions (as defined in **/usr/include/string.h**).
__MATH__	Indicates the compiler will generate inline code for certain math functions (as defined in **/usr/include/math.h**).
__ANSI__	Indicates **langlvl** is set equal to ANSI.
__SAA__	Indicates **langlvl** is set equal to SAA.
__SAA_L2__	Indicates **langlvl** is set equal to SAAL2.
__EXTENDED__	Indicates **langlvl** is set equal to extended.
__TIMESTAMP__	Indicates the date and time when the source file was last modified.

All **cpp** directive lines must begin with a **#** (pound sign). These directives are:

Item	Description
#define *Name TokenString*	Replaces subsequent instances of *Name* with *TokenString*.

Item	Description
#define *Name***(***Argument,...,Argument***)** *TokenString*	Replaces subsequent instances of the sequence *Name* (*Argument*, . . . ,*Argument*) with *TokenString*, where each occurrence of an *Argument* in *TokenString* is replaced by the corresponding token in the comma-separated list. Note that there must not be any space between *Name* and the left parenthesis.
#undef *Name*	Ignores the definition of *Name* from this point on.
#include "*File***"** or **#include <***File>*	Includes at this point the contents of *File*, which **cpp** then processes. If you enclose *File* in " " (double quotation marks) the **cpp** command searches first in the directory of *InFile*, second in directories named with the **-I** flag, and last in directories on a standard list. If you use the <*File*> notation, the **cpp** command searches for *File* only in the standard directories. It does not search the directory in which *InFile* resides.
#line *Number* ["*File*"]	Causes the implementation to behave as if the following sequence of source lines begins with a source line that has a line number as specified by *Number*. If *File* is supplied, the presumed name of the file is changed to be *File*.
#error *TokenString*	Produces a diagnostic message that includes *TokenString*.
#pragma *TokenString*	An implementation-defined instruction to the compiler.
#endif	Ends a section of lines begun by a test directive (**#if**, **#ifdef**, or **#ifndef**). Each test directive must have a matching **#endif**.
#ifdef *Name*	Places the subsequent lines in the output only if: *Name* has been defined by a previous **#define** OR *Name* has been defined by the **-D** flag, OR *Name* is a special name recognized by the **cpp** command, AND *Name* has not been undefined by an intervening **#undef**, OR *Name* has not been undefined with the **-U** flag.

Item	Description
#ifndef *Name*	Places the subsequent lines in the output only if: *Name* has never been defined by a previous **#define**, AND *Name* is not a special name recognized by the **cpp** command, OR *Name* has been defined by a previous **#define** but it has been undefined by an intervening **#undef**, OR *Name* is a special name recognized by the **cpp** command, but it has been undefined with the **-U** flag.
#if *Expression*	Places subsequent lines in the output only if *Expression* evaluates to nonzero. All the binary nonassignment C operators, the ?: operator, and the unary -, !, and - operators are legal in *Expression*. The precedence of the operators is the same as that defined in the C Language. There is also a unary operator **defined**, which can be used in *Expression* in these two forms: **defined (*Name*) or defined *Name*** This allows the utility of **#ifdef** and **#ifndef** in a **#if** directive. Only these operators, integer constants, and names that are known by **cpp** should be used in *Expression*. The **sizeof** operator is not available.
#elif *Expression*	Places subsequent lines in the output only if the expression in the preceding **#if** or **#elif** directive evaluates to false or is undefined, and this *Expression* evaluates to true.
#else	Places subsequent lines in the output only if the expression in the preceding **#if** or **#elif** directive evaluates to false or is undefined (and hence the lines following the **#if** and preceding the **#else** have been ignored). Each test directive's condition is checked in order. If it evaluates to false (0), the group that it controls is skipped. Directives are processed only through the name that determines the directive in order to keep track of the level of nested conditionals; the rest of the directives' preprocessing tokens are ignored, as are the other preprocessing tokens in the group. Only the first group whose control condition evaluates to true (nonzero) is processed. If none of the conditions evaluates to true, and there is a **#else** directive, the group controlled by the **#else** is processed; lacking a **#else** directive, all the groups until the **#endif** are skipped.

Flags

Item	Description
-C	Copies C language comments from the source file to the output file. If you omit this flag, the **cpp** command removes all C language comments except those found on a **cpp** directive line.
-D*Name*[**=***Definition*]	Defines *Name* as in a **#define** directive. The default *Definition* is **1**.
-I*Directory*	Looks first in *Directory*, then looks in the directories on the standard list for **#include** files with names that do not begin with a / (slash). See the previous discussion of **#include**.
-P	Preprocesses input without producing line control information for the next pass of the C compiler.
-q*DBCS*	Specifies double-byte character set mode.
-U*Name*	Removes any initial definition of *Name*, where *Name* is a symbol predefined by the preprocessor (except for the four preprocessor mode indicators: **__ANSI__**, **__EXTENDED__**, **__SAA__**, and **__SAA_L2__**). This flag is not recognized in ANSI mode.
-qlanglvl=*Language*	Selects a language level for processing. *Language* can be ANSI, SAA, SAAL2, or extended. The default is extended. **Note:** When *Language* is extended, _NO_PROTO is not automatically defined. Such definition can be done using the **-D** option in the **/etc/xlc.cfg** file.

Examples

1. To display the text that the preprocessor sends to the C compiler, enter:

   ```
   /usr/ccs/lib/cpp pgm.c
   ```

   This preprocesses `pgm.c` and displays the resulting text at the workstation. You may want to see the preprocessor output when looking for errors in your macro definitions.
2. To create a file containing more readable preprocessed text, enter:

   ```
   /usr/ccs/lib/cpp -P -C pgm.c pgm.i
   ```

   This preprocesses `pgm.c` and stores the result in `pgm.i`. It omits line numbering information intended for the C compiler (`-P`), and includes program comments (`-C`).
3. To predefine macro identifiers, enter:

   ```
   /usr/ccs/lib/cpp -DBUFFERSIZE=512 -DDEBUG
   pgm.c
   pgm.i
   ```

   This defines `BUFFERSIZE` with the value `512` and `DEBUG` with the value `1` before preprocessing.
4. To use **#include** files located in nonstandard directories, enter:

   ```
   /usr/ccs/lib/cpp -I/home/jim/include
   pgm.c
   ```

   This looks in the current directory for quoted **#include** files, then in `/home/jim/include`, and then in the standard directories. It looks in `/home/jim/include` for angle-bracketed **#include** files (< >) and then in the standard directories.
5. To preprocess with the ANSI definition, enter:

```
/usr/ccs/lib/cpp -qlanglvl=ansi pgm.c
```

Files

Item	Description
/usr/include	Standard directory for **#include** files.

cpuextintr_ctl Command

Purpose

Performs CPU external interrupt control related operations on CPUs.

Syntax

cpuextintr_ctl [**-R** *rsetname* | **-C** *CPUList*] **-i** [*enable* | *disable*]

cpuextintr_ctl -q [*enable* | *disable*]

cpuextintr_ctl -Q

Description

This command provides means of enabling, disabling, and querying the external interrupt state on the CPU described by the CPU resource set. Enabling or disabling a CPUs external interrupt could affect the external interrupt delivery to the CPU. Normally, on multiple CPU system, external interrupts can be delivered to any running CPU, and the distribution of interrupts among the CPU is determined by a predefined method. Any external interrupt can only be delivered to a CPU if its interrupt priority is more favored than the current external interrupt priority of the CPU. When external interrupts are disabled via this interface, any external interrupt priority less favoured than **INTMAX** will be blocked until interrupts are enabled again. This command is applicable only on selective hardware types.

> **Note:** Since this command change the way that interrupts is delivered, system performance may be affected. This service guarantees at least one online CPU will have external interrupts enabled for all device interrupts. Any **DLPAR** CPU removal can fail if the operation breaks this guarantee. On an I/O bound system, one CPU may not be enough to handle all of the external interrupts received by the partition. Performance may suffer when there are not enough CPUs enabled to handle external interrupts.

Flags

Item	Description
-R *rsetname*	The CPU resource set that is the target for minimal allowed external interrupt priority related operations.
-C *CPUlist*	List of CPUs to be in the **rset** for minimal allowed external interrupt priority related operations.
-i *enable/disable*	This operation will enable or disable external interrupts on the CPUs specified by either **rsetname** or **CPUlist**.
-q *enable/disable*	This operation will return a list of CPUs that have its external interrupt enabled or disabled.
-Q	This operation will query the external interrupt control state for all the online CPU's.

Note: The CPU id used by this command is logic CPU id.

Security

The user must have root authority with **CAP_NUMA_ATTACH** capability or **PV_KER_CONF** privilege in the RBAC environment.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To disable external interrupts on CPU 0, 4, 8, 12-40:

   ```
   cpuextintr_ctl –C 0 4 8 12-40 –i disable
   ```

2. To enable all the external interrupts on the cpu rset named test/mycpuset:

   ```
   cpuextintr_ctl –R test/mycpuset –i enable
   ```

3. To query CPU external interrupt control status on the system:

   ```
   cpuextintr_ctl -Q
   ```

   The CPUs that have external interrupts enabled:

   ```
     0    1    2    3    4    5    6    7    8    9
    10   11   12   13   14   15   16   17   18   19
    20   21   22   23   24   25   26   27   28   29
    30   31   32   33   34   35   36   37   38   39
    40   41   42   43   44   45   46   47   48   49
    50   51   52   53   54   55   56   57   58   59
    60   61   62   63   64   65   66   67   68   69
   ```

   The CPUs that have external interrupts disabled:

   ```
   70   71   72   73   74   75   76   77   78   79
   ```

4. To query CPUs that have external interrupts disabled on the system:

   ```
   cpuextintr_ctl -q enable
   ```

   The CPUs that have external interrupts enabled:

   ```
   50   51   52
   ```

5. To disable external interrupts on all online CPUs

   ```
   cpuextintr_ctl -R sys/sys0 -i disable
   ```

   The -i option failed on some of the CPUs.

This command will try to disable external interrupts on all online CPUs at the time of operation. Since there is a minimal external interrupt enabled CPU requirement, this operation will be failed on one of the CPUs. The CPU left with external interrupts enabled will be based on the system choice.

Files

Item	Description
/usr/sbin/cpuextintr_ctl	Contains the **cpuextintr_ctl** command.

cpupstat Command

Purpose

Detects configurations that could cause a CPU DR operation to fail.

Syntax

cpupstat [**-v**] **-i** *identifier*

Description

The purpose of this command is to detect configurations that could cause a CPU DR operation to fail. There are multiple steps to the command.

1. Parse and validate the input.
2. Check all the WLM class control block rsets for rsets with a single active CPU matching the passed in CPU. Class control block rsets are located in ccb[cid]->cl_rset, to iterate through all of them the value of CID must be incremented and class validity checked for each possible value. A count of the number of classes with such an rset will printed. If the verbose option is given, the names of the classes will be printed as well.
3. Check all the kernel registry rsets for rsets with a single active cpu matching the passed in CPU. A count of the number of processes with attachments to such rsets will be printed to the user. If the verbose option is given, the process IDs will be printed as well.
4. A count of **bindprocessor** attachments for the highest numbered bind ID will be printed for the user. If the verbose option is given, the process IDs will be printed as well.

Flags

Item	Description
-i	The index of the logical CPU ID.
-v	Verbose option.

Exit Status

If an error is encountered in the execution a suitable error message is written to stderr, and the command exits with a non-zero exit status.

Examples

1.
```
# cpupstat -i 2

3 WLM classes have single CPU rsets with CPU ID 2.
0 processes have single CPU rset attachments with CPU ID 2.
0 processes are bound to bind ID 2.
```

2.
```
# cpupstat -v -i 2

3 WLM classes have single CPU rsets with CPU ID 2.
        c1
        c1.Default
        c1.Shared
0 processes have single CPU rset attachments with CPU ID 2.
0 processes are bound to bind ID 2.
```

3.
```
# cpupstat -i 2

0 WLM classes have single CPU rsets with CPU ID 2.
2 processes have single CPU rset attachments with CPU ID 2.
0 processes are bound to bind ID 2.
```

4.
```
# cpupstat -v -i 2
```

```
0 WLM classes have single CPU rsets with CPU ID 2.
2 processes have single CPU rset attachments with CPU ID 2.
        16600
        26444
0 processes are bound to bind ID 2.
```

For bound processes, the last list, the output is the same as for rset attachments, where the PID gets printed if the **-v** option is specified.

Location

/usr/bin/cpupstat

craps Command

Purpose

Starts the craps game.

Syntax

craps

Description

The **craps** command starts the craps game similar to ones played in Las Vegas. The **craps** command simulates the roller while you place bets. You can bet with the roller by making a positive bet or you can bet with the house by making a negative bet.

You begin the game with a two thousand dollar bankroll. When the program prompts with `bet?`, you can bet all or part of your bankroll. You can not bet more than your current bankroll. The roller throws the dice. The payoff odds are one-to-one.

On the first roll, 7 or 11 wins for the roller; 2, 3, or 12 wins for the house; and any other number becomes the point and you roll again. On subsequent rolls, the point wins for the roller; 7 wins for the house; and any other number rolls again. For example:

```
Your bankroll is $2000
bet? 100
5       3
The point is 8
        6       6
        4       1
        2       1
        2       5
You lose your bet of $100
Your bankroll is $1900
```

In this example, the player has a bankroll of two thousand dollars and bets one hundred dollars. The first roll was 8. This became the point because neither you nor the house wins on a first roll of 8. Subsequent rolls were: 12, 5, 3, and 7. The house wins on a roll of 7 when the roller is trying to match the point. The player lost the bet of one hundred dollars. After displaying the new bankroll, the game will prompt `bet?` and the game will continue.

If you lose your bankroll, the game prompts with `marker?`, offering to lend you an additional two thousand dollars. Accept the loan by responding Y (yes). Any other response ends the game.

When you hold markers, the house reminds you before a bet how many markers are outstanding. When you have markers and your bankroll exceeds two thousand dollars, the game asks `Repay marker?`. If you want to repay part or all of your loan, enter Y (yes). If you have more than one marker, the **craps** command prompts `How many?` If you respond with a number greater than the number of markers you hold, it repeats the prompt until you enter a valid number. If you accumulate 10 markers (a total loan of twenty thousand dollars), the game tells you so and exits. If you accumulate a bankroll of more than fifty thousand dollars while holding markers, the money owed is repaid automatically.

A bankroll of more than one hundred thousand dollars breaks the bank, and the game prompts `New game?` To quit the game, press the Interrupt (Ctrl-C) or End Of File (Ctrl-D) key sequence; the game indicates whether you won, lost, or broke even, and exits.

Files

Item	Description
/usr/games	Location of the system's games.

createvsd Command

Purpose

`createvsd` – Creates a set of virtual shared disks, with their associated logical volumes.

Syntax

`createvsd`
`-n` {*node_list* | `ALL`} `-s` *size_in_MB* `-g` *vg_name*

[{`-c` *vsds_per_node* | `-L`}] [`-A`]

[{`-m` *mirror_count* | `-p` *lvm_strip_size_in_K*}] [`-v` *vsd_name_prefix*]

[`-l` *lv_name_prefix*] [`-T` *lp_size_in_MB*] [`-k` *vsd_type*] [`-x`]

Description

Use this command to create a volume group with the specified name (if one does not already exist) and to create a logical volume within that volume group. You specify the logical volume size using the **-s** flag.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_data
```

and select the **Create a virtual shared disk** option.

Flags

Note: Some examples shown in this list do not contain enough flags to be executable. They are shown in an incomplete form to illustrate specific flags.

-n *node_list*
Specifies the node numbers of the peer domain on which you are creating virtual shared disks. The backup node cannot be the same as the primary node. For nonconcurrent virtual shared disks, the format of the node list is:

```
[P/S] : disk_list1+disk_list2/
```

For concurrent virtual shared disks, the format of the node list is:

```
[S1/S2/...Sn] : disk_list1+disk_list2/
```

"P" specifies the primary server node for serially accessed shared disks, "S" specifies the backup (secondary) server node for serially accessed shared disks, and S1 and S2 specifies the server nodes for concurrently accessed shared disks. *disk_list1* is the list of local physical disks, or vpaths, for the

logical volume on the primary. In other words, this list can be made up of *hdiskx, hdisky,...* or *vpathx, vpathy,....*

Note:

1. Vpaths are available only if the "Subsystem Device Driver" is installed. Vpaths provide "virtual paths" to the same physical volume.
2. Hdisks and vpaths cannot both be specified in the same list.

disk_list1+disk_list2 is the list of local physical disks or vpaths in the volume group on the primary, if you want to have more disks in the volume group than are needed for the logical volume. The sequence in which nodes are listed determines the names given to the virtual shared disks. For example:

```
createvsd -n 1,6,4 -v PRE
```

(with the *vsd_prefix* PRE) creates virtual shared disks PRE1n1 on node 1, PRE2n6 on node 6, and PRE3n4 on node 4.

To create a volume group that spans hdisk2, hdisk3, and hdisk4 on node 1, with a backup on node 3, enter:

```
createvsd -n 1/3:hdisk2,hdisk3,hdisk4/ -v DATA
```

This command creates:

- Virtual shared disk DATA1n1 with logical volume lvDATA1n1 on a volume group with the global volume group name DATA1n1b3 on node 1, exported to node 3. The Logical Volume Manager (LVM) volume group name is DATA. The logical volumes span hdisk2, hdisk3, and hdisk4.

To create volume groups just like that one on nodes 1, 2, and 3 of a system with backup on nodes 4, 5, and 6 of the same system, enter:

```
createvsd -n 1/4:hdisk1,hdisk2,hdisk3/,2/5:hdisk5,hdisk6, \
          hdisk7/,3/6:hdisk2,hdisk4,hdisk6/ -v DATA
```

This command is shown on two lines here, but you must enter it without any spaces between the items in *node_list*.

The command creates:

- Virtual shared disk DATA1n1 with logical volume lvDATA1n1 on a volume group with the local volume group name DATA on node 1, exported to node 4. The global volume group name is DATAn1b4.
- Virtual shared disk DATA2n2 with logical volume lvDATA2n2 on a volume group with the local volume group name DATA on node 2, exported to node 5. The global volume group name is DATAn2b5.
- Virtual shared disk DATA3n3 with logical volume lvDATA3n3 on a volume group with the local volume group name DATA on node 3, exported to node 6. The global volume group name is DATAn3b6.

To create a virtual shared disk where the logical volume spans only two of the physical disks in the volume group, enter:

```
createvsd -n 1/3:hdisk1,hdisk2+hdisk3/ -v DATA
```

This command creates the virtual shared disk DATA1n1 with logical volume lvDATA1n1 spanning hdisk1 and hdisk2 in the volume group DATA, which includes hdisk1, hdisk2, and hdisk3. It exports the volume group DATA to node 3.

If a volume group is already created and the combined physical hdisk lists contain disks that are not needed for the logical volume, those hdisks are added to the volume group. If the volume group has not already been created, `createvsd` creates a volume group that spans *hdisk_list1+hdisk_list2*.

Backup nodes cannot use the same physical disk as the primary does to serve virtual shared disks.

ALL specifies that you are creating virtual shared disks on all nodes in the RSCT peer domain. No backup nodes are assigned if you use this operand. The virtual shared disks will be created on all the physical disks attached to the nodes in *node_list* (you cannot specify which physical disks to use.)

-s

Specifies the size in megabytes of each virtual shared disk.

-g

Specifies the Logical Volume Manager (LVM) volume group name. This name is concatenated with the node number to produce the global volume group name. For example:

```
createvsd -n 6 -g VSDVG
```

creates a volume group with the local volume group name VSDVG and the global volume group name VSDVG1n6 on node 6. The node number is added to the prefix to avoid name conflicts when a backup node takes over a volume group. If a backup node exists, the global volume group name will be concatenated with the backup node number as well as the primary. For example:

```
createvsd -n 6/3/ -g VSDVG
```

creates a volume group with the local volume group name VSDVG and the global volume group name VSDVGn6b3. The primary node is node 6 and the backup node for this volume group is node 3.

-c

Specifies the number of virtual shared disks to be created on each node. If *number_of_vsds_per_node* is not specified, one virtual shared disk is created for each node specified on `createvsd`. If more than one virtual shared disk is to be created for each node, the names will be allocated alternately. For example:

```
createvsd -n 1,6 -c 2 -v DATA
```

creates virtual shared disks DATA1n1 on node 1, DATA2n6 on node 6, DATA3n1 on node 1, and DATA4n6 on node 6.

-L

Allows you to create one virtual shared disk on each node without using sequential numbers, for locally accessed virtual shared disks.

-A

Specifies that virtual shared disk names will be allocated to each node in turn, for example:

```
createvsd -n 1,6 -c 2 -A DATA
```

creates DATA1n1 and DATA2n1 on node 1, and DATA3n6 and DATA4n6 on node 6.

-m

Specifies the LVM mirroring count. The mirroring count sets the number of physical partitions allocated to each logical partition. The range is from 1 to 3 and the default value is 1.

-p

Specifies the LVM strip size (a strip size multiplied by the number of disks in an array equals the stripe size). If this flag is not specified, the logical volumes are not striped. To use striping, the node on which the virtual shared disks are defined must have more than one physical disk.

-v

Specifies a prefix to be given to the names of the created virtual shared disks. This prefix will be concatenated with the virtual shared disk number, node number, and backup node number, if a backup disk is specified. For example, if the prefix PRE is given to a virtual shared disk created on node 1 and there are already two virtual shared disks with this prefix across the partition, the new virtual shared disk name will be PRE3n1. The name given to the underlying logical volume will be lvPRE3n1, unless the `-l` flag is used. The `createvsd` command continues to sequence virtual shared disk names from the last PRE-prefixed virtual shared disk.

If `-v` is not specified, the prefix `vsd` is used.

Note: The last character of the *vsd_name_prefix* cannot be a digit. Otherwise, the 11th virtual shared disk with the prefix PRE would have the same name as the first virtual shared disk with the prefix PRE1. Nor can the *vsd_name_prefix* contain the character '.' because '.' can be any character in regular expressions.

-l

Overrides the prefix *lvx* that is given by default to a logical volume by the `createvsd` command, where *x* is the virtual shared disk name prefix specified by *vsd_name_prefix* or the default (`vsd`). For example:

```
createvsd -n 1 -v DATA
```

creates one virtual shared disk on node 1 named DATA1n1 with an underlying logical volume lvDATA1n1. If the command

```
createvsd -n 1 -v DATA -l new
```

is used, the virtual shared disk on node 1 is still named DATA1n1, but the underlying logical volume is named lvnew1n1.

It is usually more helpful `not` to specify `-l`, so that your lists of virtual shared disk names and logical volume names are easy to associate with each other and you avoid naming conflicts.

-T

Specifies the size of the physical partition in the Logical Volume Manager (LVM) logical volume group and also the logical partition size (they will be the same) in megabytes. You must select a power of 2 in the range 2 - 256. The default is 4MB.

The Logical Volume Manager limits the number of physical partitions to 1016 per disk. If a disk is greater than 4 gigabytes in size, the physical partition size must be greater than 4MB to keep the number of partitions under the limit.

-k *vsd_type*

Specifies the type of virtual shared disk. The options are:

- VSD: specifies a serial access, or nonconcurrent, shared disk, or
- CVSD: specifies a concurrent access shared disk.

The default is VSD.

-x

Specifies that the steps required to synchronize the virtual shared disks on the primary and secondary nodes should **not** be performed; that is, the sequence:

- `varyoffvg` on the primary node
- `exportvg` on the secondary node
- `importvg` on the secondary node
- `chvg` on the secondary node
- `varyoffvg` on the secondary node
- `varyonvg` on the primary nodes

is not done as part of the `createvsd` processing. This speeds the operation of the command and avoids unnecessary processing in the case where several virtual shared disks are being created on the same primary/secondary nodes. In this case, however, you should either `not` specify **-x** on the last `createvsd` in the sequence or issue the volume group commands listed above explicitly.

Parameters

None.

Security

You must have root authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

-1
Indicates that an error occurred.

Restrictions

1. The backup node cannot be the same as the primary node.
2. The last character of *vsd_name_prefix* cannot be numeric.
3. The *vsd_name_prefix* cannot contain the character '**.**'.

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Standard Output

For the following command:

```
createvsd -n 1/2:hdisk13/ -s 1024 -g testvg -v testvsd -T 16
```

The messages to standard output will be similar to:

```
createvsd: calls Getopts.
createvsd: parsing node_list.
createvsd: creates task tables.
createvsd: calls checkclvm.perl on the nodes c164n11.ppd.pok.ibm.com
createvsd: calls domkvglv.perl.
OK:1:mkvg -f -y testvg -s 16 hdisk13
OK:1:mklv -a c -y lvtestvsd1n1 -e x testvg 64 hdisk13
It took about 8 seconds in mkvglv.
createvsd: calls dovaryoffvg.perl testvg on the primary node c164n11.ppd.pok.ibm.com
OK:1:chvg -a n testvg
OK:1:varyoffvg testvg
createvsd: calls doimportvg.perl testvg on the nodes c164n12.ppd.pok.ibm.com with 000048186b991a6f
importvg : testvg
importvg : OK:2:importvg -y testvg hdisk5
importvg : OK:2:chvg -a n testvg
importvg : timestamp 2 testvg 3e036cb33403c8c8
importvg : OK:2:varyoffvg testvg
importvg : It took about 10 seconds.
It took about 12 seconds in importvg.
createvsd: calls vsdvg.
OK:1:vsdvg -g testvgn1b2 testvg 1 2
It took about 12 seconds in vsdvg.
createvsd: calls dovaryonvg.perl testvg on pri nodes c164n11.ppd.pok.ibm.com
OK:1:varyonvg  testvg
createvsd: calls defvsd.
OK:1:defvsd lvtestvsd1n1 testvgn1b2 testvsd1n1
It took about 5 seconds in defvsd.
```

Examples

To create two 4MB virtual shared disks on each of three primary nodes, one of which has a backup, enter:

```
createvsd -n 3,4,7/8/ -c 2 -s 4 -g vsdvg -v TEMP
```

This command creates the following virtual shared disks:

- TEMP1n3, with logical volume lvTEMP1n3 on a volume group with the global volume group name vsdvgn3 on node 3
- TEMP2n4, with logical volume lvTEMP2n4 on a volume group with the global volume group name vsdvgn4 on node 4
- TEMP3n7, with logical volume lvTEMP3n7 on a volume group with the global volume group name vsdvgn7b8 on node 7, also imported to node 8
- TEMP4n3, with logical volume lvTEMP4n3 on a volume group with the global volume group name vsdvgn3 on node 3
- TEMP5n4, with logical volume lvTEMP5n4 on a volume group with the global volume group name vsdvgn4 on node 4
- TEMP6n7, with logical volume lvTEMP6n7 on a volume group with the global volume group name vsdvgn7b8 on node 7, also imported to node 8

To create three virtual shared disks, where the logical volume created on node 3 spans fewer disks than the volume group does, enter:

```
createvsd -n 3,4/:hdisk1,hdisk2+hdisk3/,7/8/ -s 4 -g datavg -v USER
```

This command creates:

- USER1n3, with logical volume lvUSER1n3 defined on a volume group with the global volume group name datavgn3 on node 3.
- USER2n4, with logical volume lvUSER2n4 defined on a volume group with the global volume group name datavgn4 on node 4. datavgn4 spans hdisk1, hdisk2, and hdisk3. lvUSER2n4 spans hdisk1 and hdisk2.
- USER3n7, with logical volume lvUSER3n7 defined on a volume group with the global volume group name datavgn7b8 on node 7, also imported to node 8.
- If no volume group was defined on nodes 3 and 7 before this `createvsd` command was issued, the volume groups datavgn3 and datavgn7b8 are created with one 4MB partition from a single physical disk.

Location

`/opt/rsct/vsd/bin/createvsd`

create_ova Command

Purpose

Creates an open virtual appliance (OVA) package. An OVA package is an archive file that can be deployed as a virtual machine.

Syntax

create_ova **-o** *OutDir* [**-d** *Disk*] [**-i** *Image*] [**-t** *OStype*] [**-e**] [**-f**] [**-g** *Size*]

Description

The **create_ova** command is used to create a single-volume raw disk image and to export contents of a raw disk image to a compatible OVA package format. The OVA package can be imported into any IBM Power Virtualization Center (PowerVC) environment that contains a supported storage device. You can also import the OVA package into any cloud service that supports the Open Virtualization Format (OVF) packaging standard. The imported OVA package can be deployed as a virtual machine.

The **create_ova** command automatically installs the dependent software such as **pipe viewer**, **rpm**, and **yum** before it generates contents of the OVA package without user intervention. If you cannot install

or configure any OVA package, you must perform the recovery steps and retry the installation until it is successful.

For each user session, the results of the command execution are saved in the `/var/adm/ras/<create_ova.pid$$.log>` log file.

Flags

-d *Disk*
Specifies the target disk (logical device) that is used for restoring the **mksysb** image. The target disk is also used as the medium for copying the raw disk image into an OVA package.

-e
Excludes image restoration to the target disk that is specified with the **-d** flag. Use this option when the target disk already contains the exported content.

-f
Ignores file system space warning messages.

Note: When you export the contents of the **mksysb** image or the contents of a raw disk image, the **create_ova** command stores the entire content of the volume in the raw disk image. The **create_ova** command needs enough space to make copies of the volume in the raw disk image and up to 10% extra space for metadata. For example, if the volume in the raw disk image needs 100 GB space, the **create_ova** command needs an additional 10 GB space (total of 110 GB space). When you use the **-f** flag, ensure that the output directory on the destination server has enough space for the exported content.

-g *Size*
Specifies a disk size in GB for the boot volume. The boot volume is used when you create copies of the volume in the raw disk image while importing an OVA package.

-i *Image*
Specifies the file name (absolute path) of a **mksysb** image or the contents of a raw disk image that is used as an input to package the contents into an OVA package.

Note: If you specify a **mksysb** image with this flag, the **-d** flag must be specified to restore the backup content of the image. However, if you specify the contents of a raw disk image with this flag, the contents are copied directly into the OVA package, thus eliminating the need for restoring the backup content from the image.

-o *OutDir*

Specifies the output directory for the OVA package.

Note: When you export the contents of a raw disk image into an OVA package, the raw disk image must first be copied into a file system on the destination server. Therefore, sufficient space must be available on the server to copy all volumes in the raw disk image to files in the file system. After all volumes in the raw disk image are copied to files in the file system, you can create an uncompressed OVA package from the raw disk image.

After the OVA package is created, the files that are used for creating the OVA package are removed. The resulting OVA package is compressed by using the `gzip` utility. When you compress the OVA package, you might notice that temporary files appear in the output directory. You can ignore these temporary files. After the `*.ova.gz` package is created successfully, the appropriate return code is displayed.

-t *OStype*
Specifies the operating system of the system from which the image is used to create an OVA package. The supported values are `aix`, `rhel`, and `sles`. If the **-t** flag is omitted, the default operating system type is `aix`.

Image Requirements

The virtual machine must satisfy specific requirements when you use an existing raw disk image or an existing root volume group as the source for the OVA package. If you do not prepare the virtual machine before you back up or capture a root volume group, errors might occur when you deploy the image. For example, you might not be able to ping the virtual machine that is created when the image is deployed. Before you create a **mksysb** image, or back up a root volume group into a raw disk image, perform the following steps to prepare the virtual machine that must be backed up or captured:

1. Install the cloud-init program.

   Install the cloud-init program on the virtual machine that you want to back up or capture. The cloud-init program takes the user input and configures the operating system and software on the deployed virtual machine. The cloud-init program is widely used in Open Stack for initializing virtual machines. For more information, see Installing and configuring cloud-init.

2. Prepare the virtual machine that must be backed-up or captured.

   Prepare the virtual machine for backup or capture by performing tasks such as cleaning up log files or enabling Resource Monitoring and Control (RMC) connection on the virtual machine. Several PowerVC features such as live migration and dynamic LPAR, require an active RMC connection between the Hardware Management Console (HMC) or the NovaLink partition, and the virtual machine.

   - The virtual machine must use the Virtual I/O Server and virtual storage.
   - If you are copying a disk image, you must shut down the virtual machine.

   For more information, see Set up an RMC connection.

3. Ensure that the following prerequisites are met for the Linux operating system:

   - Operating systems such as SUSE Linux Enterprise Server (SLES) 10, SLES 11, Red Hat® Enterprise Linux (RHEL) 5, and RHEL 6 that use Linux Loader (LILO) or Yaboot boot loaders have special considerations when virtual machines have multiple disks. For more information about configuring a boot loader, see the documentation for the respective Linux operating system.
   - On SLES 12, if you are deploying the image with a network configuration that has a static IP address as the primary adapter, and a dynamic host configuration protocol (DHCP) as secondary adapters, you must modify the **DHCLIENT_SET_DEFAULT_ROUTE** attribute on the virtual machine. Otherwise, the DHCP adapters might override the default gateway of the primary (static) adapter. In the `/etc/sysconfig/network/dhcp` file, set the value of the **DHCLIENT_SET_DEFAULT_ROUTE** attribute to `yes`.
   - If you want the Linux virtual machine to have Multiple Path I/O (MPIO), you must configure the Linux operating system for MPIO on the root device before you back up or capture the virtual machine.

Examples

- To restore a **mksysb** image on the target disk `/dev/hdisk2` and to package its contents into an OVA package that is stored in the `/images` directory, enter the following command:

```
create_ova -o /images -d hdisk2 -i /tmp/backup.sysb
```

- To copy a raw disk image and to package its contents into an OVA package that is stored in the `/images` directory by ignoring the file system space requirements, enter the following command:

```
create_ova -o /images -i /tmp/mysystem.img -f
```

- To export a SLES root volume group of 30 GB and to package its contents into an OVA package that is stored in the `/images` directory in addition to specifying a new disk size requirement of 80 GB, enter the following command:

```
create_ova -o /images -e -d rhdisk2_lv -g 80 -t sles
```

- To export an alternate AIX root volume group that is on the target disk `hdisk2` and to package its contents into an OVA package that is stored in the `/images` directory, enter the following command:

```
create_ova -o /images -e -d hdisk2 -t aix
```

- To export an AIX `rootvg` volume group to the target disk `/dev/hdisk1`, to install the cloud-init program, and to package contents of the volume group into an OVA package that is stored in the `/images` directory, enter the following command:

```
create_ova -o /images -d hdisk1
```

crfs Command

Purpose

Adds a file system.

Syntax

crfs **-v** *VfsType* { **-g** *VolumeGroup* | **-d** *Device* } [**-l** *LogPartitions*] **-m** *MountPoint* [**-n** *NodeName*] [**-u** *MountGroup*] [**-A** { **yes** | **no** }] [**-p** {**ro** | **rw** }] [**-a** *Attribute*=*Value* ...] [**-t** { **yes** | **no** }]

Description

The **crfs** command creates a file system on a logical volume within a previously created volume group. A new logical volume is created for the file system unless the name of an existing logical volume is specified using the **-d**. An entry for the file system is put into the **/etc/filesystems** file.

The **crfs** command ignores any *Attribute=Value* pair that the command does not understand but adds them to an appropriate stanza in the **/etc/filesystems** file.

`Example:`

```
crfs -a abcd=1G /
```

This sets the new **abcd** attribute to the value of **1G** in the root stanza in the **/etc/filesystems** file.

Note:

1. The file system is created with the **setgid** (set group ID) bit enabled. This determines the default group permissions. All directories created under the new file system will have the same default group permissions. If the command was run over an existing logical volume for a jfs2 file system the setgid bit is never set.
2. For information about creating a filesystem on a striped logical volume, refer to **File Systems on Striped Logical Volumes** in the **mklv** documentation.

You can also use the System Management Interface Tool (SMIT) **smit crfs** fast path to run this command.

Flags

Item	Description
`-a` *Attribute*=*Value*	Specifies a virtual file system-dependent attribute/value pair. To specify more than one attribute/value pair, provide multiple **-a** *Attribute=Value* parameters (see an example). The following attribute/value pairs are specific to the Journaled File System (JFS):

Item	Description

-a ag={ 8 | 16 | 32 | 64 }
Specifies the allocation group size in megabytes. An allocation group is a grouping of i-nodes and disk blocks similar to BSD cylinder groups. The default **ag** value is 8.

-a bf={ true | false }
Specifies a large file enabled file system. See "Understanding Large File Enabled File Systems" for more information. If you do not need a large file enabled file system, set this option to false; this is the default. Specifying **bf=true** requires a fragment size of 4096 and **compress=no**.

-a compress={ no | LZ }
Specifies data compression. If you do not want data to be compressed, set this option to **no**. The default compress value is **no**. Selecting compression requires a fragment size of 2048 or less.

-a frag={ 512 | 1024 | 2048 | 4096 }
Specifies the JFS fragment size in bytes. A file system fragment is the smallest unit of disk storage that can be allocated to a file. The default fragment size is 4096 bytes.

-a logname=*LVName*
Specifies the log logical volume name. The specified logical volume will be the logging device for the new JFS. The *LVName* logical volume must already exist. The default action is to use an existing logging device in the target volume group.

-a nbpi={ 512 | 1024 | 2048 | 4096 | 8192 | 16384 | 32768 | 65536 | 131072 }
Specifies the number of bytes per i-node (nbpi). The nbpi affects the total number of i-nodes on the file system. The **nbpi** value is inversely proportional to the number of i-nodes on the file system. The default **nbpi** value is 4096 bytes.

-a size=*Value*
Specifies the size of the Journaled File System. Size can be specified in units of 512-byte blocks, Megabytes or Gigabytes. If Value has the M suffix, it is interpreted to be in Megabytes. If Value has a G suffix, it is interpreted to be in Gigabytes. If the specified size is not evenly divisible by the physical partition size, it is rounded up to the closest number that is evenly divisible. This attribute is required when creating a JFS file system. See "Understanding JFS Size Limitations" for more information.

The maximum size of a JFS file system is a function of its fragment size and the NBPI value. These values yield the following size restrictions:

```
   NBPI  Minimum AG Size    Fragment Size        Maximum Size (GB)
    512         8        512, 1024, 2048, 4096          8
   1024         8        512, 1024, 2048, 4096         16
   2048         8        512, 1024, 2048, 4096         32
   4096         8        512, 1024, 2048, 4096         64
   8192         8        512, 1024, 2048, 4096        128
  16384         8             1024, 2048, 4096        256
  32768        16                   2048, 4096        512
  65536        32                         4096       1024
 131072        64                         4096       1024
```

You can have NBPI values from 512 to 128K, with corresponding maximum file system sizes.

Item	Description
	The volume group in which the file system resides defines a maximum logical volume size and also limits the file system size. **Note:** 1. The **ag**, **bf**, **compress**, **frag**, and **nbpi** attributes are set at file system creation and cannot be changed after the file system is successfully created. The **size** attribute defines the minimum file system size, and you cannot decrease it once the file system is created. 2. The root filesystem (**/**) cannot be compressed. 3. Some **nbpi** values and allocation group sizes are mutually exclusive. See "Understanding JFS Size Limitations" for information.
The following attribute/value pairs are specific to the Enhanced Journaled File System (JFS2):	
-a *Attribute=Value*	**-a agblksize={ 512 | 1024 | 2048 | 4096 }** Specifies the JFS2 block size in bytes. A file system block is the smallest unit of disk storage that can be allocated to a file. The default block size is 4096 bytes. **-a ea={v1 | v2}** Specifies the format to be used to store named extended attributes in the JFS2 file system. The v2 format provides support for scalable named extended attributes as well as support for NFS4 ACLs. The **v1** format is compatible with prior versions of AIX. The default format is **v1**. **-a efs={yes | no}** Specifies whether the file system is an Encrypted File System (EFS). **yes** The **crfs** command creates a file system that is EFS-enabled. When the file system is EFS-enabled, you do not need to specify the **ea** attribute because the **crfs** command automatically stores scalable extended attributes of the **v2** format. **no** The **crfs** command creates a file system that is not EFS-enabled. **Note:** The **crfs** commands prevents EFS from enabling the following file systems (mount points) because the security infrastructures (kernel extensions, libraries and so on) are not available during boot: • **/** • **/usr** • **/var** • **/opt** **-a isnapshot={yes|no}** Specifies whether the file system supports internal snapshots. A file system created to support internal snapshots also uses extended attributes of the **v2** format.

Item	Description

-a logname=*LVName*
Specifies the log logical volume name. The specified logical volume is the logging device for the new JFS2. The *LVName* logical volume must already exist. The default action is to use an existing logging device in the target volume group. Keyword **INLINE** can be used to place the log in the logical volume with the JFS2 file system. The **INLINE** log defaults to .4% of the logical volume size if **logsize** is not specified.

-a logsize=*Value*
Specifies the size for an INLINE log in MBytes. The input size must be a positive value. If the inline log size is greater than or equal to 1, the input size must be an integer. If the input is floating point value of less than 1 and greater than or equal to 0, the input size is ignored and the default inline log size is taken.

The input is ignored if the INLINE log not being used. It cannot be greater than 10% of the size of the file system and it cannot be greater than 2047 MBytes.

-a maxext=*Value*
Specifies the maximum size of a file extent in file system blocks. A zero value implies that the JFS2 default maximum should be used. Values less than 0 or exceeding maximum supported extent size of 16777208 are invalid. Note that existing file extents are not affected by this change.

-a mountguard={yes | no}
Guards the file system against the unsupported concurrent mounts in a PowerHA or other clustering environment. If the mountguard is enabled, the file system cannot be mounted if it appears to be mounted on another node or system. To temporarily override the mountguard setting, see the **noguard** option of the **mount** command.

-a options=*mountOptions*
Specifies which `mount` option is passed into `crfs` for the file system being created. For a list of the valid options, refer to the **mount** command.

-a quota={userquota | groupquota | userquota,groupquota | no}
Specifies the type of quotas that can be enabled on the file system. You can set the **quota** attribute to one of the following values:

userquota
The space for each user cannot exceed the space quota that is assigned for each user.

groupquota
The space for each group cannot exceed the space quota that is assigned for each group.

userquota,groupquota
Both user quota and group quota are enabled for each user and group.

no
All the quotas are disabled on the file system.

Item	Description
	-a size=*Value*
	Specifies the size of an Enhanced Journaled File System (JFS2). Size can be specified in units of 512-byte blocks, Megabytes or Gigabytes. If *Value* has the M suffix, it is interpreted to be in Megabytes. If *Value* has a G suffix, it is interpreted to be in Gigabytes. If the specified size is not evenly divisible by the physical partition size, it is rounded up to the closest number that is evenly divisible. This attribute is required when creating a JFS2 file system unless the **-d** flag has been specified. If the **-d** flag is specified, the file system is the size of the logical volume. The volume group in which the file system resides defines a maximum logical volume size and limits the file system size. The minimum size for a JFS2 file system is 16 MB. The maximum size is determined by the file system block size:

```
fs block size (byte)        MAX fssize (TB)
========================================
512                         4
1024                        8
2048                        16
4096                        32
```

-a vix={yes|no}

Specifies whether the file system can allocate i-node extents smaller than the default of 16 KB if there are no contiguous 16 KB extents free in the file system. After a file system is enabled for small free extents, the file system cannot be accessed on AIX 5.1 or earlier releases.

yes
The file system can allocate variable-length i-node extents. The **yes** value is the default value beginning with AIX 6.1.

no
The file system must use the default size of 16 KB for i-node extents. The **no** value has no effect if the file system contains variable-length i-node extents.

-A Specifies whether the file system is mounted at each system restart:

yes
File system is automatically mounted at system restart.

no
File system is not mounted at system restart (default value).

Note: The **crfs** command accesses the first letter for the auto mount **-A** option.

-d *Device* Specifies the device name of a device or logical volume on which to make the file system. This is used to create a file system on an already existing logical volume.

-g *VolumeGroup* Specifies an existing volume group on which to make the file system. A volume group is a collection of one or more physical volumes.

-l *LogPartitions* Specifies the size of the log logical volume, expressed as a number of logical partitions. This flag applies only to JFS and JFS2 file systems that do not already have a log device.

-m *MountPoint* Specifies the mount point, which is the directory where the file system will be made available.

Note: If you specify a relative path name, it is converted to an absolute path name before being inserted into the **/etc/filesystems** file.

-n *NodeName* Specifies the remote host name where the file system resides. This flag is only valid with remote virtual file systems such as the Network File System (NFS).

Item	Description
-p	Sets the permissions for the file system. **ro** Read-only permissions **rw** Read-write permissions
-t	Specifies whether the file system is to be processed by the accounting subsystem: **yes** Accounting is enabled on the file system. **no** Accounting is not enabled on the file system (default value).
-u *MountGroup*	Specifies the mount group.
-v *VfsType*	Specifies the virtual file system type.

Note: The **agblksize** attribute is set at file system creation and cannot be changed after the file system is successfully created. The **size** attribute defines the minimum file system size, and you cannot decrease it once the file system is created.

The **ea** attributes format is set at file system creation. The **chfs** command can be used to convert the extended attribute format from **v1** to **v2**, but the format cannot be converted back. The conversion is done in an on-demand manner such that any extended attribute or ACL writes cause the conversion for that file object to occur.

The **maxext** attribute is ignored in older releases even if the file system was created with it on a later release.

Security

Access Control

Only the root user or a member of the **system** group can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To make a JFS on the `rootvg` volume group with nondefault fragment size and nondefault nbpi, enter:

   ```
   crfs  -v jfs  -g  rootvg  -m /test -a \ size=32768 -a frag=512 -a nbpi=1024
   ```

   This command creates the **/test** file system on the `rootvg` volume group with a fragment size of 512 bytes, a number of bytes per i-node (nbpi) ratio of 1024, and an initial size of 16MB (512 * 32768).
2. To make a JFS on the `rootvg` volume group with nondefault fragment size and nondefault nbpi, enter:

   ```
   crfs -v jfs -g rootvg -m /test -a size=16M -a frag=512 -a nbpi=1024
   ```

   This command creates the **/test** file system on the `rootvg` volume group with a fragment size of 512 bytes, a number of bytes per i-node (nbpi) ratio of 1024, and an initial size of 16MB.
3. To create a JFS2 file system which can support NFS4 ACLs, type:

   ```
   crfs -v jfs2 -g rootvg -m /test -a size=1G -a ea=v2
   ```

This command creates the **/test** JFS2 file system on the rootvg volume group with an initial size of 1 gigabyte. The file system will store extended attributes using the **v2** format.

Files

Item	Description
/etc/filesystems	Lists the known file systems and defines their characteristics.

cron Daemon

Purpose

Runs commands automatically.

Syntax

`cron` [`-f` *configurationfile*] [`-Q`]

Description

The **cron** daemon runs shell commands at specified dates and times. The following event types are scheduled by the **cron** daemon:

- **crontab** command events
- **at** command events
- **batch** command events
- **sync** subroutine events
- **ksh** command events
- **csh** command events

The way these events are handled is specified by the **/var/adm/cron/queuedefs** file.

Regularly scheduled commands can be specified according to instructions contained in the **crontab** files. You can submit your **crontab** file with the **crontab** command. Use the **at** command to submit commands that are to be run only once. Because the **cron** daemon never exits, it should be run only once.

The **cron** daemon examines **crontab** files and **at** command files only when the **cron** daemon is initialized. When you make changes to the **crontab** files using the **crontab** command, a message indicating the change is sent to the **cron** daemon. This eliminates the overhead of checking for new or changed files at regularly scheduled intervals.

Note: When a user is no longer available, the **cron** jobs for that user are no longer run. Even if the user eventually becomes available, **cron** events for that user are no longer queued. The **cron** daemon does not log the information about user availability to the `cronlog` file.

When the **TZ** environment variable is changed, either with the **chtz** command through SMIT, the **cron** daemon must be restarted. This enables the **cron** daemon to use the correct time zone and summer time change information for the new **TZ** environment variable.

Note:

1. If you have a job that is scheduled to run between 1:00 a.m. and 2:00 a.m. on the day your time zone changes from daylight saving time to standard time, your job will run twice.
2. If you have a job that is scheduled to run between 2:01 a.m. and 2:59 a.m. on the day your time zone changes from standard time to daylight saving time, your job will not run. You can change the time these jobs run, run them manually, or with until the following day to run them. The **cron** daemon does not need to be stopped. However, if changes are made to the **TZ** environment variable, kill the current **cron** daemon so that it automatically respawns and recognizes the new **TZ** setting.

3. If you have a job that is scheduled to run at 2:00 a.m. on the day your time zone changes from standard time to daylight saving time, your job will run one second early.

The `cron` daemon reads the `/etc/cronlog.conf` configuration file provided by the user to log the information. If a configuration file has not been created, then the `cron` daemon creates a log of its activities in the `/var/adm/cron/log` file. The `cron` daemon reads the configuration file when it is activated and when it receives the hangup signal.

If the `cron` daemon is not able to create or open the user-specified logfile, then it creates a log of its activities in the `/var/adm/cron/log` file.

Flags

Item	Description
`-f` *ConfigurationFile*	Specifies an alternate configuration file.
`-Q`	Quiet mode. If specified, `-Q` disables the `cron` logging. This parameter is valid for a user-configured log file as well as the default `/var/adm/cron/log` file. This option must follow the `-f` option (if `-f` is specified).

Security

Auditing Events

If the auditing subsystem is properly configured and is enabled, the **cron** daemon generates the following audit record (event) every time the command is run:

Event	Information
CRON_Start	Lists the name of each job, whether the job was initiated by an **at** or **cron** command, and the time the job started.
CRON_Finish	Lists the user's name, process ID of the job, and the time the processing was completed.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/var/adm/cron/FIFO	A named pipe that sends messages to the **cron** daemon when new jobs are submitted with the **crontab** or **at** commands.
/var/adm/cron	Specifies the main **cron** daemon directory.
`/var/adm/cron/log`	Default log file which specifies the accounting information for all the executed `cron`. Contains information like the owner, pid, start time, command, and the exit status of the `cron` job. Rotation is not performed on this file.
`/etc/cronlog.conf`	Specifies the default `cron` configuration file for logging information.
/var/adm/cron/queuedefs	Specifies the **cron** daemon events file.
/var/spool/cron	Specifies the spool area.
/usr	Indicates directory kept open by the **cron** daemon.

Item	Description
/usr/bin	Indicates directory kept open by the **cron** daemon.
/usr/lib	Indicates directory kept open by the **cron** daemon.
/etc	Indicates directory kept open by the **cron** daemon.
/tmp	Indicates directory kept open by the **cron** daemon.

Configuration File

The configuration file informs the `cron` daemon where and how to log the information. Using the configuration file you can specify logfile names, size limits, rotation policies, compress and archive attributes.

If you do not use the `-f` flag, the `cron` daemon reads the default `/etc/cronlog.conf` configuration file.

If `cron` fails to open the configuration file, it continues with `/var/adm/cron/log`.

The `cron` daemon ignores blank lines and lines beginning with a # (pound sign).

cronadm Command

Purpose

Lists or removes **crontab** or **at** jobs.

Syntax

To List or Remove crontab Jobs

cronadm cron { { **-l** | **-v** } [*UserName*] ... | **-r** *UserName* }

To List or Remove at Jobs

cronadm at { **-l** [*UserName*] | **-r** { *UserName* | *JobName* } }

Description

The **cronadm** command is used by a root user to list or remove all users **crontab** or **at** jobs.

The **cron** jobs are listed and removed by the *UserName* parameter. One or more *UserNames* can be specified. To list all **cron** jobs, do not specify a user. The **at** jobs are listed by *UserName* and can be removed either by the *UserName* parameter or by the *JobName* parameter.

The name of a **crontab** job file is the name of the user who submitted the **crontab** job and the name of the file in the **/var/spool/cron/crontabs** directory. The name of an **at** job is the name of the user who submitted the **at** job concatenated with a code for the time the **at** job was submitted.

Flags

cronadm cron

Item	Description
-l	Lists all **crontab** files. If the *UserName* parameter is specified, only the designated **crontab** files are listed.
-r	Removes **crontab** files. The *UserName* parameter should be specified, to remove the designated **crontab** file.
-v	Lists the status of all **crontab** jobs. If the *UserName* parameter is specified, only the designated **crontab** files are listed verbosely.

cronadm at

Item	Description
-l	Lists the **at** jobs for the user specified by the *UserName* parameter.
-r	Removes the **at** job specified by either the *UserName* or *JobName* parameter.

Security

Access Control

Used only by a user with root authority.

Auditing Events

If the auditing subsystem is properly configured and is enabled, the **cronadm** command generates the following audit record (event) every time the command is run:

Event	Information
AT_JobRemove	Lists whether a **crontab** or **at** job was removed and when.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list all **crontab** jobs, enter:

   ```
   cronadm cron  -l
   ```

2. To list all **at** jobs currently queued for user bob, enter:

   ```
   cronadm at  -l bob
   ```

Files

Item	Description
/usr/bin/cronadm	Contains the **cronadm** command.

crontab Command

Purpose

Submits, edits, lists, or removes cron jobs.

Syntax

crontab [**-e** [*UserName*] | **-l** [*UserName*] | **-r** [*UserName*] | **-v** [*UserName*] | *File*]

Description

The **crontab** command submits, edits, lists, or removes cron jobs. A cron job is a command run by the **cron** daemon at regularly scheduled intervals. To submit a cron job, specify the **crontab** command with the **-e** flag. The **crontab** command invokes an editing session that allows you to create a **crontab** file. You create entries for each cron job in this file. Each entry must be in a form acceptable to the **cron** daemon. For information on creating entries, see The crontab File Entry Format.

When you finish creating entries and exit the file, the **crontab** command copies it into the **/var/spool/cron/crontabs** directory and places it in a file named for your current user name. If a file with your name already exists in the **crontabs** directory, the **crontab** command overwrites it.

Alternatively, you can create a **crontab** file by specifying the *File* parameter. If the file exists, it must be in the format the **cron** daemon expects. If the file does not exist, the **crontab** command invokes the editor. If the **EDITOR** environment variable exists, the command invokes the editor it specifies. Otherwise, the **crontab** command uses the **vi** editor.

To list the contents of your **crontab** file, specify the **crontab** command with the **-l** flag. To remove an existing file, use the **-r** flag.

The optional *UserName* parameter can be used by the owner of the **crontab** file or by the root user to edit, list, remove, or verify the status of the cron jobs for the specified user. If the *UserName* is invalid, an error message is generated and the program exits.

If the optional *UserName* parameter is not specified, the **crontab** flags are available for the root user and the current user.

Security

Only the root user or the owner of the **crontab** file can use *UserName* following the **-e**, **-l**, **-r**, and **-v** flags to edit, list, remove, or verify the **crontab** file of the specified user.

The cron Daemon

The **cron** daemon runs commands according to the **crontab** file entries. Unless you redirect the output of a cron job to standard output or error, the **cron** daemon mails you any command output or errors. If you specify a cron job incorrectly in your **crontab** file, the **cron** daemon does not run the job.

The **cron** daemon examines **crontab** files only when the **cron** daemon is initialized. When you make changes to your **crontab** file using the **crontab** command, a message indicating the change is sent to the **cron** daemon. This eliminates the overhead of checking for new or changed files at regularly scheduled intervals.

Controls on Using the crontab Command

The **/var/adm/cron/cron.allow** and **/var/adm/cron/cron.deny** files control which users can use the **crontab** command. A root user can create, edit, or delete these files. Entries in these files are user login names with one name to a line. If your login ID is associated with more than one login name, the **crontab** command uses the first login name that is in the **/etc/passwd** file, regardless of which login name you might actually be using. Also, to allow users to start **cron** jobs, the daemon attribute in the **/etc/security/user** file should be set to TRUE , using the **chuser** command.

The following is an example of an **cron.allow** file:

```
root
nick
dee
sarah
```

If the **cron.allow** file exists, only users whose login names appear in it can use the **crontab** command. The root user's log name must appear in the **cron.allow** file if the file exists. A system administrator can explicitly stop a user from using the **crontab** command by listing the user's login name in the **cron.deny** file. If only the **cron.deny** file exists, any user whose name does not appear in the file can use the **crontab** command.

A user cannot use the **crontab** command if one of the following is true:

- The **cron.allow** file and the **cron.deny** file do not exist (allows root user only).
- The **cron.allow** file exists but the user's login name is not listed in it.
- The **cron.deny** file exists and the user's login name is listed in it.

If neither the **cron.allow** nor the **cron.deny** file exists, only someone with root user authority can submit a job with the **crontab** command.

The crontab File Entry Format

A **crontab** file contains entries for each cron job. Entries are separated by newline characters. Each **crontab** file entry contains six fields separated by spaces or tabs in the following form:

```
minute  hour  day_of_month  month  weekday  command
```

These fields accept the following values:

Item	Description
minute	0 through 59
hour	0 through 23
day_of_month	1 through 31
month	1 through 12
weekday	0 through 6 for Sunday through Saturday
command	a shell command

You must specify a value for each field. Except for the *command* field, these fields can contain the following:

- A number in the specified range. To run a command in May, specify 5 in the **month** field.
- Two numbers separated by a dash to indicate an inclusive range. To run a **cron** job on Tuesday through Friday, place 2-5 in the **weekday** field.
- A list of numbers separated by commas. To run a command on the first and last day of January, you would specify 1,31 in the **day_of_month** field.
- A combination of two numbers separated by a dash to indicate an inclusive range and a list of numbers separated by commas can be used in conjunction. To run a command on the first, tenth to sixteenth and last day of January, you would specify 1,10-16,31 in the day_of_month field. The above two points can also be used in combination.
- An ***** (asterisk), meaning all allowed values. To run a job every hour, specify an asterisk in the hour field.

Note: Any character preceded by a backslash (including the %) causes that character to be treated literally. The specification of days may be made by two fields (day of the month and day of the week). If you specify both as a list of elements, both are adhered to. For example, the following entry:

```
0 0 1,15 * 1 command
```

would run command on the first and fifteenth days of each month, as well as every Monday. To specify days by only one field, the other field should contain an ***** .

Specifying Commands

The **cron** daemon runs the command named in the sixth field at the selected date and time. If you include a % (percent sign) in the sixth field, the **cron** daemon treats everything that precedes it as the command invocation and makes all that follows it available to standard input, unless you escape the percent sign (\%). Blank lines and lines whose first non-blank character is the number sign (#) will be ignored. If the arguments to the command have a backslash ('\'), the backslash should be preceded by another backslash.

Note: The shell runs only the first line of the command field. All other lines are made available to the command as standard input.

The **cron** daemon starts a subshell from your **HOME** directory. If you schedule a command to run when you are not logged in and you want commands in your **.profile** file to run, the command must explicitly read your **.profile** file.

The **cron** daemon supplies a default environment for every shell, defining **HOME, LOGNAME, SHELL** (=**/usr/bin/sh**), and **PATH** (=**/usr/bin**).

Auditing Events

If the auditing subsystem is properly configured and is enabled, the **crontab** command generates the following audit record (event) every time the command is run:

Event	Information
CRON_JobRemove	Lists which users removed a **crontab** file and when.
CRON_JobAdd	Lists which users edited a **crontab** file and when.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Flags

Item	Description
-e *UserName*	Edits a copy of the user's **crontab** file or creates an empty file to edit if the **crontab** file does not exist for a valid *UserName*. When editing is complete, the file is copied into the **crontab** directory as the user's **crontab** file.
-l *UserName*	Lists the user's **crontab** file.
-r *UserName*	Removes the user's **crontab** file from the **crontab** directory.
-v *UserName*	Lists the status of the user's cron jobs.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To copy a file called `mycronjobs` into the **/var/spool/cron/crontabs** directory, enter the following:

   ```
   crontab mycronjobs
   ```

   The file will be copied as:

   ```
   /var/spool/cron/crontabs/<username>
   ```

   where <*username*> is your current user name.
2. To write the time to the console every hour on the hour, enter:

   ```
   0 * * * * echo The hour is `date` .
   >/dev/console
   ```

3. To run the **calendar** command at 6:30 a.m. every Monday, Wednesday, and Friday, enter:

   ```
   30 6 * * 1,3,5 /usr/bin/calendar
   ```

4. To run the **calendar** command every day of the year at 6:30, enter the following:

```
30 6 * * * /usr/bin/calendar
```

5. To run a script called `maintenance` every day at midnight in August, enter the following:

```
0 0 * 8 * /u/harry/bin/maintenance
```

6. To define text for the standard input to a command, enter:

```
0 16 * 12 5 /usr/sbin/wall%HAPPY HOLIDAY!%Remember to
turn in your time card.
```

The text following the **%** (percent sign) defines the standard input to the **wall** command as:

```
HAPPY HOLIDAY!

Remember to turn in your time card.
```

Files

Item	Description
/var/adm/cron/FIFO	A named pipe that sends messages to the **cron** daemon when new jobs are submitted with the **crontab** or **at** command.
/var/spool/cron/crontabs	Specifies the crontab spool area.
/**var/adm/cron/cron.allow**	Specifies a list of users allowed access to the **crontab** command.
/var/adm/cron/cron.deny	Specifies a list of users denied access to the **crontab** command.

crvfs Command

Purpose

Creates entries in the **/etc/vfs** file.

Syntax

crvfs *VFSEntry*

Description

The **crvfs** command adds **/etc/vfs** file entries by specifying fields within the *VFSEntry* parameter. The *VFSEntry* parameter is composed of the following fields: *VFSName:VFSNumber:MountHelper:FileSystemHelper*.

All fields in the *VFSEntry* parameter are required, but the reserved word "none" can be specified for the *MountHelper* and *FileSystemHelper* fields if there is no corresponding helper. If all the arguments are satisfactory, and neither the *VFSName* nor the *VFSNumber* given on the command line already exist, a new entry is created in the **/etc/vfs** file.

Parameters

Item	Description
VFSEntry	Specifies a string in the following format: *VFSName*:*VFSNumber*:*MountHelper*:*FileSystemHelper* ***VFSName*** Specifies the name of a virtual file system type. ***VFSNumber*** Specifies the virtual file system type's internal number as known by the kernel. ***MountHelper*** Specifies the name of the backend used to mount a file system of this type. ***FileSystemHelper*** Specifies the name of the backend used by certain file system specific commands to perform operations on a file system of this type.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To create a new vfs entry called `newvfs`, enter:

```
crvfs "newvfs:4:none:/etc/helpers/newvfshelper"
```

This creates the `newvfs` entry.

Files

Item	Description
/etc/vfs	Contains descriptions of virtual file system types.

csh Command

Purpose

Invokes the C shell.

Syntax

csh [**-v** | **-V**] [**-x** | **-X**] [**-e**] [**-f**] [**-i**] [**-n**] [**-c** *String* | **-s** | **-t**] [**-b**] [*File* [*Parameter*]]

Description

The C shell is an interactive command interpreter and a command programming language that uses syntax similar to the C programming language. The shell carries out commands either interactively from a terminal keyboard or from a file. The **csh** command invokes the C shell.

When you invoke the **csh** command, it begins by looking in your home directory and executing commands from the **.cshrc** file (used to store customized user information) if it exists. If the **csh** command runs as a login shell, it executes commands from your **.cshrc** and **.login** files.

After the shell processes flag arguments, if neither the **-i**, **-c**, **-s**, nor **-t** flag is specified and the *File* [*Parameter*] is specified, then the shell executes the script file identified by the *File* [*Parameter*], including any parameters specified. The script file specified must have read permission; the shell ignores any **setuid** and **setgid** settings.

Note: You should not specify a script file if you use the **csh** command with either the **-c** or **-s** flag.

If you specify a script file, the command opens the file and saves the script file name for possible resubstitution by `$0` (dollar sign, zero). The script will then be carried out by **csh**. Remaining parameters initialize the **argv** variable.

Notes:

1. If C shell is already running, the **.cshrc** file can be read again by typing `source Pathname`, where the *Pathname* parameter is the path to the **.cshrc** file.
2. To avoid problems with remote operations, the **.cshrc** file should not contain any functions that echo output unless they test for the **$prompt** variable, which signifies that the shell is interactive. Otherwise, whenever a remote system uses the **exec** command on a command sent by the local system, both the command and the shell are carried out. For example, `exec csh rcp -t Filename` executes the **.cshrc** file and treats the echoed output as the expected response. An **if** clause can be used to check for the **$prompt** variable.

Flags

If the first argument to a shell is a **-** (minus sign), that shell is a login shell. The C shell flags are interpreted as follows:

Item	Description
-b	Forces a break from option processing, causing any further shell arguments to be treated as non-option arguments. This flag can be used to pass options to a shell script without confusion or possible subterfuge. The shell cannot run a script whose real and effective user and group IDs differ without this flag.
-c	Reads commands from the following single argument, which must be present. Any remaining arguments are placed in the **argv** variable.
-e	Exits if any invoked command ends abnormally or yields a nonzero exit status.
-f	Starts the C shell without searching for or running commands from the **.cshrc** file in your home directory.
-i	Prompts for its top-level input (an interactive shell), even if input does not appear to be coming from a workstation. Shells are interactive without this flag if their input and output are attached to workstations.
-n	Parses commands but does not run them. This flag aids you in syntactic checking of shell procedures.
-s	Takes command input from standard input.
-t	Reads and processes a single line of input. You can use a \ (backslash) to escape the new-line character at the end of the current line and continue onto another line.
-V	Sets the **verbose** shell variable before the **.cshrc** file runs.
-v	Sets the **verbose** shell variable, so that command input is echoed after history substitution.
-X	Sets the **echo** shell variable even before the **.cshrc** file runs.
-x	Sets the **echo** shell variable, so that commands are echoed after all substitutions and immediately before they run.

Files

Item	Description
$HOME/.cshrc	Read at the beginning of execution by each shell. The **.cshrc** file is user-defined.
$HOME/.login	Read by the login shell after the **.cshrc** file at login.
$HOME/.logout	Read by the login shell at logoff.
/usr/bin/sh	Contains the path to the default shell.
/tmp/sh*	Contains the temporary file for **<<**.
/etc/passwd	Contains the source of home directories for the **~***File* parameter.

csmstat Command

Purpose

`csmstat` – Provides a snapshot of cluster node reachability, power status, and network interface status.

Syntax

`csmstat` [`-h`]

`csmstat` [`-l`] [`-a`] [`-S`] [`-s` *select_string*] [`-d` *delimiter*] [`-n` *node_list*] [`-N` *nodegroups*]

Description

The `csmstat` command gathers node reachability, power status and network interface status for one or more nodes and displays the output. The default ordering for output is by host name. If there are multiple hardware control points for a node, multiple HMCs for example, then the first hardware control point in the list is shown.

Note: This command does not currently support nodes on IntelliStation workstations.

Flags

-a
Displays attribute information for all nodes. This is the default.

-d
Specifies delimiter-formatted output using the specified delimiter – colons, for example. Use this flagoption to specify a delimiter of one or more characters. This flagoption cannot be used with the –a flagoption.

-h
Displays command usage.

-l
Returns LCD values for SP Nodes, p660 servers, and HMC-attached IBM System p servers. This flagoption cannot be used with the –d flagoption.

-n *node_list*
Specifies a comma or space-separated list of node names to display attribute information. Space-separated node names must be inside double quotes. For information about specifying node ranges, see the `noderange` man page.

-N *nodegroups*
Specifies a comma or space-separated list of node groups to display attribute information. Space-separated node groups must be inside double quotes.

-s
Specifies, by column headers, which columns to display. `Hostname` is displayed by default. Other values include `HWControlPoint`, `LCDS`, `Network-Interfaces`, `Status`, `PowerStatus` and `all`. This flagoption cannot be used with the `–l` flagoption.

–S
Sorts output first by hardware control point and then by host name.

Parameters

None.

Security

The command requires root access to the cluster management server and a user ID with access to the IBM.NodeHwCtrl resource class in the RMC `ctrmc.acls` ACL file.

This command could require a `systemid` file. For more information, see the `systemid` man page.

Exit Status

Hostname
Host name for management of the node. This value will be truncated to seventeen characters. The seventeenth character is a ~ to indicate that truncation was used.

HWControlPoint
Host name of the network adapter for the hardware control point. This value will be truncated to seventeen characters. The seventeenth character is a ~ to indicate that truncation was used.

Status
Indicates if the node is reachable on the network and if the RMC subsystem on the node can communicate with the RMC subsystem on the management server. The valid states are `0` (off), `1` (on) and `127` (unknown). The English representation will be used except when using a delimiter.

PowerStatus
Indicates the current power status of the node. The valid states are `0` (off), `1` (on), `127` (unknown), and `128` (hardware control not configured). The English representation will be used except when using a delimiter.

NetworkInterface
Contains the *Name* of the device and the *OpState*.

Name
The name of the network interface. For example, `eth0` on Linux and `en0` on AIX. Switch Network interfaces are also shown.

OpState
Represents the current state of the network interface. The valid states are:

1
Online

2
Offline

Examples

1. The following command returns information in the default format:

```
csmstat

-------------------------------------------------------------------------
Hostname         HWControlPoint  Status  PowerStatus Network-Interfaces
-------------------------------------------------------------------------
clsn10.pok.ibm.c~  /dev/tty2     off     off         unknown
clsn11.pok.ibm.c~  /dev/tty3     off     off         unknown
clsn12.pok.ibm.c~  /dev/tty4     unknown on          unknown
```

```
clsn13.pok.ibm.c~  /dev/tty4     unknown on          unknown
clsn14.pok.ibm.c~  /dev/tty4     unknown off         unknown
clsn15.pok.ibm.c~  /dev/tty4     unknown on          unknown
clsn16.pok.ibm.c~  /dev/tty4     unknown on          unknown
clsn17.pok.ibm.c~  /dev/tty4     unknown on          unknown
clsn18.pok.ibm.c~  /dev/tty4     on      off         en0-Online
```

2. The following command returns information with the specified delimiter:

```
csmstat -d ::

clsn10.pok.ibm.com::/dev/tty2::0::0::unknown
clsn11.pok.ibm.com::/dev/tty3::0::0::unknown
clsn12.pok.ibm.com::/dev/tty4::127::1::unknown
clsn13.pok.ibm.com::/dev/tty4::127::1::unknown
clsn14.pok.ibm.com::/dev/tty4::127::0::unknown
clsn15.pok.ibm.com::/dev/tty4::127::1::unknown
clsn16.pok.ibm.com::/dev/tty4::127::1::unknown
clsn17.pok.ibm.com::/dev/tty4::127::1::unknown
clsn18.pok.ibm.com::/dev/tty4::1::0::en0-1::
```

3. The following command returns information for the specified column headers:

```
 csmstat -s Status,Network-Interfaces

-----------------------------------------------
Hostname          Status  Network-Interfaces
-----------------------------------------------
clsn10.pok.ibm.c~ on      en0-Online ml0-Offline
clsn11.pok.ibm.c~ on      sn1-Online sn0-Online at2-Online at1-Online at0-Online
                          en1-Offline en0-Online ml0-Offline
clsn12.pok.ibm.c~ on      en0-Online en1-Offline ml0-Offline sn1-Online sn0-Online
clsn13.pok.ibm.c~ off     unknown
clsn14.pok.ibm.c~ on      en0-Online en1-Offline at0-Online at1-Online at2-Online
                          at3-Online sn1-Online sn0-Online ml0-Offline
clsn15.pok.ibm.c~ on      en0-Online en1-Offline at0-Online at1-Online at2-Online
                          at3-Online ml0-Offline sn1-Online sn0-Online
clsn16.pok.ibm.c~ unknown unknown
```

Location

`/opt/csm/bin/csmstat`

csplit Command

Purpose

Splits a file into individual files.

Syntax

csplit [**-f** *Prefix*] [**-k**] [**-n** *Number*] [**-s**] *File Argument* ...

Description

The **csplit** command copies the specified file and separates the copy into segments. The original input file, which remains unaltered, must be a text file.

The **csplit** command writes the segments to files **xx00** . . . **xx99**, depending on how many times the *Argument* parameter is specified (99 is the maximum). By default, the *Argument* parameter expects a line number. The following rules apply when you specify multiple line numbers:

- File **xx00** contains the lines from the beginning of the original file up to, but not including, the line number specified in the first *Argument* parameter.

- File **xx01** contains lines beginning with the number specified by the first *Argument* parameter up to, but not including, the line referenced by the second *Argument* parameter. Each line number specified as an argument marks the beginning of a new file.
- File **xx***nn* (the last file created) contains lines beginning with the number specified by the last *Argument* parameter through the end of the file.

For example, if the original file had 108 lines and you entered:

```
csplit orginal.txt 11 72 98
```

the **csplit** command would create four files: the **xx00** file would contain lines 1-10, the **xx01** file would contain lines 11-71, the **xx02** file would contain lines 72-97, the **xx03** file would contain lines 98-108.

The *Argument* parameter can also contain the following symbols and pattern strings:

Item	Description
/*Pattern***/**	Creates a file that contains the segment from the current line up to, but not including, the line containing the specified pattern. The line containing the pattern becomes the current line.
%*Pattern***%**	Makes the line containing the specified pattern the current line, but does not create a file for the segment.
+*Number*	Moves forward the specified number of lines from the line matched by the preceding pattern. For example, `/Page/+5` searches for `Page`, then advances 5 lines.
-*Number*	Moves backward the specified number of lines from the line matched by the preceding pattern. For example, `/Page/-5` searches for `Page`, then backs up 5 lines.
{*Number***}**	Repeats the preceding option the specified number of times. This number can follow any pattern or line number. If it follows a pattern, the **csplit** command reuses that pattern the specified number of times. If it follows a line number, the **csplit** command splits the file from that point for the number of lines specified by the line number.

Put quotation marks around all patterns that contain spaces or other characters special to the shell. Patterns may not contain embedded new-line characters. In an expression such as `[a-z]`, the - (minus sign) means *through*, according to the current collating sequence. A collating sequence may define *equivalence classes* for use in character ranges.

Flags

Item	Description
-f *Prefix*	Specifies the prefix to be used for the created file segments. The default value for this variable is **xx**.
-k	Leaves created file segments intact in the event of an error.
-n *Number*	Changes the number of decimal places used in the created file names. The default is two decimal places, or **xx00** . . . **xx99**. If you specify the **-n 4** flag, for example, new files are named **xx0000** . . . **xx0099**.
-s	Suppresses the display of character counts.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.

Item	Description
>0	An error occurred.

Examples

1. To split the text of book into a separate file for each chapter, enter:

   ```
   csplit book "/^ Chapter *[k.0-9]k./" {9}
   ```

   This creates 10 files, **xx00** through **xx09**. The **xx00** file contains the front matter that comes before the first chapter. Files **xx01** through **xx09** contain individual chapters. Each chapter begins with a line that contains only the word `Chapter` and the chapter number.
2. To specify the prefix `chap` for the files created from book, enter:

   ```
   csplit  -f chap book "/^ Chapter *[k.0-9]k./" {9}
   ```

   This splits book into files named **chap00** through **chap09**.

Files

Item	Description
/usr/bin/csplit	Contains the **csplit** command.

csum Command

Purpose

The **csum** command calculates a message digest for the specified files using the specified hash algorithm.

Syntax

csum [**-o** *outputfile*] [**-h** *algorithm*] [**-a**] [*File1*, *File2*, ... | **-**]

csum -i *inputfile*[**-h** *algorithm*]

Description

The **csum** command calculates a message digest for the specified files using the specified hash algorithm. This provides a reliable way to verify file integrity.

The **csum** command writes message digests to a specified file which can later be used to verify file integrity. Note that a file can be specified using absolute or relative path names.

Specifying multiple **-i**, **-o** or **-h** flags is not considered an error; the last instance of the flag specified will be used. However, it is an error to use both the **-i** and **-o** flags at the same time.

Flags

Item	Description
-	Specifies input from stdin.
-a	Specifies that one message digest will be generated for all files.

Item	Description
-h *algorithm*	Specifies which hash algorithms the **csum** command will use to generate a message digest or verify the message digest values when using the **-i** option. The following options are available: • SHA1: Uses the SHA-1 algorithm to generate a 20 byte message digest. • MD5: Uses the MD5 algorithm to generate a 16 byte message digest. **Note:** these options are case sensitive. If this **-h** option is not used, then the **csum** command will default to using the MD5 algorithm for both generating and verifying message digests.
-i *inputfile*	Specifies an input file, generated by the **-o** flag, which contains trusted message-digest values. The **csum** command calculates the message-digest values of the files specified in the input file and verifies that they match the actual message-digest values of the existing file. The **-h** flag should be used with the **-i** flag to specify which cryptographic hash algorithm is used to generate the input file. If it is not specified, the MD5 algorithm will be used. If a file specified in the input file generates a message-digest value different than the value stored in the input file or the file does not exist, the test for that file will fail and the **csum** command will continue to process the files specified in the input file.
-o *outputfile*	Specifies an output file that the **csum** command will use to write message-digest values. This flag cannot be used with the **-i** flag. If the file specified already exists, it will be overwritten.

Exit Status

The command returns the following values:

Item	Description
0	Success.
>0	An error occurred.

Examples

1. To calculate the message digest for the files `cars` and `trucks`, type:

   ```
   csum cars trucks
   ```

   Because the **-h** option is not specified, MD5 values are calculated for the files `cars` and `trucks`.

   If 9875DD0B18C15899988F29E9D85346A4 and E8C3ABB5E1D48FA519135EAB0FE40932 are the MD5 values for `cars` and `trucks`, respectively, the **csum** command outputs the following:

   ```
   9875DD0B18C15899988F29E9D85346A4        cars
   E8C3ABB5E1D48FA519135EAB0FE40932        trucks
   ```

2. To calculate the message digest for all files with file names beginning with *file* and store the output in a file called `mdvalues`, type:

   ```
   csum -o mdvalues file*
   ```

The output file, `mdvalues`, will contain the following text if the directory where the **csum** command is executed contains the files `file1`, `file2`, and `file3` and the MD5 values for those files are as listed below:

```
B026324C6904B2A9CB4B88D6D61C81D1        file1
26AB0DB90D72E28AD0BA1E22EE510510        file2
D7FCE9FEE471194AA8B5B6E47267F03         file3
```

3. To verify that the message digests in the file `mdvalues` match the current message-digest values for those same files, type:

```
csum -i mdvalues
```

4. To calculate the message digest for the file **user.dat** using the SHA-1 algorithm, type:

```
csum -h SHA1 user.dat
```

If the SHA-1 value for the **user.dat** file is A77CBB748AC336558AFA1AE7F2B73F3765728E7B, the **csum** command will output the following:

```
A77CBB748AC336558AFA1AE7F2B73F3765728E7B        user.dat
```

Location

/usr/bin/csum

ct Command

Purpose

Dials an attached terminal and issues a login process.

Syntax

ct [**-h**] [**-s***Speed*] [**-v**] [**-w***Number*] [**-x***Number*] *TelephoneNumber* ...

Description

The **ct** command is a Basic Networking Utilities (BNU) command that enables a user on a remote terminal, such as an 3161, to communicate with a workstation over a telephone line attached to a modem at each end of the connection. The user on the remote terminal can then log in and work on the workstation.

A user on the local system issues the **ct** command with the appropriate telephone number to call the modem attached to the remote terminal. When the connection is established, the **ct** command issues a login prompt that is displayed on the remote terminal. The user on the remote terminal enters a login name at the prompt and opens a new shell. The user at the remote terminal then proceeds to work on the workstation just like a local user.

The **ct** command is useful in the following situations:

- A user working off-site needs to communicate with a local system under strictly supervised conditions, and the local user does not want to disclose the workstation's phone number. Because the local system contacts the remote terminal, the remote user does not need to know the telephone number of the local system. Additionally, the local user issuing the **ct** command can monitor the work of the remote user.
- The cost of the connection should be charged either to the local site or to a specific account on the calling workstation. If the remote user has the appropriate access permission and can make outgoing calls on the attached modem, that user can make the equivalent of a collect call. The remote user calls the specified local system, logs in, and issues the **ct** command with the telephone number of the remote terminal, but without the **-h** flag. The local system hangs up the initial link so that the remote terminal is free for an incoming call and then calls back the modem attached to the remote terminal.

If there are no free lines, the **ct** command displays a message to that effect and asks if the local user wants to wait for one. If the reply is no, the **ct** command hangs up. If the local user wants to wait for a free line, the **ct** command prompts for the number of minutes to wait. The **ct** command continues to dial the remote system at one-minute intervals until the connection is established or until the specified amount of time has elapsed.

In order to establish a **ct** connection, the remote user contacts the local user with a regular telephone call and asks the local user to issue the **ct** command. However, if such connections occur regularly at your site, your system administrator may prefer to set up BNU in such a way that a specified local system automatically issues the **ct** command to one or more specified terminals at certain designated times.

Notes:

1. Before issuing the **ct** command, be certain that the remote terminal is attached to a modem that can answer the telephone.
2. If the user issuing the **ct** command does not have root authority, the port used for the connection must be a shared or delayed port. Otherwise, the remote login will fail. In addition, for the **ct** command to succeed on a shared or delayed port, the user invoking the command must be a member of the UNIX-to-UNIX copy program (uucp) user group.

The **ct** command is not as flexible as the BNU **cu** command. For example, the user cannot issue commands on the local system while connected to a remote system through the **ct** command. However, the **ct** command does have two features not available with the **cu** command:

- The user can instruct the **ct** command to continue dialing the specified telephone number until the connection is established or a set amount of time has elapsed.
- The user can specify more than one telephone number at a time to instruct the **ct** command to continue dialing each modem until a connection is established over one of the lines.

If the local user specifies alternate dialing paths by entering more than one number on the command line, the **ct** command tries each line listed in the BNU **Devices** file(s) (by default, the **/etc/uucp/Devices** file) until it finds an available line with appropriate attributes or runs out of entries. If there are no free lines, the **ct** command asks if it should wait for one and, if so, for how many minutes. The **ct** command continues to try to open the dialers at one-minute intervals until the specified time is exceeded. The local user can override this prompt by specifying a time with the **-w***Number* flag when entering the command.

After the user logs off, the **ct** command prompts the user on the remote terminal with a reconnect option; the system can either display a new login prompt or drop the line.

Flags

Item	Description
-h	Prevents the **ct** command from hanging up the current line to answer an incoming call.
-s*Speed*	Specifies the rate at which data is transmitted. The default is 1200 baud.
-v	Allows the **ct** command to send a running narrative to standard error output.
-w*Number*	Specifies the maximum number of minutes that the **ct** command is to wait for a line. The command then dials the remote modem at one-minute intervals until the connection is established or until the specified time has elapsed.
-x*Number*	Starts debugging, which displays detailed information about the command's execution on standard error output on the local system. The *Number* variable specifies the debugging level, and is a single digit from 0 to 9. The recommended debugging level is 9.

Item	Description
TelephoneNumber	Specifies the telephone number of the modem attached to the remote terminal. The *TelephoneNumber* variable can include the digits 0 through 9, - (minus signs) representing delays, = (equal signs) representing secondary dial tones, * (asterisks), and # (pound signs). The telephone number can contain a maximum of 31 characters.

Examples

1. To dial a modem attached to a remote terminal with an internal telephone number, enter:

   ```
   ct 41589
   ```

   The internal telephone number of 4-1589 is dialed. The - (hyphen) is optional. The system responds:

   ```
   Allocated dialer at 1200 baud
   Confirm hang_up? (y to hang_up)
   ```

2. To dial a modem attached to a remote terminal with a local telephone number, enter:

   ```
   ct  -w3 9=5553017
   ```

   The **ct** command dials the local telephone number of 555-3017, where dialing 9 is required to reach an outside dial tone. A three-minute wait is specified as the maximum number of minutes that the **ct** command is to wait for a line.

3. To dial a modem attached to a remote terminal with a long-distance telephone number, enter:

   ```
   ct  -w5 9=12345557003
   ```

   The command dials the long-distance telephone number of 1-234-555-7003, where 9 is required to reach an outside dial tone. A five-minute wait is specified as the maximum number of minutes that the **ct** command is to wait for a line.

Files

Item	Description
/usr/bin/ct	Contains the **ct** command.
/etc/uucp/Devices	Lists information about available devices.
/etc/uucp/Dialcodes	Contains dialing code abbreviations.
/etc/uucp/Dialers	Defines modem dialers.
/etc/uucp/Systems	Lists accessible remote systems.
/etc/uucp/Sysfiles	Specifies alternate files to be used as **Systems**, **Devices**, and **Dialers** files.

ctaclfck Command

Purpose

Verifies the contents of a cluster security services ACL file.

Syntax

`ctaclfck -f` *acl_file_name* `[-s] [-c] [-u` *user_name*`] [-v] [-h]`

Description

The `ctaclfck` command checks the contents of the cluster security services ACL file specified by the `-f` flag. The check is limited to syntactical errors; a semantic check is not performed.

The command opens the ACL file, and reads and compiles one ACL entry at a time. If the command encounters an error, it will report the error to standard output. If the `-c` flag is provided, the command will continue processing after encountering errors until it reaches the end of the file. Otherwise processing will stop after the first error is found and reported.

The `-u` flag directs the command to verify the ACL file contents owned by the specified operating system user identity. The command user must have permission to change to the home directory of the user specified by the `-u` flag, and must also have permission to read files in that directory. If the `-s` flag is specified along with the `-u` flag, the command user must also have permission to set its effective user identity to this identity (see the man page for the operating system command `su` for examples).

When the `-u` flag is specified, the file name provided in the `-f` flag is expected to be the base name of a file that resides in the home directory of the named user. In this case, the file name specified by the `-f` flag must not contain any directory names, including the `./` and `../` directories.

If the `-s` flag is specified, the command creates a file to contain the compiled contents of the ACL file. This permits applications to compile the ACL data buffer in advance to starting the application that uses it, saving the application this processing during its startup procedure or its ACL reading process. The compiled ACL file will have the same name as the ACL file with the extension `.cacl`. The ownership and file system permissions of the new `*.cacl` file will be set to the same ownership and permissions as the ACL file. If the ACL file is not currently owned by the command user, the command user must be capable of changing its effective user identity to the identity of the user that owns the ACL file. If the command is unable to do this, it will not create the ACL buffer file, but will complete verification of the ACL file.

The command checks for the correct ACL entry type, for the proper identity format, and for a valid permission. A valid permission is defined as one containing only operations that are defined by the permission template. The permission template set defined by cluster security services and used by this command follows.

Entry Type	Description
r	• Format: 0x1 • Permission: read • Operation: generic read operation
w	• Format: 0x2 • Permission: write • Operation: generic write operation
c	• Format: 0x4 • Permission: control • Operation: generic control operation or RMC refresh configuration operation
x	• Format: 0x8 • Permission: run • Operation: generic execute operation
C	• Format: 0x10 • Permission: cancel • Operation: generic cancel operation

Entry Type	Description
q	• Format: 0x20 • Permission: query • Operation: RMC query resource operation
l	• Format: 0x40 • Permission: list • Operation: RMC enumerated resources operation
e	• Format: 0x80 • Permission: event • Operation: RMC event registration, unregistration, and querying
d	• Format: 0x100 • Permission: define • Operation: RMC define and undefine resource operation
v	• Format: 0x200 • Permission: validate • Operation: RMC validate resource handle operation
s	• Format: 0x400 • Permission: set • Operation: RMC set attribute operation

If the `-u` flag is specified, the command searches for the ACL file in the home directory of the specified user. The user must own the file and the permission must be write-only by the user. When the `-u` flag is specified, the ACL file name specified by the `-f` flag must not contain a relative or full path to the file; it must specify the file name only.

Flags

-f *acl_file_name*
Specifies the cluster security services ACL file to be verified. The file name can be a full or relative path name, unless the `-u` flag is specified.

-s
Caches the ACL buffer (that resulted from the compilation of the ACL file) into a file. If the ACL file is not owned by the command user, the command user must be able to set its effective user identity to the owner of the ACL file.

-c
Instructs the command to continue after encountering errors until the end of file is reached. All errors encountered will be reported regardless of whether or not the `-v` flag is specified. If not specified, command processing will stop after the first error is encountered and reported.

-u *user_name*
Specifies the user name in whose home directory the ACL file resides. When this flag is used, the file name specified by the `-f` flag must be the base name of a file that resides in the named user's home directory; the file cannot contain any directory information, including the `./` and `../` directory names.

-v
Writes the command's verbose messages to standard output.

-h
Writes the command's usage statement to standard output.

Security

The file system permission of the ACL file is determined by the end user or the application owning the file. If the invoker does not have sufficient authority to read the file or to create the requested compiled ACL file with the same ownership, the command fails.

Restrictions

The `ctaclfck` command works only on ACL files formatted for cluster security services.

Examples

1. To verify the contents of the ACL file `/my_acl_file`:

```
ctaclfck -f /my_acl_file
```

2. To verify the contents of the ACL file `../my_acl_file` (relative to the current directory) and provide detailed output:

```
ctaclfck -f ../my_acl_file -v
```

3. To completely verify the contents of the ACL file `/u/fluffy/my_acl_file`, which is owned by the operating system user `fluffy`, and store the compiled ACL buffer into a file for later use:

```
ctaclfck -c -u fluffy -f  my_acl_file  -v -s
```

Location

/opt/rsct/bin/ctaclfck
Contains the `ctaclfck` command

ctadmingroup Command

Purpose

Defines a cluster administration group.

Syntax

To define a group:

`ctadmingroup` [`-h`] [`-TV`] *group_name*

To remove a group:

`ctadmingroup -u` [`-h`] [`-TV`] [*group_name*]

Description

The `ctadmingroup` command is used to define a cluster administration group. This command sets group ownership for trace files, so users who belong to a cluster administration group have the permissions needed to examine trace files that are produced by Reliable Scalable Cluster Technology (RSCT) subsystems. `ctadmingroup` changes existing trace files to the new permissions and group ownership. Trace files, which are created after the `ctadmingroup` command is run, contain the new permissions. This command does not create the specified group, nor does it add users to this group; it only gives users of this group access to the trace files.

If you run the `ctadmingroup` command with:

- a different group name, the new group that is specified becomes the cluster administration group, thus replacing the previous group.
- no flagsoptions or parameters, it displays the group name and ID of the cluster administration group. If no cluster administration group is defined, this command does not produce any output.
- the `-u` flagoption, it removes the cluster administration group. After the group is removed, users who belong to that group might not be able to examine trace files. If no cluster administration group is defined, this command does not produce any output.

The location of the trace file of the security subsystem is configurable. To determine the location of the trace file, the `ctadmingroup` command requests information from the `/var/ct/cfg/ctcasd.cfg` file (if it is present) and the `/opt/rsct/cfg/ctcasd.cfg` file.

Parameters

group_name
Specifies the name of the cluster administration group. This group must exist in the group database (`/etc/group`, for example).

Flags

-u
Removes the cluster administration group. After the group is removed, users who belong to that group might not be able to examine trace files. If no cluster administration group is defined, this command does not produce any output.

-h
Writes the command usage statement to standard output.

-T
Writes the command trace messages to standard error. For your software service organization use only.

-V
Writes the command verbose messages to standard output.

Files

/etc/group
The group database.

/var/ct/cfg/ctgroups
Stores the administration group name and caches the corresponding group ID.

/var/ct/cfg/ctcasd.cfg
The primary location of the cluster security configuration file, which contains the location of the trace file of the security subsystem.

/opt/rsct/cfg/ctcasd.cfg
The secondary location of the cluster security configuration file. The `ctadmingroup` command requests information from this file if the `/var/ct/cfg/ctcasd.cfg` file is not present.

Exit status

0
The command has run successfully.

1
The group name that was specified on the command line is not in the group database.

2
An internal error occurred.

3
An incorrect flagoption was entered on the command line.

4
An incorrect operand was entered on the command line.

Security

Only `root` users can run this command.

Standard output

When the `-h` flagoption is specified, this command usage statement is written to standard output. All verbose messages are written to standard output.

Standard error

All trace messages are written to standard error.

Restrictions

Unpredictable results can occur if the mapping of the group name and group ID is changed after the command is run.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIXpackage for Linux.

Location

/opt/rsct/bin/ctadmingroup

Examples

1. To display the group name and ID of the cluster administration group, enter:

```
ctadmingroup
```

ctags Command

Purpose

Makes a file of tags to help locate objects in source files.

Syntax

ctags [**-u** | **-x**] [**-B** | **-F**] [**-a**] [**-m**] [**-o**] [**-t**] [**-v**] [**-w**] [**-f** *TagsFile*] *File*. ..

Description

The **ctags** command creates a tags file for use with the ex and vi editors from the specified C, Pascal, FORTRAN, yacc, lex, and LISP source files. The tags file consists of locators of programming language specific objects (such as functions and type definitions) within the source files. A locator consists of the object name, the file in which it is defined, and either a basic regular expression or a line number that can be used in searching for the object definition. Specifiers are given in separate fields on the line, separated by spaces or tabs. Using the tags file, ex and vi can quickly find these object definitions.

The following file name suffixes are supported by the **ctags** command:

Item	Description
.c	Treated as C-language source code and searched for C routine and macro definitions.
.h	Treated as C-language source code and searched for C routine and macro definitions.
.f	Treated as FORTRAN-language source code.
.l	Treated as LISP-language source code if its first nonspace character is **[** (open bracket), **(** (open parenthesis), or **;** (semicolon). Treated as lex-language source code otherwise.

File names ending with any other suffixes are first examined to see if they contain any Pascal or FORTRAN routine definitions. If not, they are processed again as C-language source code. Files without a **.** (dot) suffix are processed as C-language source code.

The **main** tag is treated specially in C programs. The tag formed is created by prefixing **M** to the file name, removing a trailing **.c** (if any), and removing the leading path name components. This makes use of **ctags** practical in directories with more than one program.

Notes:

1. Recognition of the keywords **function**, an address specification for the **subroutine**, and **procedure** in FORTRAN and Pascal code ignores block structure. The **ctags** command may yield inadequate results if any two Pascal procedures have the same name, even though they are in different blocks.
2. The **ctags** command does not recognize **#if** and **#ifdef** statements.
3. If both the **-B** and **-F** options are specified, the last one specified will take precedence.
4. The **-x** option takes precedence over any options (**-a**, **-u**, or **-f**) that would otherwise create a tags file.
5. When the **-v** option is specified, the **-x** option is implied.
6. The output of the **ctags** command is always sorted by object identifier.

Flags

Item	Description
-a	Appends to the tags file. After appending, **ctags** sorts the tags file.
-B	Causes **ctags** to use backward searching patterns (?. . .?).
-F	Causes **ctags** to use forward searching patterns (/. . ./). This is the default searching pattern.
-f *TagsFile*	Creates a tags file with the name specified by *TagsFile* instead of the default **tags** file.
-m	Causes **ctags** to not create tags for macro definitions.
-o	Causes **ctags** to generate line numbers for typedefs instead of a basic regular expression which is used in searching for the object definition.
-t	Creates tags for typedefs. This flag is on by default due to standards conformance.
-u	Updates the specified files in tags; that is, all references to them are deleted, and the new values are appended to the file. This flag may slow the processing of the command. (It is usually faster to simply rebuild the tags file.)
-v	Produces an index of the form expected by the **vgrind** command on the standard output. This listing contains the function name, file name, and page number (assuming 64-line pages).
-w	Causes **ctags** to suppress diagnostic warning messages.

Item	Description
-x	Causes the **ctags** command to display a list of object names, the line number and file name on which each is defined, as well as the text of that line. This provides a simple, readable, function index. If you specify this flag, the **ctags** command does not build, update, or append a tags file, but writes to standard output.

Examples

1. To write the output of the **ctags** command to standard output for the C-language source files, **x.c**, **y.c**, and **z.c**, enter:

```
ctags -x x.c y.c z.c
```

2. To create a tags file named **foo_tags** for all the C-language source files within the current directory, enter:

```
ctags -f foo_tags *
```

3. To add additional tags,including type definitions, to the **foo_tags** tags file for the C-language source file **zip.c**, enter:

```
ctags -utf foo_tags zip.c
```

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	An error occurred.

Files

Item	Description
usr/bin/more/tags	Output tags file.

ctcasd Daemon

Purpose

Provides and authenticates the credentials of the RSCT host-based authentication (HBA) and enhanced host-based authentication (HBA2) security mechanisms for the cluster security services.

Syntax

`ctcasd` [`-b`]

Description

The `ctcasd` daemon is used by the cluster security services library when the RSCT HBA security mechanism is configured and active within the cluster environment. The cluster security services use `ctcasd` when service requesters and service providers try to create a secured execution environment through a network connection. `ctcasd` is not used when service requesters and providers establish a secured execution environment through a local operating system connection such as a UNIX domain socket.

When a service requester and a service provider have agreed to use HBA authentication through the cluster security services, the cluster security services library uses `ctcasd` to obtain and authenticate HBA credentials. Cluster security services does not provide a direct interface to the daemon that can be invoked by user applications.

The `ctcasd` daemon can be started or stopped using system resource controller (SRC) commands.

During startup, the daemon obtains its operational parameters from the `ctcasd.cfg` configuration file. The daemon expects to find this file in the `/var/ct/cfg/` directory. System administrators can modify the operational parameters in this file to suit their needs. If this file is not located, the daemon will use the default configuration stored in `/opt/rsct/cfg/ctcasd.cfg`.

RSCT HBA and HBA2 credentials are derived from the local node's private and public keys. These keys are located in files that are configured in `ctcasd.cfg`. These credentials are encrypted using the public key of the receiving node. Public keys for the nodes within the cluster are stored in a trusted host list file on each node. The location of this file is also defined in the `ctcasd.cfg` configuration file. The system administrator is responsible for creating and maintaining this trusted host list, as well as for synchronizing the lists throughout the cluster.

If the daemon detects that both the node's public and private key files are not present, `ctcasd` assumes that it is being started for the first time and creates these files. The daemon also creates the initial trusted host list file for this node. This file contains an entry for `localhost` and the host names and IP addresses associated with all AF_INET-configured and active adapters that the daemon can detect. Inadvertent authentication failures could occur if the public and private key files were accidentally or intentionally removed from the local system before the daemon was restarted. `ctcasd` creates new keys for the node that do not match the keys stored on the other cluster nodes. If RSCT HBA and HBA2 authentications suddenly fails after a system restart, this is a possible source of the failure.

Critical failures detected by the daemon that cause shutdown of the daemon are recorded to persistent storage. In AIX-based clusters, records are created in the AIX error log and the system log. In Linux-based clusters, records are created in the system log.

Flags

`-b`
Starts the daemon in bootstrap mode. The daemon runs as a foreground process and is not controlled by the system resource controller (SRC).

Restrictions

- The `ctcasd` daemon does not encrypt the HBA identity credentials.
- Cluster security services supports its own file formats, private key formats, and public key formats only. Cluster security services does not support secured remote shell formats.

Implementation specifics

This daemon is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/opt/rsct/bin/ctcasd
Contains the `ctcasd` daemon

Files

/opt/rsct/cfg/ctcasd.cfg
Default configuration for the `ctcasd` daemon

/var/ct/cfg/ctcasd.cfg
Configuration for the `ctcasd` daemon, which can be modified by the system administrator

/var/ct/cfg/ct_has.pkf
Default location of the cluster security services public key file for the node

/var/ct/cfg/ct_has.qkf
Default location of the cluster security services private key file for the node

/var/ct/cfg/ct_has.thl
Default location of the cluster security services trusted host list for the node

ctctrl Command

Purpose

Modifies or displays the trace attributes of system components. You can specify persistent attribute values for components that have not yet been created.

Syntax

To modify the trace attributes of some or all components, use the following command:

ctctrl [**-nru**] *ComponentSelector ... subcommand ...*

To dump component buffers into files, use the following command:

ctctrl [**-ru**] {**-D** [**-d** *dirName*] } *ComponentSelector ...*

To specify persistent attribute values for components that have not been created yet, use the following command:

ctctrl -p [**-ru**] *ComponentSelector ... subcommand ...*

To specify persistent attribute values that will take effect after the next restart, use the following command:

ctctrl -P [**-ru**] *ComponentSelector ... subcommand ...*

To delete persistent attribute customizations, use the following command:

ctctrl -x {**-P**|**-p**} [**-ru**] *ComponentSelector ...*

To query trace attributes of existing components or to query existing persistent attribute customization, use the following command:

ctctrl -q [**-rupP**] {*ComponentSelector ...*}

To display a usage message, use the following command:

ctctrl {**-h** | **-?**}

To enable or disable memory tracing for all components persistently, use the following command:

ctctrl -P {**memtraceon** | **memtraceoff**}

The values of the *ComponentSelector* parameter are as follows:

-c
componentPatternList

-l
aliasPatternList

-t
typePatternList

Each list consists of one or more patterns that are separated by blank spaces or commas. Patterns can contain special characters as described by the **fnmatch** subroutine. You can use the following pattern characters:

- **?**

- *****
- **[ ]**

You cannot use character classes and collation sequences inside brackets (**[ ]**). Specifying **-c all** selects all components, if no other *ComponentSelector* parameter is specified.

Description

The **ctctrl** command modifies or displays the trace settings of some or all components. Components are selected by name, by alias, or by type or subtype. The **ctctrl** command can also be used with the **-p** or **-P** flag to specify persistent attribute customization. See the Persistent Customizations section.

To enable or disable component-level tracing for all components immediately and persistently, specify the **memtraceon** or **memtraceoff** subcommand with the `-P` flag. You cannot specify other flags or subcommands with the **-P** flag. You must use the `bosboot` command to make settings persistent across boots.

The modified attribute depends on the subcommand that is passed to the **ctctrl** command. Multiple subcommands can be used in a single **ctctrl** invocation. You can specify the following subcommands:

Item	Description
`memtraceon`	Turns on memory trace mode.
`memtraceoff`	Turns off memory trace mode.
`memtraceresume`	Resumes memory trace mode.
`memtracesuspend`	Suspends the memory trace mode.
`memtracebufsize=sz`	Changes the size of the private buffer allocated in memory trace mode.
`memtraceminimal`	Changes memory trace mode level to 1.
`memtracenormal`	Changes memory trace mode level to 3.
`memtracedetail`	Changes memory trace mode level to 7.
`memtracemax`	Changes the level of the memory trace mode to the maximum detail level 9.
`memtracelevel=d`	Changes the level of trace of the memory trace mode. Sets it to the specified level.
`memtracefilltime`	Displays the data retention time (that is, the estimated time to fill the private memory buffer). This is available only if the memory trace mode is on.
`systraceon`	Turns on the tracing through the system trace.
`systraceoff`	Turns off the tracing through the system trace.
`systraceminimal`	Changes system trace mode level to 1.
`systracenormal`	Changes system trace mode level to 3.
`systracedetail`	Changes system trace mode level to 7.
`systracemax`	Changes the level of system trace mode to the maximum detail level 9.
`systracelevel=d`	Changes the level of trace used to trace through the system trace. Sets it to the specified value.

Note: The **memtracesuspend**, **memtraceresume**, and **memtracefilltime** subcommands cannot be used with the **-p** or the **-P** flag, because these subcommands cannot be used in persistent customizations.

Other subcommands that are not in the previous list can be recognized by individual components. A subcommand that is not recognized by a component is ignored.

Current attribute values can be displayed by using the **-q** flag. If you do not specify the *ComponentSelector* parameter, attribute values are displayed for all components that use component-level tracing.

Persistent Customizations

The **-p** and **-P** flags allow attribute values to be specified for system components that have not been created yet. Thus, attributes for newly created components can be customized before the components become active. The **-p** flag is used to specify customizations for components that will be created in the future, but before you restart the AIX operating system. The **-P** flag is used to specify customizations that will take effect after the next restart. These customizations are added to the **/var/adm/ras/raspertune** file. You must run the **bosboot** command to save these customizations in the boot image and restart the AIX operating system for the customizations to take effect.

The component specified by the *ComponentSelectors* parameter with the **-p** and **-P** flags can contain pattern-matching characters. Thus, a persistent customization can apply to more than one component. In addition, multiple customizations can apply to the same component, if different components are used. If conflicting attribute values are specified in multiple customizations, the last customization takes precedence. If a customization already exists for a specified component, the new customization replaces the old one.

You can specify multiple components with the *ComponentSelectors* parameter when persistent customizations are specified. In all cases, using multiple selectors is equivalent to specifying multiple commands, each with a single component selector. For example, the customization `ctctrl -p -l hdisk0 -l hdisk1 memtracenormal` is equivalent to the following two customizations:

```
ctctrl -p -l hdisk0 memtracenormal
ctctrl -p -l hdisk1 memtracenormal
```

When you use the `-D` flag, a snapshot of trace buffers for selected components is dumped into files. The default directory is `/var/adm/ras/trc_ct`, but you can specify an optional destination directory. One trace file per component is used; all files are named with the full components names. The files are generated and managed in the same way the `trace` command does for multiple processor files.

Customizations specified with the **-p** or **-P** flag are not deleted even after they are used. Therefore, a single customization can affect multiple new components. You can specify the **-x** flag to delete persistent customizations. You must specify the *ComponentSelector* parameter identically to the way you specify it when the customization is created. For example, if a customization is created with the component specified by `-l hdisk0`, the customization cannot be deleted with the component specified by `-l hdisk[0]`, even though both components match the same component alias. When a persistent customization is deleted, no change is made to the attributes of components that are created when the customization is active.

Persistent customizations that are deleted with the **-x** and **-P** flags remain in effect unless you run the **bosboot** command and restart the AIX operating system. You can delete a persistent customization that is created with the **-P** flag after the restart by using the **-x** and **-p** flags. In this case, the customization are active again if you restart the AIX operating system.

If you do not know the customizations that have been made but want to restore the default system setting, you can use one of the following ways:

- In the **/var/adm/ras/raspertune** file, delete the lines relevant to the customizations. Then run the **bosboot** command and restart the AIX operating system.
- Read the **/var/adm/ras/raspertune** file to figure out the appropriate flags and parameters that have been specified. Then use the **-x** flag to delete the customizations as shown in Example “11” on page 765. Run the **bosboot** command and restart the AIX operating system.

The **-r** and **-u** flags can be used when specifying persistent customizations. Using one flag specifies a different name space for the specified component selectors. Using both flags at the same time is equivalent to two separate command invocations, each with one of the flags. For example, the persistent

customization `ctctrl -p -l hdisk0 -u -r memtracedetail` is equivalent to the following two separate customizations:

```
ctctrl -p -l hdisk0 -u memtracedetail
ctctrl -p -l hdisk0 -r memtracedetail
```

The following persistent customizations are all distinct, and can be modified or deleted independently.

```
ctctrl -p -l hdisk0 memtracedetail
ctctrl -p -l hdisk0 -r memtracedetail
ctctrl -p -l hdisk0 -u memtracedetail
```

Recursive-down customizations (specified by the **-r** flag) take precedence over all other customizations, regardless of the order in which they are specified relative to other non-recursive-down customizations.

You can query persistent customizations by using the **-q** flag with either the **-P** or **-p** flag. Specifying the **-q** flag with the **-P** flag displays lines from the **/var/adm/ras/raspertune** file. Specifying the **-q** flag with the **-p** and **-r** flags displays the persistent customizations that you originally specified with the **-r** flag. Without the **-r** flag, the **-q** and **-p** flags display the persistent customizations that you specify with or without the **-u** flag.

You can specify multiple subcommands for a persistent customization. If you specify conflicting subcommands, the last subcommand is used. For example, the **memtracenormal** and **memtracedetail** subcommands specify different values for the same error-checking attribute, so the last specified subcommand is used.

Flags

Item	Description
`-n`	Applies subcommands immediately. This flag is the default if neither the **-p** nor the **-P** flag is used.
`-c` *componentList*	Specifies a list of component names. Separate the names in the list using a comma or blank space. The `-c all` flag selects all components if it is the only *ComponentSelector*.
`-D`	Takes a snapshot of the component's private memory buffer and dumps it into files (one file per component). The default output directory can be changed with the `-d` flag.
`-d` *dirName*	Specifies the directory used for the dump. The default directory is `/var/adm/ras/trc_ct`. If some files already exist, they are overwritten by the new dump request. The **-p** and **-P** flags are mutually exclusive with the **-d** flag.
`-h` or `-?`	Displays a usage message.
`-l` *aliasList*	Specifies a list of component aliases. Separate the aliases using a comma or blank space.
`-P`	Specifies subcommands that will persist across restarts. You must run the **bosboot** command and restart AIX for these commands to be used.
`-x`	Deletes the persistent customization for the specified components. The *ComponentSelector(s)* must be entered exactly as they were entered when the customization was originally specified.
`-p`	Specifies persistent subcommands. The specified subcommands are applied to newly created components.
`-q`	Displays the component trace settings of the components. This flag can also be used with the **-p** or **-P** flag to display persistent customizations.

Item	Description
`-r`	Applies the subcommands recursively to all subcomponents of the selected components.
`-t` *type_subtype*	Specifies a list of *type* or *type_subtype* names. Separate the names using a comma or blank space. Valid *type* names include `device`, `filesystem`, `network`, `services`, `storage`, and `ui`. A complete list of *type* and *type_subtype* names is in the **/usr/include/sys/ras_base.h** header file.
`-u`	Applies the subcommands recursively to the ancestors of the specified components.

Note: You can use the **-u** and **-r** flags together. You can use multiple **-c**, **-l**, and **-t** flags on the command line.

Exit Status

Item	Description
`0`	The command completes successfully.
`>0`	An error occurs.

Examples

1. To dump the contents of all Component Trace buffers, use the following command:

   ```
   ctctrl -D -c all
   ```

2. To dump the contents of the mbuf Component Trace buffer to `/tmp`, use the following command:

   ```
   ctctrl -D -d /tmp -c mbuf
   ```

3. To query the state of all Component Trace aware components, use the following command:

   ```
   ctctrl -q
   ```

4. To query the state of only the `netinet` components, use the following command:

   ```
   ctctrl -c netinet -q -r
   ```

5. To turn on memory tracing mode for the `socket` component, use the following command:

   ```
   ctctrl memtraceon -c socket
   ```

6. To persistently turn off component tracing for all components, use the following command:

   ```
   ctctrl -P memtraceoff
   ```

   Note: A `bosboot` is required to make the command persistent across boots.

7. To specify a persistent customization for the userdata component of new JFS2 file systems, use the following command:

   ```
   ctctrl -p -c 'jfs2.filesystem.*.userdata' memtraceminimal
   ```

   Note: The existing userdata components are not affected.

8. To specify a customization that will persist across restarts, use the following command:

   ```
   ctctrl -P -c 'jfs2.filesystem.*.userdata' memtraceminimal
   ```

If you run the **bosboot** command and restart AIX, minimal component tracing will be in effect for all JFS2 userdata components.

9. To set minimal component tracing for all JFS2 userdata components, use the following command:

```
ctctrl -npP -c 'jfs2.filesystem.*.userdata' memtraceminimal
```

10. To specify multiple persistent attribute values for the ethernet component, use the following command:

```
ctctrl -P -c ethernet memtraceminimal memtracebufsize=1m
```

11. To delete the customization specified in example 7, use the following command:

```
ctctrl -p -x -c 'jfs2.filesystem.*.userdata'
```

12. To list all persistent, recursive-down attribute customization, use the following command:

```
ctctrl -q -p -r
```

13. To enable all component traces for the netmalloc component, use the following command:

```
ctctrl memtracedetail -c netmalloc
```

or

```
ctctrl memtracelevel=7 -c netmalloc
```

14. To collect **net_malloc_police** trace events in the component trace buffer, use the following command:

```
ctctrl  memtracedetail -c netmalloc.police
```

Location

`/usr/sbin/ctctrl`

Files

Item	Description
`/var/adm/ras/raspertune`	A file containing persistent attribute customization that will be applied after a restart, if you run the **bosboot** command first.
`/var/adm/ras/trc_ct`	The default directory where all snapshots of buffers are saved.
`trc_ct.master`	A master trace file that points to the trace files of all components.

cthactrl Command

Purpose

Controls subsystems within a cluster.

Syntax

`cthactrl -i` <*init_opt*> | `-s` | `-k` | `-b` | `-r` | `-d` | `-z` | `-h`

Description

The `cthactrl` command establishes and controls cluster subsystem information and manages topology services and group services.

Flags

-i <*init_opt*>
Initializes the group services and topology services subsystems, where <*init_opt*> can be specified as:

-c <*cluster_name*>
Specifies the cluster name.

-n <*nodenum*>
Specifies the node number.

-e <*environ*>
Specifies the subdirectory that contains the cluster access modules.

[-p <*portspec*>]
Specifies the UDP port numbers for group services and topology services.

For example:

```
cthactrl -i -c filesys -n 1 -e filesys -p "cthats=12347,cthags=12348"
```

-s
Starts the group services and topology services subsystems.

-k
Stops the group services and topology services subsystems.

-b
Rebuilds the group services and topology services subsystems configurations (`machines.lst`, for example).

-r
Refreshes the group services and topology services subsystems.

-d
Deletes the group services and topology services subsystems.

-z
Uninstalls the group services and topology services subsystems.

-h
Writes the command's usage statement to standard output.

Security

You must have `root` authority to run this command.

Exit Status

0
Successful completion.

non-zero
A failure has occurred.

Restrictions

This command applies to the `cthags` and `cthats` subsystems only.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output.

Examples

1. To initialize the local node as a part of the cluster of `filesys1` and designate 12347 as the UDP port number for `cthags` and 12348 as the UDP port number for `cthags`, enter:

```
cthactrl -i -c filesys1 -n 1 -p "cthats=12347,cthags=12348" -e filesys1
```

2. To start the group services and topology services subsystems (`cthags` and `cthats`), enter:

```
cthactrl -s
```

3. To stop the group services and topology services subsystems (`cthags` and `cthats`), enter:

```
cthactrl -k
```

Location

`/opt/rsct/bin/cthactrl`

cthagsctrl Command

Purpose

Controls the group services subsystem.

Syntax

`cthagsctrl` { `-a` [`-p` *port-number*] `-s` | `-k` | `-d` | `-r` | `-z` | `-h` | `-t` | `-o` }

Description

The `cthagsctrl` control command controls the operation of the group services subsystem (`cthags`) under the control of the system resource controller (SRC).

An instance of the group services subsystem runs on every node of a cluster.

From an operational point of view, the group services subsystem group is organized as follows:

Subsystem
group services

Subsystem group
`cthags`

SRC subsystems
`cthags`

The `cthags` subsystem is associated with the `hagsd` daemon.

The subsystem name on the nodes is `cthags`. There is one subsystem per node and each of these subsystems is associated with the cluster to which the node belongs.

Daemon
`hagsd`

Provides the group services functions.

In general, the `cthagsctrl` command is not issued from the command line. It is normally called by the`cthactrl` command during the creation of the cluster.

The `cthagsctrl` command provides a variety of controls for operating the group services subsystems:

- Adding, starting, stopping, and deleting the subsystems
- Cleaning up the subsystems (deleting them from the cluster)

- Unconfiguring the subsystems from the cluster
- Turning tracing on and off

`Adding the subsystem`

When the `-a` flag is specified, the control command adds the group services subsystems to the SRC. The control command:

1. Makes sure the `cthags` subsystem is stopped.
2. Gets the port number for the `cthags` subsystem from the cluster data.
3. Removes the `cthags` subsystem from the SRC (in case it is still there).
4. Adds the `cthags` subsystem to the SRC.
5. Does not currently add an entry for the `cthags` group to the `/etc/inittab` file. As a result, `cthags` is required to be started by another subsystem when it is needed.

`Starting the subsystem`

When the `-s` flag is specified, the control command uses the `startsrc` command to start the group services subsystem, `cthags`.

`Stopping the subsystem`

When the `-k` flag is specified, the control command uses the `stopsrc` command to stop the group services subsystem, `cthags`.

`Deleting or cleaning the subsystem`

When the `-d` flag is specified, the control command uses the `rmssys` command to remove the group services subsystems from the SRC. The control command:

1. Makes sure the `cthags` subsystem is stopped.
2. Removes the `cthags` subsystem from the SRC using the `rmssys` command.
3. Removes the port number from the `/etc/services` file.

`Turning tracing on`

When the `-t` flag is specified, the control command turns tracing on for the `hagsd` daemon using the `traceson` command.

`Turning tracing off`

When the `-o` flag is specified, the control command turns tracing off (returns it to its default level) for the `hagsd` daemon using the `tracesoff` command.

`Refreshing the subsystem`

The `-r` flag refreshes the `cthags` subsystem.

`Logging`

While they are running, the group services daemons provide information about their operation and errors by writing entries in three log files in the `/var/ct/`*cluster_name*`/log/cthags` directory. The log files are:

- `/var/ct/`*cluster_name*`/log/cthags_`*nodenum_instnum*`.`*cluster_name*
- `/var/ct/`*cluster_name*`/log/cthags_`*nodenum_instnum.cluster_name*`.long`
- `/var/ct/`*cluster_name*`/log/cthags.default.`*nodenum_instnum*

The log files contain the log of the `hagsd` daemons on the nodes.

The log file names include these variables:

- *nodenum* is the node number on which the daemon is running.
- *instnum* is the instance number of the daemon.
- *cluster_name* is the name of the cluster in which the daemon is running.

Each daemon limits the log size to a pre-established number of lines. The default is 5000 lines. When the limit is reached, the daemon appends the string `.bak` to the name of the current log file and begins a new log. If a `.bak` version already exists, it is removed before the current log is renamed.

Flags

-a [-p *port number*]
Adds the subsystem.

-s
Starts the subsystem.

-k
Stops the subsystem.

-d
Deletes the subsystem.

-t
Turns tracing on for the subsystem.

-o
Turns tracing off for the subsystem.

-r
Refreshes the subsystem.

-z
Uninstalls the `cthags` subsystem.

-h
Writes the command's usage statement to standard output.

Security

You must have `root` authority to run this command.

Exit Status

0
Indicates that the command completed successfully.

a non-zero value
Indicates that an error occurred.

Restrictions

This command is valid in a peer domain only.

Use this command *only* under the direction of the IBM Support Center.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output.

Standard Error

This command writes error messages, as necessary, to standard error.

Examples

1. To add the group services subsystems to the SRC in the current cluster, enter:

```
cthagsctrl -a
```

2. To add the group services subsystems with a port number of 12347, enter:

```
cthagsctrl -a -p 12347
```

3. To start the group services subsystems in the current cluster, enter:

```
cthagsctrl -s
```

4. To stop the group services subsystems in the current cluster, enter:

```
cthagsctrl -k
```

5. To delete the group services subsystems from the SRC in the current cluster, enter:

```
cthagsctrl -d
```

6. To turn tracing on for the group services daemon in the current cluster, enter:

```
cthagsctrl -t
```

7. To turn tracing off for the group services daemon in the current cluster, enter:

```
cthagsctrl -o
```

Location

/opt/rsct/bin/cthagsctrl
Contains the `cthagsctrl` command

cthagstune Command

Purpose

Changes the group services subsystem tunable parameters at run time.

Attention: Starting with RSCT 2.5.5.0, the **cthagstune** command is not supported for controlling the group services trace output. You can use trace spooling to control group services trace output. For more information, see Configuring trace spooling.

Syntax

`cthagstune` [`-l` *log_length*] [`-d` *log_dirsize*]

`cthagstune` [`-h`]

Description

The `cthagstune` command changes the group services subsystem tunable parameters at run time.

Flags

-l
Specifies the maximum log file length. If the value is `0` or a negative number, a default log file length is used.

-d
Specifies the maximum log directory size in kilobytes. If the value is `0` or a negative number, a default log directory size is used.

-h
Writes the command's usage statement to standard output.

Security

You must have `root` authority to run this command.

Exit Status

0
Indicates that the command completed successfully.

a non-zero value
Indicates that an error occurred.

Restrictions

This command is valid in a peer domain only.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output.

Standard Error

This command writes error messages, as necessary, to standard error.

Examples

To change the log file length to 6000 lines and to set the log directory size to approximately 7 megabytes, enter:

```
cthagstune -l 6000 -d 7000
```

Location

/opt/rsct/bin/cthagstune
Contains the `cthagstune` command

cthatsctrl Command

Purpose

Controls the topology services subsystem.

Syntax

`cthatsctrl` { `-a` [`-p` *port-number*] | `-s` | `-k` | `-d` | `-b` | `-t` | `-o` | `-r` | `-h` }

Description

The `cthatsctrl` control command controls the operation of the topology services subsystem. The subsystem is under the control of the system resource controller (SRC) and belongs to a subsystem group called `cthats`. Associated with each subsystem is a daemon and a command that configures and starts the daemon.

An instance of the topology services subsystem runs on every node of a cluster.

Adding the subsystem

When the `-a` flag is specified, the control command uses the `mkssys` command to add the topology services subsystem to the SRC. The control command:

1. Makes sure the `cthats` subsystem is stopped.
2. Gets the port number from the cluster data makes sure the port number is set in the `/etc/services` file.

   The service name that is entered in the `/etc/services` file is `cthats`.
3. Removes the `cthats` subsystem from the SRC (in case it is still there).
4. Adds the `cthats` subsystem to the SRC.

Starting the subsystem

When the `-s` flag is specified, the control command uses the `startsrc` command to start to start the topology services subsystem, `cthats`.

Stopping the subsystem

When the `-k` flag is specified, the control command uses the `stoptsrc` command to stop the topology services subsystem, `cthats`.

Deleting the subsystem

When the `-d` flag is specified, the control command uses the `rmssys` command to remove the topology services subsystem from the SRC. The control command:

1. Makes sure the `cthats` subsystem is stopped
2. Removes the `cthats` subsystem from the SRC using the `rmssys` command
3. Removes the `cthats` port number from the `/etc/services` file

Rebuilding the configuration

When the `-b` flag is specified, the control command reads the configuration information from the cluster data and builds a configuration file, `machines.lst`, for the topology services daemon.

Turning tracing on

When the `-t` flag is specified, the control command turns tracing on for the topology services daemon using the `traceson` command.

Turning tracing off

When the `-o` flag is specified, the control command turns tracing off (returns it to its default level) for the topology services daemon using the `tracesoff` command.

Refreshing the subsystem

When the `-r` flag is specified, the control command refreshes the subsystem using the `refresh` command. The `-r` flag signals the daemon to read the rebuilt information.

Flags

-a [-p *port-number*]
Adds the subsystem.

-s
Starts the subsystem.

-k
Stops the subsystem.

-d
Deletes the subsystem.

-t
Turns tracing on for the subsystem.

-o
Turns tracing off for the subsystem.

-b
Rebuilds the topology services configuration file from the configuration information in the cluster data.

-r
Refreshes the subsystem.

-h
Writes the command's usage statement to standard output.

Security

You must have `root` authority to run this command.

Exit Status

0
Indicates that the command completed successfully.

a non-zero value
Indicates that an error occurred.

Restrictions

This command is valid in a peer domain only.

Use this command *only* under the direction of the IBM Support Center.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output.

Standard Error

This command writes any error messages to standard error.

Examples

1. To add the topology services subsystem to the SRC, enter:

```
cthatsctrl -a
```

2. To start the topology services subsystem, enter:

```
cthatsctrl -s
```

3. To stop the topology services subsystem, enter:

```
cthatsctrl -k
```

4. To delete the topology services subsystem from the SRC, enter:

```
cthatsctrl -d
```

5. To turn tracing on for the topology services daemon, enter:

```
cthatsctrl -t
```

6. To turn tracing off for the topology services daemon, enter:

```
cthatsctrl -o
```

7. To rebuild the topology services configuration file from the configuration information in the cluster data, enter:

```
cthatsctrl -b
```

8. To signal all the topology services daemons in the cluster to read the new configuration file, enter:

```
cthatsctrl -r
```

9. To write usage information to standard output, enter:

```
cthatsctrl -h
```

Location

/opt/rsct/bin/cthatsctrl
Contains the `cthatsctrl` command.

cthatstune Command

Purpose

Views and changes the topology services subsystem's tunable parameters at run time.

Syntax

`cthatstune` [`-f` [*network1*]`:`*frequency1*[`,`[*network2*]`:`*frequency2*...]] [**-g** [[*network*]**:***grace*]] [`-s` [*network1*]`:`*sensitivity1*[`,`[*network2*]`:`*sensitivity2*...]] [`-p` *priority*] [`-l` *log_length*] [`-m` *pin_object*] [`-r`] [`-v`] [`-h`]

Description

The `cthatstune` command changes the topology services subsystem's tunable parameters at run time. The topology services subsystem has two types of tunable parameters:

subsystem-wide
Affects the behavior of the topology services subsystem. This type includes the fixed priority level, the maximum length of the log file, and the object to be pinned in main memory.

per-network
Affects the behavior of each network. This type includes the heartbeat frequency and sensitivity.

The `cthatstune` command changes the parameters in the cluster data. The new values will not take effect until the topology services daemon reads in the new values from the cluster data. You can use a refresh operation to instruct the topology services daemon to read the new values from the cluster data. You can start a refresh operation by issuing the `cthatsctrl -r` command or the `cthatstune -r` command on one of the nodes in the cluster.

In addition to the real values, two special values: `VIEW` and `DEFAULT`, can be used to display the current setting and to use the default value of the tunable parameter, respectively.

For per-network tunable parameters, in addition to the network name, an empty network name or the special network name `ALL` can be used to specify that the value following the network name applies to all networks.

Flags

-f [*network1*]:*frequency1*[,[*network2*]:*frequency2*...]
Specifies the *heartbeat frequency*, which is the interval in seconds between heartbeats, for one or more networks.

The value of *frequency* can be an integer from 1 to 30. The default value is 1.

-g [[*network*]:*grace*]
Specifies the grace period that is used when heartbeats are no longer received. When a heartbeat is missed, an Internet Control Message Protocol (ICMP) echo packet is sent to the failed node. If the echo is returned, the grace period is initiated.

The grace period is specified in seconds and is significant to milliseconds. It can be specified as an integer, a floating-point number, or one of these values:

0
Specifies that the grace period is disabled.

-1 | d
Specifies that the topology services subsystem controls the grace period. This is the default value.

-s [*network1*]:*sensitivity1*[,[*network2*]:*sensitivity2*...]
Specifies the maximum number of missing heartbeats for one or more networks. If this maximum is exceeded, the topology services daemon considers the peer to be inactive.

The value of *sensitivity* can be any integer from 4 to 40. The default value is 4.

-p *priority*
Specifies the fixed priority level. The value of *priority* can be 0, which means "do not run in fixed priority level," or an integer from 1 to 80. The default value is 30.

-l *log_length*
Specifies the maximum log file length (in number of lines). The value of *log_length* can be any integer from 2000 to 1 000 000. The default value is 5000.

-m *pin_object* [,*pin_object*...]
Specifies the object to be pinned in main memory. Valid values are:

NONE
Does not pin any object in main memory.

TEXT
Specifies the TEXT object to be pinned in main memory.

DATA
Specifies the DATA object to be pinned in main memory.

STACK
Specifies the STACK object to be pinned in main memory.

PROC
Specifies that all pinnable objects should be pinned in main memory. This is the default value.

-r
Applies the new tunables and refreshes the topology services subsystem.

-v
Provides verbose output.

-h
Writes the command's usage statement to standard output.

Security

You must have root authority to run this command.

Exit Status

0
Indicates that the command completed successfully.

a non-zero value
Indicates that an error occurred.

Restrictions

This command is valid in a peer domain only.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

This command writes any error messages to standard error.

Examples

1. To change the fixed priority level to 40, view the current setting of the maximum log file length, and pin default objects in main memory, without making the new setting take effect immediately, enter:

   ```
   cthatstune -p 40 -l VIEW -m DEFAULT
   ```

2. To make the new setting (previously changed by `cthatstune`) take effect, enter:

   ```
   cthatstune -r
   ```

3. To change the fixed priority level to normal, pin program and data segments in main memory, and make the new settings take effect immediately, enter:

   ```
   cthatstune -p 0 -m TEXT,DATA -r
   ```

4. To change the heartbeat frequency of `filesys_net` to 2 and all other networks to 4, change the sensitivity of all other networks to the default value, and make the new settings take effect immediately, enter:

   ```
   cthatstune -f filesys_net:2,:4 -s :DEFAULT -r
   ```

5. To change the heartbeat frequency of `filesys_net` to the default value and `service_net` to 3, change the sensitivity of all networks to 8, pin the entire topology services subsystem in main memory, and make the new settings take effect immediately, enter:

   ```
   cthatstune -f filesys_net:DEFAULT,service_net:3 -s :8 -m PROC -r
   ```

   You can also do this using the following method:

   ```
   cthatstune -f filesys_net:DEFAULT,service_net:3
   cthatstune -s :8
   cthatstune -m PROC
   cthatstune -r
   ```

6. To change the period for network communication group **CG3** to 2345 milliseconds, enter:

   ```
   cthatstune -f CG3:2.345
   ```

7. To change the grace period for network communication group **CG3** to 30500 milliseconds, enter:

   ```
   cthatstune -g CG3:30.5
   ```

Location

/opt/rsct/bin/cthatstune
Contains the `cthatstune` command

ctlvsd Command

Purpose

Sets the operational parameters for the virtual shared disk subsystem on a node.

Syntax

ctlvsd
[`-r` *node_number*... | `-R` | `-p` *parallelism* |

`-k` *node_number*... | `-t` | `-T` | `-v` *vsd_name* ... |

`-V` | `-C` | `-K` | `-M` *IP_max_message_size*]

Description

The `ctlvsd` command changes some parameters of the virtual shared disk subsystem. When called with no arguments, the command displays the current and maximum cache buffer count, the request block count, the pbuf count, the minimum buddy buffer size, the maximum buddy buffer size, and the overall size of the buddy buffer.

Sequence number information may or may not be displayed. In general, sequence numbers and the options that reset them are managed entirely within the virtual shared disk and recoverable virtual shared disk subsystems.

Flags

-r
Resets the outgoing and expected sequence numbers for the nodes specified on the node on which the command is run. Use this flag when another node has either been rebooted, cast out, or all virtual shared disks have been reconfigured on that node. The specified nodes are also cast in.

Note: This option should be used only under direct guidance from IBM Service. It should never be used under normal circumstances.

-R
Resets the outgoing and expected sequence number for all nodes on the node on which the command is run. Use this flag after rebooting the node. All nodes in the virtual shared disk network will be cast in.

Note: This option should be used only under direct guidance from IBM Service. It should never be used under normal circumstances.

-p
Sets the level of virtual shared disk parallelism to the number specified. The valid range is 1 to 9. The default is 9. A larger value can potentially give better response time to large requests. (See *RSCT for AIX 5L: Managing Shared Disks* for more information regarding tuning virtual shared disk performance.)

This value is the *buf_cnt* parameter on the `uphysio` call that the virtual shared disk IP device driver makes in the kernel. Use `statvsd` to display the current value on the node on which the command is run.

-k
Casts out the node numbers specified on the local node. The local node ignores requests from cast out nodes. Use `-r` to cast nodes back in.

Note:

1. Before using this flag, refer to the “Restrictions” section that follows.
2. This option should be used only under direct guidance from IBM Service. It should never be used under normal circumstances.

-t
Lists the current routing table and mbuf headers cached by the virtual shared disk driver.

-T
Clears or releases all cached routes.

-v *vsd_name ...*
Resets the statistics in the number of read and write requests on the specified virtual shared disks.

-V
Resets all the configured virtual shared disk's statistics in the number of read and write requests.

-C
Resets the virtual shared disk device driver counters displayed by the `statvsd` command. Exceptions are the outgoing and expected request sequence numbers among the client and server nodes.

-K
Casts out all nodes on the local node. Local requests are still honored.

Note:

1. Before using this flag, refer to the “Restrictions” section that follows.
2. This option should be used only under direct guidance from IBM Service. It should never be used under normal circumstances.

-M
Sets the virtual shared disk maximum IP message size. This is the largest sized block of data the virtual shared disk sends over the network for an I/O request. This limit also affects local virtual shared disk I/O block size. The value is in bytes and must not be greater than the maximum transmission unit (MTU) size of the network. All nodes should use the same value. The recommended values are:

- 61440 (60KB) for a switch
- 8192 (8KB) for jumbo frame Ethernet
- 1024 (1KB) for 1500-byte MTU Ethernet

Parameters

vsd_name
Specifies a defined virtual shared disk.

Security

You must have `root` authority to run this command.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, see *RSCT Administration Guide* .

Examples

1. To display the current parameters, enter:

```
ctlvsd
```

   The system displays a message similar to the following:

```
The minimum buddy buffer size is 4096.
The maximum buddy buffer size is 65536.
The total buddy buffer size is 4 max buffers, 262144 bytes.
```

2. To display the current IP routing table, enter:

```
ctlvsd -t
```

   The system displays the following information:

```
Route cache information:

 destination  interface  ref  status  direct/gateway   min managed mbuf
     1           ml0      2     Up        Direct            256
```

Location

/opt/rsct/vsd/bin/ctlvsd

ctmonfs command

Purpose

Monitors a file system for a specific condition.

Syntax

To add or monitor a new file system condition, use the following syntax:

```
ctmonfs -a -n condition_name -f <fsname> -c <condition_type[,attr=value,...]> [-b
<action_name> ]
[-k <action_type[,attr=value,...]> [-i <interval>] [-m 0|1] [-e 0|1]
```

To change attributes of a file system condition, use the following syntax:

```
ctmonfs -u -n condition_name [-f <fsname>] [-c <attr=value,...>] [-b|-B <action_name> ]
[-k <attr=value,...>] [-i <interval>] [-m 0|1] [-e 0|1]
```

To remove a file system condition, use the following syntax:

```
ctmonfs -r -F | -n condition_name
```

To list a specific file system condition or to list all file system conditions, use the following syntax:

```
ctmonfs -l [-n condition_name]
```

Description

The **ctmonfs** command is used to monitor a file system for a specific condition. *Conditions* define the criteria against which a file system must be monitored. *Actions* define the commands that must be run when the condition occurs. You do not need to specify any action or you can specify actions that must be run according to the priority of action. If the priority of the actions is not specified, the actions are run in the order that you specified the actions. If you do not specify any action, the condition is logged in trace

file without running any action. Each condition or action consists of attributes that define the additional data of the condition or action.

The following condition type is supported:

fsspace
Monitors free space of the file system. The `fsspace` condition type contains the following attributes:

method
Specifies how the file system space is monitored. Valid values are as follows:

free_percent
Monitors file system by percentage of total remaining space.

free_space
Monitors file system by units of remaining space in MB.

threshold_free_percent
Specifies a threshold percentage value for a condition of free or remaining space in a file system. That is, the condition becomes true if percentage of free space is equal to or lesser than this value.

Note: This attribute is valid only when the **method** attribute is set to `free_percent`.

threshold_free_space
Specifies a threshold value in MB for a free space condition in a file system. That is, the condition becomes true if the free space is equal to or lesser than this value.

Note: This attribute is valid only when the **method** attribute is set to `free_space`.

rearm_value
Stops evaluating the **threshold_free_percent** or **threshold_free_space** values when the condition becomes true and starts evaluating the **rearm_value** attribute value. The rearm condition becomes true if the `free_percent` or `free_space` value becomes equal to or greater than this value.

Note: The **rearm_value** attribute is optional. In addition, the **rearm_value** and **rearm_waittime** attributes are mutually exclusive. If the **rearm_value** attribute is specified, the **rearm_waitime** attribute must not be specified.

rearm_waittime
Stops evaluating the **threshold_free_percent** or **threshold_free_space** values when the condition becomes true and resumes the file system monitoring after the time that is specified in the **rearm_waittime** attribute. That is, file system monitoring is suspended for the duration that is specified in the **rearm_waittime** attribute.

Note: The **rearm_waittime** attribute is optional. In addition, the **rearm_value** and **rearm_waittime** attributes are mutually exclusive. If the **rearm_waittime** attribute is specified, the **rearm_value** attribute must not be specified.

The following action type is supported:

useraction
Starts the user script or an executable file. The `useraction` action type contains the following attributes:

path
Specifies the name of the executable file.

priority
Specifies the priority of the action. Valid values are positive integers. A lesser number indicates more priority. A value of 0 means no priority.

The configuration settings that are specified in the **ctmonfs** command are stored in the `/var/ct/cfg/fsmon.cfg` file. By default, the `/var` file system is monitored for the `fsspace` condition with an associated `useraction` type. When the `fsspace` condition becomes true, a default script is called to clean unnecessary files in the `/var/ct` directory.

Flags

-a
Creates a file system condition to monitor the file system.

-u
Modifies an existing file system condition.

-r
Removes a file system condition. If the **-F** flag is specified, all the file system conditions are removed.

-l
Lists all file system conditions. If you specify the **-n** flag, only a specific file system condition is listed.

-n *condition_name*
Specifies a unique name for a file system condition. The name can be used to list, modify, or remove the file system condition.

-f *fsname*
Specifies the name of the file system that you want to monitor.

-c *condition_type*
Specifies the condition type that must be monitored. The *condition_type* parameter consists of a condition type followed by comma-delimited attributes. To modify attribute values, use the **-u** option and specify the attributes that must be changed.

-b *action_name*
Specifies a unique name for an action. The action name can be used to modify or to remove the action.

-B *action_name*
Removes the specified action.

-k *action_type*
Specifies the action that must be performed when the condition becomes true. The *action_type* parameter consists of the action type followed by comma-delimited attributes. To modify attributes of the action type, use the **-u** option and specify the attributes that must be changed.

Note: You can define only one action when a new file system condition is created. Use the **-u** flag to add more actions to the condition. If no action is configured for a condition, the event is logged in trace files when the condition becomes true.

-i *interval*
Specifies the interval, in seconds, at which the condition is evaluated.

-m [0|1]
Enables or disables monitoring of the file system. The following values are valid:

1
Enables monitoring of the file system.

0
Disables monitoring of the file system.

-e [0|1]
Enables or disables error logging. The following values are valid:

1
An error log entry is created when the condition becomes true.

0
Error logging is disabled when the condition becomes true.

Exit status

0
The command completed successfully.

1
An error occurred.

Examples

1. To list all monitoring conditions for file system, enter the following command:

```
ctmonfs -l
```

2. To monitor the file system space (`fsspace`) and to run the `/tmp/mycleanup` command when the free space percentage is less than 10%, enter the following command:

```
ctmonfs -a -n "var mon" -f "/var" -c "fsspace,method=free_percent,threshold_free_percent=10"
-b "run cleanup" -k "useraction,path=/tmp/mycleanup" -i 1800 -m 1 -e 1
```

3. To modify the interval at which the condition is evaluated, enter the following command:

```
ctmonfs -u -n "var mon" -i 900
```

4. To modify the path attribute value of an existing action (`run cleanup`), enter the following command:

```
ctmonfs -u -n "var mon" -b "run cleanup" -k "path=/tmp/varcleanup"
```

5. To remove an action, enter the following command:

```
ctmonfs -u -n "var mon" -B "run cleanup"
```

6. To remove a file system condition, enter the following command:

```
ctmonfs -r -n "var mon"
```

7. To set or modify the priority of an existing action (`run cleanup`), enter the following command:

```
ctmonfs -u -n "var mon" -b "run cleanup" -k "priority=1"
```

Location

/opt/rsct/bin/ctmonfs
Contains the **ctmonfs** command.

Files

/var/ct/cfg/fsmon.cfg
Contains the configuration settings for monitoring a file system.

ctmsskf Command

Purpose

Displays and manages the contents of a message security services (MSS) key file.

Syntax

`ctmsskf {-a | -d | -l | -h} [-f` *key_file*`] [-t` *key_type*`] [-v` *key_version*`] [-k` *key_value*`]`

Description

The `ctmsskf` command displays and manages the contents of a message security services (MSS) typed key file. Use this command to add a key to, delete a key from, or list the contents of a key file.

Adding a key:

When you use this command to add a key entry to a key file, you must specify the following:

- the name of the key file where the key is to be added

- the type of the key to add
- optionally, the version of the key that is to be added to the key file
- the 16-digit value of the key

If the specified key file does not exist, it is created. If the specified key file *does* exist, the `ctmsskf` command verifies that the key type specified for the new key matches the type used by the keys already recorded within the file. Only keys of the same type can be added to an existing key file. When a key is successfully added to the file, that version of the key becomes the *active key version.* If a key version is specified using the `-v` *key_version* flag, *key_version* is used as the new version number and is made the active version. If *key_version* is not specified, the key is added using a key version value that is one greater than the previous active key version number.

Existing versions of a key cannot be replaced. To replace an existing version of a key or to change the value of an existing version of a key, that key version must first be deleted using the `-d` flag, and then added again using the `-a` flag. The command returns an error if you try to add a key that uses a version number already in use by a key within an existing key file. In general, key replacements should only be performed on the value of the key that is currently active, as replacing the value of an older key version makes the older key version active.

Because key versions can be added to the key file in any order, the highest key version number may or may not be the key version that is currently active. Use the `-l` flag to determine which key version is currently active for a file.

Deleting a key:

When you use this command to delete a key entry from a key file, you must specify the following:

- the name of the key file from where the key is to be deleted
- optionally, the type of key to delete
- optionally, the version of the key to delete

If the key specified is empty, does not exist, or does not have a proper header, the command returns an error. If the key type is specified and it does not match the key type in the header of the, the command returns an error. If the key version is specified, the command locates the record corresponding to the version provided and purges it from the file. If there is no such record, the command returns an error. If no key version is provided, the command purges only the records that are marked as inactive.

Listing the contents of a key file:

When you use this command to list the contents of a key file, the following information is displayed:

- the header of the key file.
- the list of keys in the key file.

  The following information is displayed for each key:

  – an indication of whether the record is inactive
  – the version of the key
  – the type of the key
  – the 16-digit value of the key

Flags

-a
Adds a key to the key file. The `-f`, `-k`, and `-t` flags must also be specified.

-d
Deletes a key from the key file. The `-f` and `-v` flags must also be specified. If the `-t` flag is specified, the command checks to see if the type of the key file is the same as the key type provided.

-l
Lists the contents of the key file. The `-f` flag must also be specified. If the `-v` flag is specified, the command lists only the key that matches the version number provided.

-f *key_file*
Specifies the name of the key file. The key file must be a valid key file created by MSS API or by this command.

-t *key_type*
Specifies the type of the key to add. If the specified key file is not empty, the command checks to see if the key type specified matches the key type in the header of the key file. The valid key type values are: `3des_md5`, `aes256_md5`, `des_cbc`, `des_md5`, `rsa512_sha`, and `rsa1024_sha`.

-v *key_version*
Specifies the version of the key.

-k *key_value*
Specifies the 16-digit value of the key.

-h
Writes the command's usage statement to standard output.

Security

The file system permission of the key files is determined by the application owning the file. If the invoker doesn't have sufficient authority to open the file, the command fails.

Exit Status

0
The command completed successfully.

4
The caller invoked this command incorrectly, omitting required flags and parameters, or using mutually-exclusive flags. This command terminated without processing the request.

6
A memory allocation request failed during the operation of this command. The command was unable to complete the requested action.

9
If the `-a` flag was specified, the command detected a key within the key file that used the same version number as the one specified by the `-v` flag. If the `-d` flag was specified, the command was unable to locate a key in the key file using the version number specified by the `-v` flag. The key file was not modified.

21
The key file could not be located. Verify that the path name for the key file specified by the `-f` flag is correct.

27
The key type specified by the `-t` flag does not match the type for keys stored in the file specified by the `-f` flag. The requested action was not performed.

30
`ctmsskf` was unable to obtain exclusive use of the key file. Another instance of this command may be running and attempting to modify the same file, or the process that makes use of this key file may be examining the file. Retry the command at a later time.

36
The command user does not have sufficient permission to modify the contents of the key file.

37
The key file appears to be corrupted. Try to list the contents of the file using the `-l` flag to verify if the file is corrupted. Follow the problem resolution advice listed in the error message for further recovery action.

Restrictions

This command works only on MSS-formatted key files.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. When the `-i` flag is specified, the list of available key generation methods is displayed. When the `-l` flag is specified, one or more keys from the key file are displayed.

Standard Error

Descriptive information for any detected failure condition is written to standard error.

Examples

1. To view the keys contained in the key file `/my_key_file`, enter:

```
ctmsskf -l -f /my_key_file
```

2. To view the key with version 9 from the key file `/my_key_file`, enter:

```
ctmsskf -l -v 9 -f /my_key_file
```

3. To add a key to the key file `/my_key_file`, enter:

```
ctmsskf -a -t des_cbc -f /my_key_file -k 16_digit_value
```

4. To delete a key from the key file `/my_key_file`, enter:

```
ctmsskf -d -f /my_key_file -v 10
```

5. To delete all inactive keys in the key file `/my_key_file`, enter:

```
ctmsskf -d -f /my_key_file
```

Location

/opt/rsct/bin/ctmsskf
Contains the `ctmsskf` command

Files

/opt/rsct/cfg/ctcasd.cfg
Default configuration for the `ctcasd` daemon

/var/ct/cfg/ctcasd.cfg
Configuration for the `ctcasd` daemon, which can be modified by the system administrator

/var/ct/cfg/ct_has.pkf
Default location of the cluster security services public key file for the node

/var/ct/cfg/ct_has.qkf
Default location of the cluster security services private key file for the node

/var/ct/cfg/ct_has.thl
Default location of the cluster security services trusted host list for the node

ctscachgen Command

Purpose

Creates or replaces an on-disk version of a key cache.

Syntax

`ctscachgen -c` *file-name* [`-f`] [`-i` | `-n` *enc-key-name* | `-k` *enc-key-value* `-t` *key-type* | `-q`] [`-m` *key-gen-method*] [`-s` *cache-size*] [`-h`]

Description

The `ctscachgen` command generates a key cache and stores the completed cache to an on-disk file named in *file-name*. This file can later be used and updated by applications through the `libct_skc` library interfaces.

Flags allow you to specify the type of key to be generated, using the mnemonics that are used for symmetric key types by the `ctmsskf` command. You can also specify a key value to be used to encrypt the keys available in this cache. The keys are not encrypted by default. In addition, you can specify the number of keys to be stored in the file.

If the file specified in *file-name* exists, it is overwritten, even if the current contents do not match the flags specified on the command line.

Flags

-c *file-name*
Specifies the name of the key cache file. It can be either the full path or the relative path to the current directory.

-f
Instructs the command to overwrite an existing key cache file with the same name without asking the invoker to confirm its overwriting.

-i
Displays information about the key cache file specified with the `-c` flag. The information displayed contains the version of the cache file, the read count, the number of keys in the cache, the type of keys in the cache, and whether they are encrypted with a pre-encryption key. This flag cannot be used in conjunction with the `-n`, `-k`, `-t`, or `-q` flag.

-n *enc-key-name*
Provides the name of the file that contains the encryption typed key. This flag cannot be used in conjunction with the `-i`, `-k`, `-t`, or `-q` flag.

-k *enc-key-value*
Specifies the key value, expressed in hexadecimal form (`6fe45d20a`, for example), to be used as the pre-encryption key. By default, no pre-encryption key value is used. This flag must be used with the `-t` flag. It cannot be used in conjunction with the `-i`, `-n`, or `-q` flag.

-t *key-type*
Provides the type of the encryption key specified by the `-k` option. The valid key types are: `3des_md5`, `aes256_md5`, `des_cbc`, `des_md5`, `rsa512_sha`, and `rsa1024_sha`. This flag must be used with the `-k` flag. It cannot be used in conjunction with the `-i`, `-n`, or `-q` flag.

-q
Instructs the command to use the host's HBA private key as encryption key used for pre-encrypting the session keys in the on-disk key cache file. This flag cannot be used in conjunction with the `-i`, `-k`, `-t`, or `-n` flag.

-m *key-gen-method*
Provides the session key generation method. Valid values are: `3des_md5`, `aes256_md5`, and `des_md5`. If you do not specify this flag, the default method for generating the session keys is `des_md5`.

-s *cache-size*
Provides the size of the on-disk key cache file in terms of number of keys in the cache. If you do not specify this flag, the default cache size is 128 keys.

-h
Writes the command's usage statement to standard output.

Security

Permissions on the `ctscachgen` command permit only `root` to run the command.

Exit Status

Upon successful completion, the command returns an exit status code of `0` and generates an on-disk key cache file. In the event of a failure, the routine returns the error code and may remove the existing key cache file that the invoker wants to overwrite.

0
The command completed successfully.

4
Flags are mismatched or not valid. *file-name* remains unmodified.

6
A memory allocation request failed during the operation of this command. The command was unable to complete the requested action.

12
The command user cannot remove the existing key cache file (*file-name* remains unmodified) or access or write to the directory where *file-name* resides.

21
There is not enough space to store *file-name* or the *file-name* contents appear corrupt.

27
The key stored in the file specified by the `-c` flag is not valid or is corrupted. *file-name* remains unmodified.

36
The invoker cannot access the file specified by the `-c` flag. *file-name* remains unmodified.

Restrictions

- On-disk key caches are intended to be used solely upon the system on which they were generated. They are not intended to be shared between systems or migrated to another system. If multiple systems access the same key cache file, the protections offered by these keys is lost, because multiple systems and applications have access to information that is supposed to remain secret to a specific application. Therefore, any files created by this command should not be stored in shared file systems or networked file systems.
- Files generated by this command are generated in a host-ordered binary format. This format makes it impossible for a key cache file generated on one architecture (such as a Power platform) to be used on a different architecture (such as an Intel platform).

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. When the `-i` flag is specified, information about the key cache file is written to standard output.

Standard Error

Descriptive information for any detected failure condition is written to standard error.

Examples

1. To view the keys contained in the key file /my_key_file, enter:

```
ctmsskf -l -f /my_key_file
```

2. To view the key with version 9 from the key file /my_key_file, enter:

```
ctmsskf -l -v 9 -f /my_key_file
```

3. To add a key to the key file /my_key_file, enter:

```
ctmsskf -a -t des_cbc -f /my_key_file -k 16_digit_value
```

4. To delete a key from the key file /my_key_file, enter:

```
ctmsskf -d -f /my_key_file -v 10
```

5. To delete all inactive keys in the key file /my_key_file, enter:

```
ctmsskf -d -f /my_key_file
```

Location

/opt/rsct/bin/ctscachgen
Contains the ctscachgen command

Files

/opt/rsct/cfg/ctcasd.cfg
Default configuration for the ctcasd daemon

/var/ct/cfg/ctcasd.cfg
Configuration for the ctcasd daemon, which can be modified by the system administrator

/var/ct/cfg/ct_has.pkf
Default location of the cluster security services public key file for the node

/var/ct/cfg/ct_has.qkf
Default location of the cluster security services private key file for the node

/var/ct/cfg/ct_has.thl
Default location of the cluster security services trusted host list for the node

ctscfg Command

Purpose

Lists and modifies the contents of the cluster security services configuration file.

Syntax (ctscfg -a)

ctscfg -a { -c *MPM_code* } { -n *MPM_name* } { -o *MPM_object_module* } { -p *MPM_priority* } [-f i | u | z] [-l] [-h]

Syntax (ctscfg -d)

ctscfg -d { -c *MPM_code* | -n *MPM_name* } [-l] [-h]

ctscfg -u { { -c *MPM_code* } | { -n *MPM_name* } } { { -f i | u | z } | { -p *MPM_priority* } } [-l] [-h]

ctscfg -l

```
ctscfg -h
```

Description

The `ctscfg` command lists and modifies the contents of the cluster security services configuration file, `ctsec.cfg`. This file provides configuration information about the authentication methods that cluster security services can use for client-server authentication. Each authentication method is handled by a mechanism pluggable module (MPM). Each MPM configuration is defined by a one-line entry in the `ctsec.cfg` file. The entry contains information about:

- the priority of the MPM when cluster security services choose the authentication method for the client-server authentication
- the numeric code of the MPM, which is unique among all of the MPMs in the configuration file
- the mnemonic of the MPM, which is unique among all of the MPMs in the configuration file
- the name of the binary module that implements the functions of the MPM
- miscellaneous flags used by cluster security services mechanism abstract layer (MAL) when handling the MPM

Cluster security services include a default `ctsec.cfg` file in the `/opt/rsct/cfg/` directory. The `ctscfg` command does not modify this default configuration file. Instead, `ctscfg` makes a copy (if one does not exist already) of the default `ctsec.cfg` file and copies it to the `/var/ct/cfg/` directory. If a working copy of this file does exist already and there is enough space, the previous version is recorded to `/var/ct/cfg/ctsec.cfg.bak`.

Using this command, system administrators can create an "empty" security subsystem configuration, where no security MPMs are configured. In this configuration, all parties are to be considered not authentic.

Flags

-a
Adds a new configuration entry for a new MPM to the working copy of the `ctsec.cfg` file in the `/var/ct/cfg/` directory. If there is no working copy in that directory, `ctscfg` creates a working copy and modifies it. A configuration entry must include the MPM priority, numeric code, mnemonic, binary object, and, optionally, any flags. This flag requires the `-c`, `-n`, `-o`, and `-p` flags.

-c *MPM_code*
Specifies the code to be used by the security subsystem to refer to this MPM. *MPM_code* must be expressed as a hexadecimal value in the form of `"0x`*value*`"` (`"0x1a"` or `"0x9F"`, for example). This flag is required by the `-a` and `-d` flags.

-d
Removes an existing entry for a security MPM from the working copy of the `ctsec.cfg` file in `/var/ct/cfg`. If there is no working copy in that directory, `ctscfg` creates a working copy and modifies it. The `-c` flag or the `-n` flag must be specified to indicate which entry is to be removed.

-f i | **u** | **z**
Specifies the flags required by the security subsystem when adding an MPM to the configuration file. This option is required by the `-a` flag if the MPM has any miscellaneous flags or by the `-u` flag if the invoker intends to update the MPM flags. The MAL supports these miscellaneous flags:

i
Instructs MAL to initialize the MPM upon loading it in the virtual memory of the process.

u
Instructs MAL that it is safe to unload the MPM when it is no longer required.

z
Specifies the authorization method used for that MPM. An MPM with the same mnemonic as the authorization method must also exist and be configured in `ctsec.cfg`.

The flags must be specified with no space between them (`-f iuz`, for example).

-l
Lists the contents of the working `ctsec.cfg` file. If this option is specified with `-a`, `-d`, or `-u`, the resulting configuration is listed.

-n *MPM_name*
Specifies the mnemonic to be used for the security MPM. The mnemonic must be a short string value (`mymech`, for example). This flag is required by the `-a` and `-d` flags.

-o *MPM_object_module*
Specifies the location of the MPM, including the full path subdirectory. The MPM must exist as a file. If a symbolic link is used, the symbolic link must reference an existing file. The path must be expressed as an absolute path (`/usr/lib/mymech`, for example). This flag is required by the `-a` flag.

-p *MPM_priority*
Specifies the priority associated with this security mechanism pluggable module (MPM). Lower values have a higher priority. Priority values do not need to be consecutive, but no two MPMs can share priority. Negative values and a zero value are not permitted for a priority. This option is required by the `-a` flag and the `-u` flag if the invoker intends to update the MPM priority.

-u
Updates an existing configuration entry of an MPM in the working copy of the `ctsec.cfg` file in `/var/ct/cfg`. If there is no working copy in that directory, `ctscfg` creates a working copy and modifies it. The configuration entry must be specified by either the MPM numeric code or mnemonic. The only fields that can be updated are the MPM priority and flags. This flag requires the `-c` flag or the `-n` flag (in order to identify the configuration entry to modify) and `-f` flag or the `-p` flag (to specify the new values used for updating the selected configuration entry).

-h
Writes the command usage statement to standard output.

Standard output

When the `-h` flag is specified, this command usage statement is written to standard output.

Standard error

Descriptive information for any detected failure condition is written to standard error.

Exit status

0
The command completed successfully.

4
Flag error. One or more of the flags provided is not valid or is missing a value.

21
Configuration error. The MAL configuration file content is not valid or is corrupted.

30
Lock error. An error occurred during the locking of the MAL configuration file.

36
Permission error. The invoker does not have permission to list or modify the MAL configuration file.

105
File error. An error occurred during the reading or writing of the MAL configuration file.

Files

/var/ct/cfg/ctsec.cfg
Working copy of the MAL configuration file

/var/ct/cfg/ctsec.cfg.bak
Backup of the working copy of the MAL configuration file

Security

This command lists and modifies the MAL configuration file. The default version of the MAL configuration file that is installed by RSCT is protected using the file system permission bit mask of 444 (that is, read-only for everybody). Administrators who create a working copy of this file must preserve the permission bit mask in order to maintain the security of the system.

This command uses the working copy of the MAL configuration file in `/var/ct/cfg/`. If there is no such working copy, the command creates a file with the same ownership and permission bit mask as the default configuration file. If the invoker of the command has no permission to do that, the command returns a permission error.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the `rsct.core.sec` fileset for AIX.

Location

/opt/rsct/bin/ctscfg

Examples

1. To list the contents of the working copy of the `ctsec.cfg` file, either in `/opt/rsct/cfg/` or in `/var/ct/cfg/`, enter:

   ```
   /opt/rsct/bin/ctscfg -l
   ```

2. To add the HBA2 MPM to the working copy of the `ctsec.cfg` file in `/var/ct/cfg/`, enter:

   ```
   /opt/rsct/bin/ctscfg -a -n hba2 -p 2 -c 0x2 -o /opt/rsct/lib/hba2.mpm -f i
   ```

   This adds the following record to the working copy of the `ctsec.cfg` file in `/var/ct/cfg/`:

   ```
   1      hba2        0x00002     /usr/lib/hba2.mpm       i
   ```

3. To delete the UNIX MPM from the working copy of the ctsec.cfg file in /var/ct/cfg/, enter:

   ```
   /opt/rsct/bin/ctscfg -d -n unix
   ```

4. To update the HBA2 MPM with the UNIX MPM as the new authorization method in the working copy of the `ctsec.cfg` file in `/var/ct/cfg/`, enter:

   ```
   /opt/rsct/bin/ctscfg -u -n hba2 -f iz [unix]
   ```

5. To update the priority of the HBA2 MPM to a value of 2 in the working copy of the `ctsec.cfg` file in `/var/ct/cfg/`, enter:

   ```
   /opt/rsct/bin/ctscfg -u -n hba2 -p 2
   ```

ctsidmck Command

Purpose

Verifies the cluster security library identity mapping.

Syntax

`ctsidmck -h | -i | { [ -dl | -dm | -dh ] -m` *security_mechanism network_ID* `}`

Description

A system administrator can use the `ctsidmck` command to verify the mapping that would be obtained by the cluster security library (`libct_sec`) for a specific security network identifier.

The cluster security library establishes a security context through the exchange between a client of a trusted service and the trusted service server. During the creation of the security context, the cluster security library tries to map the client application's security network identity to an identity that may be present on the server node, called the *mapped identity.* The cluster security library uses the mapped identity later on the server in authorization functions such as access control verification. Whether the client application has a mapped identity on the server depends on whether the following identity mapping definition files are present on the server, and whether any of the entries within these files correspond to the security identity being used by the client application:

- `/opt/rsct/cfg/ctsec_map.global`
- `/var/ct/cfg/ctsec_map.local`
- `/var/ct/cfg/ctsec_map.global`

The location of definitions within these files is important; entries at the head of the file are processed before entries positioned towards the end of the file. The definition rules also allow for wildcarding of entry information and for expansion of certain reserved words. If a definition is incorrectly specified within one of these files, the mapping result may not be as intended. Also, if a definition is positioned after another definition that can successfully map a security network identifier, the mapping result may not be as intended.

This command allows an administrator to verify that the correct identity mapping definition is used by the cluster security library to map a security network identity. This command is to be executed on the system that would act as the server. By specifying a security network identifier to this command on the server, the administrator can determine what the mapped identity for that security network identity would be on that system, and what entry was used from the identity mapping definition files to obtain this mapping.

Flags

-h
Writes the command's usage statement to standard output.

-i
Displays a list of the supported security mechanisms on this system. The command examines the cluster security library configuration on this node, obtains a list of supported security mechanisms, and displays this list. The mechanisms are listed by the mnemonic used by the cluster security library to refer to these mechanisms.

-d
Specifies the level of detail in the command output. One of three levels of detail is permitted:

1. low (`l`): the command will only display the mapped identity for *network_ID*. This is the default detail level.
2. medium (`m`): the command will display the mapped identity for *network_ID*, as well as the entry from the identity mapping definition files that yielded the map.
3. high (h): the command will display every entry from the identity mapping definition files that is processed until a mapped identity for *network_ID* is found, or until all entries are processed.

-m *security_mechanism*
Specifies the security mechanism that was used to create the security network identifier provided by *network_ID*. *security_mechanism* is a mnemonic that would be used by the cluster security library to refer to this security mechanism. This flag must be specified when the `-h` and the `-i` flags are not provided.

Use the `-i` flag to display a list of the security mechanisms that this system supports.

Parameters

network_ID
Specifies the security network identifier to be mapped. This should be an identity that can be assumed by a client application of a trusted service.

Security

This command is executable only by the root system user and members of the system user group. It is intended for administrator use only, to verify the security configuration of the system. Because the output of the command could be used as a means for determining how to sabotage or circumvent system security, the permissions on this command should not be altered.

Exit Status

0
This command successfully found a mapped identity for *network_ID*.

3
This command detected a failure in the operation of the cluster security library mechanism pluggable module (MPM) corresponding to the security mechanism that was requested. `ctsidmck` was unable to search for a possible mapped identity for *network_ID* in this case. This failure may be accompanied by descriptive output indicating the nature of the MPM failure. Consult this output and perform any recommended actions.

4
The caller invoked this command incorrectly, omitting required flags and parameters, or using mutually-exclusive flags. `ctsidmck` terminated without trying to find a mapped identity for *network_ID*.

6
A memory allocation request failed during the operation of this command. `ctsidmck` was unable to search for a possible mapped identity for *network_ID* in this case.

21
This command was unable to locate any of the identity mapping definition files on the local system. `ctsidmck` was unable to search for a possible mapped identity for *network_ID* in this case. Verify that at least one identity mapping definition file exists on the system.

22
This command was unable to dynamically load the cluster security library mechanism pluggable module (MPM) corresponding to the security mechanism what was requested. The module may be missing, corrupted, or one of the shared libraries used by this module may be missing or corrupted. `ctsidmck` was unable to search for a possible mapped identity for *network_ID* in this case. This failure may be accompanied by descriptive output indicating the nature of the MPM failure. Consult this output and perform any recommended actions.

37
At least one of the identity mapping definition files on the system appears to be corrupted. The command was unable to search for a possible mapped identity for *network_ID* in this case. Verify that none of the identity mapping files are corrupted, truncated, or contain syntax errors.

38
The **ctsidmck** command cannott locate a mapped identity for *network_ID*. No entry within any of the identity mapping definition files yielded a mapped identity for the specified security network identifier.

Restrictions

This command works only on MSS-formatted key files.

Standard Output

The `ctsidmck` command writes any mapped identity found for the security network identifier to standard output. If a medium or high level of detail is requested, any definitions displayed by this command are also written to standard output.

When the `-h` flag is specified, this command's usage statement is written to standard output.

Standard Error

Descriptive information for any detected failure condition is written to standard error.

Examples

1. To get a list of the security mechanisms that the local system supports, before verifying an identity map, enter:

   ```
   ctsidmck -i
   ```

2. To get only the mapped identity for the RSCT host-based authentication (HBA) mechanism security network identity `zathras@greatmachine.epsilon3.org`, enter:

   ```
   ctsidmck -m unix zathras@greatmachine.epsilon3.org
   ```

3. To see every identity mapping definition that the command checks while searching for a mapped identity for the HBA mechanism's security network identity `glorfindel@rivendell.elvin.net@endor`, enter:

   ```
   ctsidmck -d h -m unix glorfindel@rivendell.elvin.net@endor
   ```

Location

/opt/rsct/bin/ctsidmck
Contains the `ctsidmck` command

Files

/opt/rsct/cfg/ctsec_map.global
The default identity mapping definition file. This file contains definitions required by the RSCT cluster trusted services in order for these systems to execute properly immediately after software installation. This file is ignored if the cluster-wide identity mapping definition file `/var/ct/cfg/ctsec_map.global` exists on the system. Therefore, any definitions within this file should also be included in the cluster-wide identity mapping definition file, if that file exists.

/var/ct/cfg/ctsec_map.local
Local override to the cluster-wide identity mapping definitions. Definitions within this file are not expected to be shared between nodes within the cluster.

/var/ct/cfg/ctsec_map.global
Cluster-wide identity mapping definitions. This file is expected to contain identity mapping definitions that are common throughout the cluster. If this file exists on the system, the default identity mapping definition file is ignored. Therefore, if this file exists, it should also contain any entries that would also be found in the default identity mapping definition file.

ctskeygen Command

Purpose

Generates cluster security services private and public keys for the local system and stores these keys in locally-mounted files.

Syntax

ctskeygen -n [-f] [-m *method*] [-p *public-file*] [-q *private-file*] | -d | -i | -h

Description

The ctskeygen command generates host identifier keys — a private key and public key pair — to be used by the cluster security services library (libct_sec) in RSCT host-based authentication (HBA). The command creates a new private key for the node, derives a public key from the new private key, and stores these keys to files on the local node.

Whenever the node's private and public keys are modified, the node's new public key must be distributed to all nodes within the cluster and placed in the trusted host list files on these nodes, replacing the previous value stored there for this node. If this is not done, the node that has generated new private and public keys will be unable to authenticate with other nodes in the cluster using HBA authentication.

Flags

-n
Generates host identifier keys (private and public keys).

-f
Forces ctskeygen to record the keys it generates to the private and public key files if these files already exist. By default, the command will not overwrite these files if they exist, because the presence of the files indicates that the cluster security services service may be active. Removing or modifying these files without informing other nodes of the change in the public key value will cause failures in HBA authentications on this node. This flag is not valid with the -h or the -i flag.

-m *method*
Instructs the command to use the specified key generation method in creating the host identifier keys. Valid parameters for this flag can be displayed using the -i flag. This flag is not valid with the -h and -i flags.

-p *public-file*
Specified the fully-qualified path name of the file to be used to store the local host's public key. If this file exists, the command will not overwrite the contents of this file unless the -f flag is also specified. If the -p flag is not specified, the command records this key to the /var/ct/cfg/ct_has.pkf file. This flag is not valid with the -h and -i flags.

-q private-file
Specified the fully qualified path name of the file to be used to store the private key of the local host. If this file exists, the command will not overwrite the contents of this file unless the -f flag is also specified. If the -q option is not specified, the command records this key to the file /var/ct/cfg/ct_has.qkf. This flag is not valid with the -h and -i flags.

-d
Displays the current public key value for the local system.

-i
Displays information about the key generation methods supported by this version of the command. ctskeygen displays messages to indicate which values are currently supported as arguments to the -m flag, and what the command will use as a default setting for the -m flag.

-h
Writes the command's usage statement to standard output.

Parameters

network_ID
Specifies the security network identifier to be mapped. This should be an identity that can be assumed by a client application of a trusted service.

Security

Permissions on the `ctskeygen` command permit only `root` to run the command.

Exit Status

0
The command completed successfully.

4
The caller invoked this command incorrectly, omitting required flags and parameters, or using mutually-exclusive flags. This command terminated without processing the request.

6
A memory allocation request failed during the operation of this command. The command was unable to complete the requested action.

12
The command user does not have sufficient permission to view or modify the contents of the key file.

21
The key file could not be located or could not be created.

30
`ctskeygen` was unable to obtain exclusive use of the public or private key file. Another instance of this command may be running and attempting to modify the keys, or the `ctcasd` daemon may be examining these files. Retry the command at a later time.

37
The public or private key file appears to be corrupted. Try to view the public key value using the `-d` flag to verify if the file is corrupted. Follow the problem resolution advice listed in the error message for further recovery action.

Restrictions

- Cluster security services supports its own file formats, private key formats, and public key formats only.
- Trusted host lists are modifiable using the `ctsthl` command only.
- Cluster security services does not provide an automated utility for creating, managing, and maintaining trusted host lists throughout the cluster. This is a procedure left to either the system administrator or the cluster management software.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. When the `-d` flag is specified, the public key value stored in the public key file is written to standard output.

Standard Error

Descriptive information for any detected failure condition is written to standard error.

Examples

1. To obtain the list of supported key generation methods:

   ```
   ctskeygen -i
   ```

2. To create new host identifier keys for the local system using the default settings:

   ```
   ctskeygen -n
   ```

3. To create new host identifier keys for the local system using 512-bit RSA private keys, storing these keys in locations other than the default location:

```
ctskeygen -n -m rsa512 -p /mysec/public -q /mysec/private
```

Location

/opt/rsct/bin/ctskeygen
Contains the `ctskeygen` command

Files

/opt/rsct/cfg/ctsec_map.global
The default identity mapping definition file. This file contains definitions required by the RSCT cluster trusted services in order for these systems to execute properly immediately after software installation. This file is ignored if the cluster-wide identity mapping definition file `/var/ct/cfg/ctsec_map.global` exists on the system. Therefore, any definitions within this file should also be included in the cluster-wide identity mapping definition file, if that file exists.

/var/ct/cfg/ctsec_map.local
Local override to the cluster-wide identity mapping definitions. Definitions within this file are not expected to be shared between nodes within the cluster.

/var/ct/cfg/ctsec_map.global
Cluster-wide identity mapping definitions. This file is expected to contain identity mapping definitions that are common throughout the cluster. If this file exists on the system, the default identity mapping definition file is ignored. Therefore, if this file exists, it should also contain any entries that would also be found in the default identity mapping definition file.

ctsnap Command

Purpose

Gathers configuration, log, and trace information about the Reliable Scalable Cluster Technology (RSCT) components.

Syntax

ctsnap [**-a**] [**-c** *cluster_name_pattern*] [**-C** *cluster_ID_pattern*] [**-d** *output_dir*] [**-D** *daemon_name_pattern*] [**-k** *stackdump_default*] [**-n** *node_name_pattern*] [**-N** *node_ID_pattern*] [**-p** *days* | { **-f** *from_date* **-t** *to_date* }] [**-s** *spool_dir*] [**-S** *size*] [**-x** *runrpttr*] [**-h**] [**-z**]

Description

The **ctsnap** command gathers configuration, log, and trace information about the RSCT components that are installed with AIX or PowerHA. This command collects data only for the local node on which it is running. Depending on the programs that are installed, information about the following components may be included:

- Audit log resource manager (`IBM.AuditRM`)
- Cluster security services (`ctsec`)
- Common information model resource manager (`IBM.CIMRM`)
- Configuration resource manager (`IBM.ConfigRM`)
- Event management (`ha_em`)
- Event response resource manager (`IBM.ERRM`)
- File system resource manager (`IBM.FSRM`)
- First failure data capture (`ct_ffdc`)
- Group services (`cthags`)

- Host resource manager (`IBM.HostRM`)
- Least-privilege resource manager (`IBM.LPRM`)
- Low-level application programming interface (`lapi`)
- Management domain resource manager (**IBM.MgmtDomainRM**)
- Microsensor resource manager (**IBM.MicroSensorRM**)
- Recovery resource manager (**IBM.RecoveryRM**)
- Resource monitoring and control (`ctrmc`)
- Sensor resource manager (`IBM.SensorRM`)
- Storage resource manager (**IBM.StorageRM**)
- Topology services (`cthats`)
- Virtual shared disk (`vsd`) (on AIX 6.1)
- Recoverable virtual shared disk (`rvsd`) (on AIX 6.1)

If a problem occurs with any of these components, you can run this command in order to provide information to your software service organization.

The output of the **ctsnap** command consists of a compressed tar file (**ctsnap.***node_name***.***nnnnnnnn***.tar.Z**) and a log file (**ctsnap.***node_name***.***nnnnnnnn***.log**), where *node_name* is the name of the node on which **ctsnap** was run, and *nnnnnnnn* is the time stamp of when the **ctsnap** command was run. Provide both of these files to your software service organization. By default, **ctsnap** puts these files in the **/tmp/ctsupt** directory. Use the **-d** flag to specify a different output directory.

When needed, you can use **ctsnap** to collect information about spooled trace files. Use the **-c**, **-C**, **-D**, **-f**, **-n**, **-N**, **-p**, **-s**, **-S**, and **-t** flags to capture a subset of trace information. You can use the **ctsnap -k stackdump_default** command to produce a stack dump for the following RSCT subsystems:

- Audit log resource manager (**IBM.AuditRM**)
- Common information model resource manager (**IBM.CIMRM**)
- Configuration resource manager (**IBM.ConfigRM**)
- Event response resource manager (**IBM.ERRM**)
- File system resource manager (**IBM.FSRM**)
- Generic resource manager (**IBM.GblResRM**)
- Group services (**cthags**)
- Least-privilege resource manager (**IBM.LPRM**)
- Microsensor resource manager (**IBM.MicroSensorRM**)
- Recovery resource manager (**IBM.RecoveryRM**)
- Resource monitoring and control (**ctrmc**)
- Sensor resource manager (**IBM.SensorRM**)
- Storage resource manager (**IBM.StorageRM**)
- Topology services (**cthats**)

To format the trace file contents of all of the RSCT resource managers, use the **-x** flag.

You can also use the **ctsnap** command to obtain the trace and logging root directory from the RSCT File configuration file (**ctfile.cfg**).

Flags

-a
Collects information pertinent only to High Availability Cluster Multi-Processing (HACMP) clusters on the Linux operating system.

-c *cluster_name_pattern*
Specifies a selection pattern that will limit trace collection to certain cluster names. The pattern is interpreted as a Perl-language regular expression.

-C *cluster_ID_pattern*
Specifies a selection pattern that will limit trace collection to certain cluster IDs. The pattern is interpreted as a Perl-language regular expression.

-d *output_dir*
Specifies the output directory. The default directory is **/tmp/ctsupt**.

-D *daemon_name_pattern*
Specifies a selection pattern that will limit trace collection to certain daemons. The pattern is interpreted as a Perl-language regular expression.

-f *from_date*
Specifies the date from which you want to collect information. The format of the *from_date* parameter is:

```
yyyy-mm-dd[.hh[:mm[:ss]]]
```

Note: Use **-f** in conjunction with the **-t** flag.

-k stackdump_default
Produces a stack dump for these RSCT subsystems: **cthags**, **cthats**, **ctrmc**, **IBM.AuditRM**, **IBM.CIMRM**, **IBM.ConfigRM**, **IBM.ERRM**, **IBM.FSRM**, **IBM.GblResRM**, **IBM.LPRM**, **IBM.MicroSensorRM**, **IBM.RecoveryRM**, **IBM.SensorRM**, and **IBM.StorageRM**.

-n *node_name_pattern*
Specifies a selection pattern that limits the trace collection to certain node names. The pattern is interpreted as a Perl-language regular expression.

-N *node_ID_pattern*
Specifies a selection pattern that limits the trace collection to certain node IDs. The pattern is interpreted as a Perl-language regular expression.

-p *days*
Specifies how many previous days' worth of spooled trace information to collect.

-s *spool_dir*
Captures trace files for the specified spooling directory.

-S *size*
Specifies the maximum cumulative size of all of the trace files to collect (in megabytes).

-t *to_date*
Specifies the date to which you want to collect information. The format of the *to_date* parameter is:

```
yyyy-mm-dd[.hh[:mm[:ss]]]
```

Note: Use **-t** in conjunction with the **-f** flag.

-x runrpttr
Formats the trace file contents of all of the RSCT resource managers.

Using this flag increases the size of the **ctsnap** output files, so you might need to increase the size of the file system that contains the output directory.

-h
Writes the command's usage statement to standard output.

-z
Prevents collecting the **snap caa** information even in a Cluster Aware AIX (CAA) environment.

Security

Only `root` users can run this command.

Exit Status

0
The command ran successfully.

1
The command was not successful.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output.

Standard Error

Error messages are written to standard error (and to the `ctsnap.`*host_name*`.`*nnnnnnnn*`.log` file).

Implementation Specifics

This command is part of the **rsct.core.utils** fileset for the AIX® operating system.

Examples

1. To gather RSCT support information, enter:

```
ctsnap
```

2. To gather RSCT support information and place it in the `/tmp/mydir` directory, enter:

```
ctsnap -d /tmp/mydir
```

3. To capture all trace files for the **/opt/traces** directory, enter:

```
ctsnap -s /opt/traces
```

4. To capture all trace files for the **/opt/traces** directories of the configuration resource manager daemons, enter:

```
ctsnap -s /opt/traces -D '.*ConfigRM.*'
```

5. To capture all trace files for the **/opt/traces** directory for the date range `08-28-2008` to `08-29-2008`, enter:

```
ctsnap -s /opt/traces -f 08-28-2008 -t 08-29-2008
```

6. To capture all trace files for the **/opt/traces** directory for the previous four days, enter:

```
ctsnap -s /opt/traces -p 4
```

7. To capture all trace files for the **/opt/traces** directory for the most recent `50` MB of trace information, enter:

```
ctsnap -s /opt/traces -S 50
```

Location

/opt/rsct/bin/ctsnap
Contains the `ctsnap` command

Files

/tmp/ctsupt
Location of the default directory that contains the output files.

/tmp/ctsupt/ctsnap.*host_name.nnnnnnnn*.log
Location of the log file of the command execution, where *nnnnnnnn* is a timestamp and *host_name* is the name of the host on which the command was run.

tmp/ctsupt/ctsnap.*host_name.nnnnnnnn*.tar.Z
Location of the compressed tar file that contains the collected data, where *nnnnnnnn* is a timestamp and *host_name* is the name of the host on which the command was run.

ctsthl Command

Purpose

Displays and modifies the contents of a cluster security services trusted host list file.

Syntax

`ctsthl` {`-a` | `-d` | `-h` | `-l` | `-s` } [`-f` *trusted_host_list_file*] [`-n` *host_name*] [`-m` *method*] [`-p` *identifier_value*]

Description

This command displays and modifies the contents of a cluster security services trusted host list file. Unless the `-f` flag is provided, the command performs its operations on the trusted host list file configured in the `ctcasd.cfg` file. `ctsthl` allows the command user to add, modify, or remove entries in the trusted host list for specific hosts. When a host is added or modified, the command user must provide the following information:

- The identity of the host (`zathras.ibm.com` or `129.34.128.54`, for example)
- The host identifier value to be used for this host, in a character string format representing the identifier's hexadecimal value (`b87c55e0`, for example)
- The method that was used to generate the host identifier (see the description of the `ctskeygen -i` command)

The command validates the generation method name, converts the character string representation to binary form, and creates a new entry within the trusted host list file for this host. Generally, the host identifier value is quite large. For instance, the character representation of a RSA 1024-bit generated identifier is over 256 characters in size. This can cause a problem on systems such as AIX, which limit the command line length to a smaller size. To avoid this problem, use the `ctsthl -a` command from a shell script, or in conjunction with the `xargs` command.

When the contents of the trusted host list file are displayed, `ctsthl` provides the following information for each entry:

- The network identity of the host
- The host identifier value for that host, represented as a character string
- The method used to generate the host identifier

Flags

-a
Adds to or replaces a host entry in the trusted host list. The `-n`, `-m`, and `-p` flags also must be provided. If the host specified already exists in the trusted host list file, the entry for that host is modified to match the information provided to this command.

-d
Removes a host's entry from the trusted host list file. The `-n` flag also must be provided to indicate the host being removed.

-h
Writes the command's usage statement to standard output.

-l
Instructs the command to list the contents of the trusted host list file. If this flag is combined with the `-a` or `-d` flags the contents are displayed after these flags are processed. If this flag is combined with the `-s` flag, any new entries made by the command are displayed, as well as any public key mismatches detected for host names and IP addresses supported by the local system.

-f *trusted_host_list_file*
Specifies the fully-qualified path name of the trusted host list file. If this flag is not provided, the trusted host list file configured in the `ctcasd.cfg` file is used.

-n *host_name*
Specifies the identity of the host to be used in this operation. The identity should be a host name or IP address specification by which the host is known to the cluster's network.

-m *method*
Instructs the command to use the specified key generation method in creating the host identifier keys. You can use the `ctskeygen -i` command to display valid values for *method*.

-p *identifier_value*
Specifies the host identifier value to be stored for the host. This is a character string that represents the hexadecimal value of the host identifier to be stored for this identifier. For example, if the host identifier value is `0xB87C55E0`, this flag would be specified as `-p b87c55e0`. Generally, In AIX, host identifier keys will be much longer than this example, making it too large for the command line limit on some systems such as AIX. If the resulting command line is too large, use `xargs` to extend it, or issue the command from a shell script.

-s
Explores the local system for all known IP addresses and host names associated with AF_INET-configured and active adapters that the daemon can detect. For any host name or IP address on the local system that is not found in the local system's trusted host list file, an entry is added to associate that value with the local system's public key value.

Parameters

network_ID
Specifies the security network identifier to be mapped. This should be an identity that can be assumed by a client application of a trusted service.

Security

Permissions on the `ctsthl` command permit only `root` to run the command.

Exit Status

0
The command completed successfully.

4
The caller invoked this command incorrectly, omitting required flags and parameters, or using mutually exclusive flags. This command terminated without processing the request.

6
A memory allocation request failed during the operation of this command. The command was unable to complete the requested action.

10
The command was unable to locate any configured and active network (AF_INET) interfaces for the local system while processing the `-s` flag. The local system's identities may not be properly recorded to the trusted host list. Verify that at least one AF_INET or AF_INET6 interface is defined and active on the local system and reissue the command.

12
The command user does not have sufficient permission to view or modify the contents of the trusted host list file.

21
The trusted host list file could not be located, or could not be extended to contain a new public key value.

30
`ctsthl` was unable to obtain exclusive use of the trusted host list file. Another instance of this command may be running and attempting to modify the keys, or the `ctcasd` daemon may be examining these files. Retry the command at a later time.

31
The public key value specified by the `-p` flag does not end on a full byte boundary. Make sure the value contains an even number of digits.

37
The key file appears to be corrupted. Try to view the public key value using the `-d` flag to verify if the file is corrupted. Follow the problem resolution advice listed in the error message for further recovery action.

Restrictions

- Cluster security services supports its own host identifier format and trusted host list file format only.
- Trusted host lists are modifiable using this command only.
- Cluster security services does not provide an automated utility for creating, managing, and maintaining trusted host lists throughout the cluster. This is a procedure left to either the system administrator or the cluster management software.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. When the `-l` flag is specified, the contents of the trusted host list file are written to standard output.

Standard Error

Descriptive information for any detected failure condition is written to standard error.

Examples

1. To view the contents of the trusted host contained in the file `/mythl`, enter:

   ```
   ctsthl -l -f /mythl
   ```

2. To add an entry to the default trusted host list file for the system `zathras.ibm.com`, enter:

   ```
   ctsthl -a -n zathras.ibm.com -m rsa1024 -p 120400a9...
   ```

   Note that this example does not complete the entire identifier value.

3. To add an entry to the default trusted host list file for the system `129.23.128.76`, enter:

   ```
   ctsthl -a -n 129.23.128.76 -m rsa1024 -p 120400a9...
   ```

   Note that this example does not complete the entire identifier value.

4. To remove an entry for zathras.ibm.com from the default trusted host list, enter:

   ```
   ctsthl -d -n zathras.ibm.com
   ```

Location

/opt/rsct/bin/ctsthl
Contains the `ctsthl` command

Files

/opt/rsct/cfg/ctsec_map.global
The default identity mapping definition file. This file contains definitions required by the RSCT cluster trusted services in order for these systems to execute properly immediately after software installation. This file is ignored if the cluster-wide identity mapping definition file `/var/ct/cfg/ctsec_map.global` exists on the system. Therefore, any definitions within this file should also be included in the cluster-wide identity mapping definition file, if that file exists.

/var/ct/cfg/ctsec_map.local
Local override to the cluster-wide identity mapping definitions. Definitions within this file are not expected to be shared between nodes within the cluster.

/var/ct/cfg/ctsec_map.global
Cluster-wide identity mapping definitions. This file is expected to contain identity mapping definitions that are common throughout the cluster. If this file exists on the system, the default identity mapping definition file is ignored. Therefore, if this file exists, it should also contain any entries that would also be found in the default identity mapping definition file.

ctstrtcasd Utility

Purpose

Serves as the launch utility of the `ctcasd` daemon for the cluster security services.

Syntax

`ctstrtcasd [-a ] [-v ]`

Description

The `ctstrtcasd` utility is started by the cluster security services to start the `ctcasd` daemon. This utility is provided as a set-user-identity-on-execution binary file, providing the clients of cluster security services the ability to start the `ctcasd` daemon through the system resource controller (SRC).

The `ctcasd` daemon is used by the cluster security services library when the RSCT host-based authentication (HBA) or enhanced host-based authentication (HBA2) security mechanism is configured and active within the cluster environment. The cluster security services use `ctcasd` when service requesters and service providers try to create a secured execution environment.

When a service requester and a service provider agree to use the RSCT HBA or HBA2 mechanism through the cluster security services, the cluster security services library uses `ctcasd` to obtain and authenticate the RSCT HBA or HBA2 credentials. The cluster security services do not provide a direct interface to the daemon that can be started by user applications.

The `ctcasd` daemon is registered with the SRC as the `ctcas` subsystem. This subsystem is not activated by the SRC until the cluster security services receive a request for the RSCT HBA or HBA2 mechanism. SRC subsystems can be activated only by the system superuser. To allow the cluster security services to process HBA or HBA2 requests for any system user, the cluster security services must be able to activate the `ctcas` subsystem for normal system users as well as the system superuser if the service is not already active. To grant normal system users this ability, the cluster security services start the `ctstrtcasd` utility to start the `ctcas` subsystem if the service is not active. This utility temporarily grants the clients of cluster security services sufficient privilege to start the `ctcas` subsystem.

Flags

-a
Verifies that the `ctcas` subsystem is operational and can process requests from the cluster security services after it is started.

-v
Specifies that the `ctstrtcasd` utility shows status information to standard output and error information to standard error in verbose mode.

Standard output

When the `-v` flag is specified, the status information of this command is written to the standard output.

Standard error

When the `-v` flag is specified, the error information of this command is written to the standard error.

Security

The `ctstrtcasd` utility, a set-user-identity-on-execution binary file, is owned by the `root` system user. This special permission and ownership are required to temporarily grant the clients of the cluster security service the ability to start the `ctcas` subsystem if it is not already active on the system. Without this permission and ownership, some clients of cluster security services might not be able to start the `ctcasd` daemon to handle cluster security services requests, which can result in authentication failures.

See the "Diagnosing cluster security services problems" chapter of the *RSCT: Diagnosis Guide* for more information about the ownership and permissions required for this utility.

Restrictions

This utility is only intended for use by the cluster security services library or as directed by an IBM service representative.

Implementation specifics

This utility is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the `rsct.core.sec` fileset for AIX and `rsct.core` Linux package.

Location

/opt/rsct/bin/ctstrtcasd

ctsvhbac Command

Purpose

Verifies the configuration for the RSCT host-based authentication (HBA) security mechanism on the local system.

Syntax

`ctsvhbac` [[`-d` | `-h` | `-m` | `-s`] | [`-e` *msgnum*[,*msgnum*...]] [`-l` { `1` | `2` | `3` | `4` } | `-b`] [`-p` *pubkeyfile*] [`-q` *pvtkeyfile*] [`-t` *thlfile*]]

Description

The `ctsvhbac` command is a verification utility for the RSCT host-based authentication (HBA) security mechanism. Use the `ctsvhbac` command to verify that the local system has configuration and credential files and information, such as private keys and a trusted host list, ready for the HBA security mechanism to use.

This command performs the following series of tests on the configuration of the HBA security mechanism:

- Verifies that the HBA mechanism configuration file is available and can be processed.
- Verifies that the HBA private key file exists and can be processed.
- Verifies that the HBA public key file exists and can be processed.
- Verifies that the private and public keys for the local system are in pair, which means that the public key is known to be derived from the private key.
- Verifies that the HBA trusted host list file exists and can be processed.
- Checks the contents of the HBA trusted host list for all of the host names and network addresses supported by the local node, determining whether entries exist in the trusted host list file for them. If a host name or network address is found, the command verifies that the same public key value that was used in earlier tests is listed for the name or address.

The command user may specify the private key file, public key file, and trusted host list file to use in the command. By default, this information is extracted from the configuration file for the HBA security mechanism.

Flags

-b
Produces brief output. When this option is used, the command displays only summary output of the tests and any errors detected. Further details of any errors can be determined by reissuing this command without this option. If the `-l` option is specified, this option is ignored.

-d
Displays the list of probes required for successful execution of this command.

-e
Specifies a list of error messages that are not to be displayed by this command during its execution. One or more message numbers may be specified. Message numbers must be in the xxxx-yyy format. Multiple messages are to be separated by commas (,) with no white space characters.

-h
Displays a help message for this command.

-l
Allows the Cluster System Management (CSM) Probe Infrastructure to set the detail level of the output. Accepted levels are:

1
Verbose mode. Displays the command purpose summary and status information for all tests.

2
Displays the command purpose summary and any attention or error conditions detected in any tests.

3
Displays any attention or error conditions detected in any tests.

4
Silent mode. Displays errors detected during the tests.

-m
Displays a detailed description of the command and its purpose.

-p
Specifies the path name of the public key file that is to be used by the command. If this option is not specified, the command will use the public key file currently configured for the HBA security mechanism.

-q
Specifies the path name of the private key file that is to be used by the command. If this option is not specified, the command will use the private key file currently configured for the HBA security mechanism.

-s
Displays a summary of the purpose for the command.

-t
Specifies the path name of the trusted host list file that is to be used by the command. If this option is not specified, the command will use the trusted host list file currently configured for the HBA security mechanism.

Parameters

None.

Security

Permissions on the `ctsvhbac` command permit members of the `bin` user group to execute this command.

Exit Status

Exit status conforms to the CSM Probe Infrastructure conventions.

0
No problems detected. Any messages displayed either are informational or indicate only minor alerts. No administration intervention is required.

10
No problems were detected, but some items found warrant administrator attention. This exit status most commonly occurs if an IP address or host name supported by the local system is not listed in the trusted host list, or is listed with an incorrect public key value. For this exit status, the system administrator should examine the output to determine which conditions were detected, and whether they require corrective action.

To correct the most commonly reported conditions:

- Ensure that any IP addresses or host names that are not in the trusted host list were purposely omitted. If not, update the trusted host list on the local system.
- Repair any entries for local IP addresses and host names that use incorrect public keys.

20
One or more problems were detected. This exit status occurs for the following conditions:

- The HBA security mechanism is configured incorrectly.
- Public and private keys might not be in pair.
- The trusted host list contains none of the IP address or host name values supported by the local system.

Unless these conditions are corrected, authentication requests using the HBA mechanism probably will not be successful on this system. For this exit status, the system administrator must examine the command output to identify and resolve reported problems. To correct reported problems, follow the problem-resolution advice listed in the command output.

127
Unexpected failure in this command. For this exit status, the administrator should verify that at least one network interface is both configured and active on this system.

Restrictions

- Cluster security services supports its own host identifier format and trusted host list file format only.
- Trusted host lists are modifiable using this command only.

- Cluster security services does not provide an automated utility for creating, managing, and maintaining trusted host lists throughout the cluster. This is a procedure left to either the system administrator or the cluster management software.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. When the `-l` flag is specified, the contents of the trusted host list file are written to standard output.

Standard Error

Descriptive information for any detected failure condition is written to standard error.

Examples

To verify the HBA security mechanism, enter:

```
ctsvhbac
```

Output would be similar to:

```
------------------------------------------------------------------------
Host Based Authentication Mechanism Verification Check

Private and Public Key Verifications

      Configuration file:  /opt/rsct/cfg/ctcasd.cfg
                  Status:  Available
                Key Type:  rsa512
                           RSA key generation method, 512-bit key

        Private Key file:  /var/ct/cfg/ct_has.qkf
                  Source:  Configuration file
                  Status:  Available
                Key Type:  rsa512
                           RSA key generation method, 512-bit key

         Public Key file:  /var/ct/cfg/ct_has.pkf
                  Source:  Configuration file
                  Status:  Available
                Key Type:  rsa512
                           RSA key generation method, 512-bit key

              Key Parity:  Public and private keys are in pair

Trusted Host List File Verifications

  Trusted Host List file:  /var/ct/cfg/ct_has.thl
                  Source:  Configuration file
                  Status:  Available

                Identity:  avenger.pok.ibm.com
                  Status:  Trusted host

                Identity:  9.117.10.4
                  Status:  Trusted host

                Identity:  localhost
                  Status:  Trusted host

                Identity:  127.0.0.1
                  Status:  Trusted host

Host Based Authentication Mechanism Verification Check completed
```

Location

/opt/rsct/bin/ctsvhbac
Contains the `ctsvhbac` command

Files

/opt/rsct/cfg/ctsec_map.global
The default identity mapping definition file. This file contains definitions required by the RSCT cluster trusted services in order for these systems to execute properly immediately after software installation. This file is ignored if the cluster-wide identity mapping definition file `/var/ct/cfg/ctsec_map.global` exists on the system. Therefore, any definitions within this file should also be included in the cluster-wide identity mapping definition file, if that file exists.

/var/ct/cfg/ctsec_map.local
Local override to the cluster-wide identity mapping definitions. Definitions within this file are not expected to be shared between nodes within the cluster.

/var/ct/cfg/ctsec_map.global
Cluster-wide identity mapping definitions. This file is expected to contain identity mapping definitions that are common throughout the cluster. If this file exists on the system, the default identity mapping definition file is ignored. Therefore, if this file exists, it should also contain any entries that would also be found in the default identity mapping definition file.

ctsvhbal Command

Purpose

Displays the possible identities that the local system may use to identify itself in RSCT host-based authentication (HBA) security mechanism credentials.

Syntax

`ctsvhbal` [[`-d` | `-h` | `-m` | `-s`] | [`-e` *msgnum*[,*msgnum*...]] [`-l` { `1` | `2` | `3` | `4` } | `-b`]

Description

The `ctsvhbal` command is a verification utility for the RSCT host-based authentication (HBA) security mechanism. It displays the possible identities that the local system may use to identify itself in HBA credentials.

The HBA security mechanism might use either a host name or a network address value as part of the identification information within a credential, depending on the method chosen by the application. If the local system is to service requests from remote systems, at least one network address and host name for that remote system must appear in the trusted host list on the local system. To verify that the remote system can successfully authenticate the local system, system administrators use a combination of RSCT cluster security commands:

1. On both the local and remote system, issue the `ctsvhbac` command to verify that each system has a valid HBA security mechanism configuration.
2. On the local system, issue the `ctsvhbal` command to determine the values that the HBA security mechanism will use to identify this host to a remote system.
3. On the remote system, issue the `ctsvhbar` command, specifying the local system host name or IP address, to determine the value that the remote system will use to verify HBA credentials transmitted from the local system.
4. Compare the `ctsvhbal` and `ctsvhbar` command output to determine whether the two systems are using the same scheme for host-name resolution. If an exact host-name match does not appear in the output, repair the host-name resolution scheme, and repeat the steps above until both commands yield an exact match.

Completing these steps verifies successful authentication in one direction; in other words, the procedure verifies only that the remote system can authenticate requests from the local system. Because RSCT subsystems often use mutual authentication, system administrators also should verify that the local

system can successfully authenticate the remote system. To complete the verification, the following additional steps are required:

- On the remote system, issue the `ctsvhbal` command to determine the values that the HBA security mechanism will use to identify that host to the local system.
- On the local system, issue the `ctsvhbar` command, specifying the remote system host name or IP address, to determine the value that the local system will use to verify HBA credentials transmitted from the remote system.
- Compare the `ctsvhbal` and `ctsvhbar` command output to determine whether the two systems are using the same scheme for host-name resolution. If an exact host-name match does not appear in the output, repair the host-name resolution scheme, and repeat the steps above until both commands yield an exact match.

Completing these additional steps verifies successful authentication when traffic flows in the opposite direction, from the remote system to the local system.

For more detailed instructions and examples, see the cluster security topics in *RSCT Administration Guide*.

Flags

-b
Produces brief output. When this option is used, the command displays only the host identities found for the local system and any errors detected. If the `-l` option is specified, this option is ignored.

-d
Displays the list of probes required for successful execution of this command.

-e
Specifies a list of error messages that are not to be displayed by this command during its execution. One or more message numbers may be specified. Message numbers must be in the xxxx-yyy format. Multiple messages are to be separated by commas (,) with no white space characters.

-h
Displays a help message for this command.

-l
Allows the Cluster System Management (CSM) Probe Infrastructure to set the detail level of the output. Accepted levels are:

1
Verbose mode. Displays the command purpose summary and status information for all tests.

2
Displays the command purpose summary and any attention or error conditions detected in any tests.

3
Displays any attention or error conditions detected in any tests.

4
Silent mode. Displays errors detected during the tests.

-m
Displays a detailed description of the command and its purpose.

-s
Displays a summary of the purpose for the command.

Parameters

None.

Security

Permissions on the `ctsvhbal` command permit members of the `bin` user group to execute this command.

Exit Status

Exit status conforms to the CSM Probe Infrastructure conventions.

0
No problems detected. Any messages displayed are informational. No administration intervention is required.

10
No problems were detected, but the local system is unable to authenticate itself to any remote systems. The local system does not have any active network interfaces, which is a configuration that RSCT permits. For this exit status, however, the system administrator should verify that this configuration is appropriate.

20
One or more problems were detected. Host-name resolution mechanisms that the local system uses are unable to obtain host names of network interfaces that the local system supports. Unless this condition is corrected, authentication requests using the HBA mechanism probably will not be successful on this system. For this exit status, the system administrator should follow the problem-resolution advice listed in the command output.

127
Unexpected failure in this command.

Restrictions

- Cluster security services supports its own host identifier format and trusted host list file format only.
- Trusted host lists are modifiable using this command only.
- Cluster security services does not provide an automated utility for creating, managing, and maintaining trusted host lists throughout the cluster. This is a procedure left to either the system administrator or the cluster management software.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. When the `-l` flag is specified, the contents of the trusted host list file are written to standard output.

Standard Error

Descriptive information for any detected failure condition is written to standard error.

Examples

To display the possible identities that the local system may use to identify itself in HBA credentials, enter:

```
ctsvhbal
```

Output would be similar to:

```
ctsvhbal: The Host Based Authentication (HBA) mechanism identities for
the local system are:

                  Identity:  zathras.pok.ibm.com

                  Identity:  9.127.100.101

ctsvhbal: At least one of the above identities must appear in the
trusted host list on the node where a service application resides in order
```

```
for client applications on the local system to authenticate successfully.
Ensure that at least one host name and one network address identity from the
above list appears in the trusted host list on the service systems used by
applications on this local system.
```

Location

/opt/rsct/bin/ctsvhbal
Contains the `ctsvhbal` command

Files

/opt/rsct/cfg/ctcasd.cfg
Default configuration for the `ctcasd` daemon

/var/ct/cfg/ctcasd.cfg
Configuration for the `ctcasd` daemon, which can be modified by the system administrator

ctsvhbar Command

Purpose

Returns the host name that the RSCT host-based authentication (HBA) security mechanism uses on the local node to verify credentials from a specified host.

Syntax

`ctsvhbar` [[`-d` | `-h` | `-m` | `-s`] | [`-e` *msgnum*[,*msgnum*...]] [`-l` { `1` | `2` | `3` | `4` } | `-b`] {*hostname* | *address*} [*hostname*... | *address*...]

Description

The `ctsvhbar` command is a verification utility for the RSCT host-based authentication (HBA) security mechanism. Use this command when you need to determine which host name the HBA security mechanism uses to verify credentials from a remote system.

The HBA security mechanism might use either a host name or a network address value as part of the identification information within a credential, depending on the method chosen by the application. If the local system is to service requests from remote systems, at least one network address and host name for that remote system must appear in the trusted host list on the local system. To verify that the remote system can successfully authenticate the local system, system administrators use a combination of RSCT cluster security commands:

1. On both the local and remote system, issue the `ctsvhbac` command to verify that each system has a valid HBA security mechanism configuration.
2. On the local system, issue the `ctsvhbal` command to determine the values that the HBA security mechanism will use to identify this host to a remote system.
3. On the remote system, issue the `ctsvhbar` command, specifying the local system host name or IP address, to determine the value that the remote system will use to verify HBA credentials transmitted from the local system.
4. Compare the `ctsvhbal` and `ctsvhbar` command output to determine whether the two systems are using the same scheme for host-name resolution. If an exact host-name match does not appear in the output, repair the host-name resolution scheme, and repeat the steps above until both commands yield an exact match.

Completing these steps verifies successful authentication in one direction; in other words, the procedure verifies only that the remote system can authenticate requests from the local system. Because RSCT subsystems often use mutual authentication, system administrators also should verify that the local

system can successfully authenticate the remote system. To complete the verification, the following additional steps are required:

- On the remote system, issue the `ctsvhbal` command to determine the values that the HBA security mechanism will use to identify that host to the local system.
- On the local system, issue the `ctsvhbar` command, specifying the remote system host name or IP address, to determine the value that the local system will use to verify HBA credentials transmitted from the remote system.
- Compare the `ctsvhbal` and `ctsvhbar` command output to determine whether the two systems are using the same scheme for host-name resolution. If an exact host-name match does not appear in the output, repair the host-name resolution scheme, and repeat the steps above until both commands yield an exact match.

Completing these additional steps verifies successful authentication when traffic flows in the opposite direction, from the remote system to the local system.

For more detailed instructions and examples, see the cluster security topics in *RSCT Administration Guide*.

Flags

-b
Produces brief output. When this option is used, the command displays the host identities provided by the command user, the fully qualified host identities obtained for them, and any errors. If the `-l` option is specified, this option is ignored.

-d
Displays the list of probes required for successful execution of this command.

-e
Specifies a list of error messages that are not to be displayed by this command during its execution. One or more message numbers may be specified. Message numbers must be in the xxxx-yyy format. Multiple messages are to be separated by commas (,) with no white space characters.

-h
Displays a help message for this command.

-l
Allows the Cluster System Management (CSM) Probe Infrastructure to set the detail level of the output. Accepted levels are:

1
Verbose mode. Displays the command purpose summary and status information for all tests.

2
Displays the command purpose summary and any attention or error conditions detected in any tests.

3
Displays any attention or error conditions detected in any tests.

4
Silent mode. Displays errors detected during the tests.

-m
Displays a detailed description of the command and its purpose.

-s
Displays a summary of the purpose for the command.

Parameters

hostname
The host name of a remote system.

address
The network address of a remote system.

Security

Permissions on the `ctsvhbar` command permit members of the `bin` user group to execute this command.

Exit Status

Exit status conforms to the CSM Probe Infrastructure conventions.

0
No problems detected. Any messages displayed are informational. No administration intervention is required.

10
No problems were detected. The command was unable to resolve the host name or IP address provided by the command user. The command user should verify that the correct host name or IP address was used. If the correct name or address was used, the system administrator should verify that the host-name resolution scheme used by the local system permits that name or address to be resolved.

127
Unexpected failure in this command.

Restrictions

- Cluster security services supports its own host identifier format and trusted host list file format only.
- Trusted host lists are modifiable using this command only.
- Cluster security services does not provide an automated utility for creating, managing, and maintaining trusted host lists throughout the cluster. This is a procedure left to either the system administrator or the cluster management software.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. When the `-l` flag is specified, the contents of the trusted host list file are written to standard output.

Standard Error

Descriptive information for any detected failure condition is written to standard error.

Examples

To return the host name that the HBA security mechanism would use on the local node to verify credentials from the host identified by the host name `zathras`, you would enter:

```
ctsvhbar zathras
```

The output would look like this:

```
Host name or network address: zathras
Fully qualified host name
     used for authentication: zathras.ibm.com
```

To return the host name that the HBA security mechanism would use on the local node to verify credentials from the host identified by the network address `9.127.100.101`, you would enter:

```
ctsvhbar 9.127.100.101
```

The output would look like this:

```
Host name or network address: 9.127.100.101
Fully qualified host name
     used for authentication: epsilon3.pok.ibm.com
```

To return the host name that the HBA security mechanism would use on the local node to verify credentials from both the host identified by the host name `zathras`, and the host identified by the network address `9.127.100.101`, you would enter:

```
ctsvhbar zathras 9.127.100.101
```

The output would look like this:

```
Host name or network address: zathras
Fully qualified host name
     used for authentication: zathras.ibm.com
Host name or network address: 9.127.100.101
Fully qualified host name
     used for authentication: epsilon3.ibm.com
```

Location

/opt/rsct/bin/ctsvhbar
Contains the `ctsvhbar` command

Files

/opt/rsct/cfg/ctcasd.cfg
Default configuration for the `ctcasd` daemon

/var/ct/cfg/ctcasd.cfg
Configuration for the `ctcasd` daemon, which can be modified by the system administrator

ctsyschk command

Purpose

The **ctsyschk** command identifies issues while creating a Reliable Scalable Cluster Technology (RSCT) cluster on a node.

Syntax

ctsyschk [*-V*] [*-U* **username**] [*-P* **password**] [*-C* **cluster_type**] [*-i* **ip-addr-type**] [*-p* **checking_subsys_port**] [*-m*] [*-n*] [*-s* **session_auth_type**] *node_name1* [**node_name2...**]

Description

The **ctsyschk** command verifies system settings and networking behaviors that might cause problems in various RSCT-based clustering solutions. If you run the **ctsyschk** command without any options, the usage of the command is displayed as output. Whenever you run the **ctsyschk** command without specifying any hostname, the command checks the local system settings. If you specify one or more hostnames or IP addresses, the **ctsyschk** command runs remote shell commands, such as `ssh` or `rsh`, to collect from the remote targets. The **ctsyschk** command uses information about the local nodes, the available interfaces and node ID, and the port number availability on the remote nodes to make a coordinated decision about the creation on an RSCT cluster.

For running remote commands, the **ctsyschk** command requires the user to set up one of the following:

- A password-less `ssh/rsh`
- A `tcl/tk` expect on node from where the command is being run. In this case, the user must use the *-U* option for the username, which is `root` by default, and the *-P* option for specifying the password for the remote login to all of the nodes.

Limitation: All of the nodes must have the same username and password in order for remote login to work.

Flags

-V
Writes the verbose message of the **ctsyschk** command to the standard output.

-U username
Specifies the username for remote login. The default user is `root`. When there are multiple nodes as input, a single username must exist on all of the nodes.

-P password
Specifies the password for the remote user. You must supply the password for the username that is specified with the *-U* option if the password-less remote-login, `rsh` or `ssh`, is not set. You must specify the password for the `root` user when the username is not specified. If there are multiple nodes as input, there must be a single password for remote login on all of the nodes.

-C cluster_type
Specifies the cluster type. The following values are valid:

p (Peer domain)

If the **ctsyschk** command is run with **cluster_type** option *p*, the following items are checked:

- Existence of IPv4 or IPv6 connectivity between the specified input nodes.
- Resource Monitoring and Control (RMC), High Availability Group Services (HAGS), or Host Access Transformation Service (HATS) port availability in the input nodes.
- Whether any firewall is configured for the RMC port among the specified input nodes.
- The maximum transmission unit (MTU) size compatibility between the nodes.
- The node ID uniqueness among the input nodes.

m (Management domain)

If the **ctsyschk** command is run with the **cluster_type** option *m*, the following items are checked:

- The availability of the RMC port by determining whether a firewall is configured for the RMC port between the logical partitions (LPARs) and Hardware Management Console (HMC).
- The MTU size compatibility between the LPAR and the HMC.

a (Peer Domain and management domain)

If the **ctsyschck** command is run with the **cluster_type** option *a*, the following items are checked:

- Existence of IPv4/IPv6 connectivity between the specified input nodes.
- RMC/HAGS/HATS port availability on the specified input nodes.
- Whether a firewall is configured for the RMC port in the specified input nodes and checks the RMC port availability on the LPAR.
- Whether a firewall is configured for the RMC port between the LPARs and the HMC.

-p checking_subsys_port
Specifies the subsystem to check the port availability. The following values are valid:

r
RMC port.

s
HATS and HAGS port

a
All ports (RMC, HAGS, and HATS)

The default RMC port is 657. If the **ctsyschck** command is run with the *-r* option, the command determines whether the default port is used by any application or any firewall is configured on the RMC port. The default port for topology services is 12347 and default port for group-services is 12348. If the **ctsyschck** command is run with the *s* option, the command checks whether these default ports are free for use. If the default port is not free, a warning or error message is displayed.

-i ip_addr_type

Specifies the IP address type. The following values are valid:

4

IPv4 interfaces

If the **ctsyschck** command is run with the interface value of 4, the command checks for valid IPv4 interfaces and compatible subnets on the input nodes. The subnets that have the same subnet IDs are compatible. If two nodes do not have subnets with common subnet IDs, the **ctsyschck** command reports the information about the subnets.

Depending on the results of the command, you can modify the possible network configurations to create RSCT clusters.

6

IPv6 interfaces

If the **ctsyschck** command is run with a value of 6, the command checks for non-link local interfaces and valid IPv6 interfaces, and the command does not run any subnet compatibility verification for IPv6 interfaces.

Note: To check the information about valid interfaces on remote nodes, the `rsh` or `ssh` protocol is used. The usage of the `rsh` or `ssh` protocol depends on the *CT_SYSCHK_SHELL* environment variable.

a

Checks the availability of IPv4 and IPv6 interfaces.

–m

The **ctsyschk** command checks for compatible MTU size between the specified input nodes based on the below conditions.

Tool should have identified valid IPv4 communication path between the nodes. And then taking the interface names between which, the valid communication path is possible, MTU size compatibility check is performed.

-n

Specifies whether the node ID check to be performed.

This check is valid for more than one node. If only one node is specified, this check is not performed and a default status of `success` is returned.

If the **ctsyschk** command is run with the *-n* option and more than one node name is specified as the input, the command checks for the uniqueness of node ID among all the specified nodes.

-s

Specifies the RMC session authentication type to verify. The following values are valid:

u

Unauthenticated RMC session to the remote nodes from a local node. For example, the *u* option might attempt to query the *IBM.HostPublic* resource class of the remote nodes.

a

Authenticated RMC session to the remote nodes from a local node. For example, the *a* option might attempt to query the *IBM.Host* resource class of the remote nodes.

Parameters

node_name1 [node_name2...]
Specifies the nodes to be included in the peer domain definition by running the **ctsyschk** command. The node name is the long name or short name of the Domain Name System (DNS) host name.

Environment Variables

CT_SYSCHK_SHELL
Shell to contact remote nodes. `ssh` and `rsh` are two examples of shells used to contact remote nodes.

CT_SYSCHK_TS_PORT
User-configured port for HATS. If this environment variable is set, the default port 12347 is not used.

CT_SYSCHK_GS_PORT
User-configured port for HAGS. If this environment variable is set, the default port 12348 is not used.

Exit Status

0
The command completed successfully.

1
No valid IP connectivity available between the given nodes.

2
No compatible subnet exists between the given nodes.

3
Port check failed.

4
Firewall settings do not allow the required communication to go through.

5
`NodeId` is not unique across the given nodes.

6
Incompatible MTU size detected between the specified nodes.

7
Connection failure occurred. Remote node cannot be reached.

8
Other error.

Restrictions

- All the input nodes must be reachable from the node on which this command runs and none of the input node names must be an IP address.
- The **ctsyschk** command is part of the RSCT fileset for the AIX operating system.
- When the *-h* flag is specified, the usage statement of the command is written to the standard output. All verbose messages are written to the standard output.
- All error messages are written to standard error.

Examples

1. To check whether nodes *nodeA* and *nodeB* can form Peer Domain, run the following command:

```
ctsyschk -C p nodeA nodeB
```

2. To check whether the IPV4 communication path exists between the nodes *nodeA* and *nodeB* run the following command:

```
ctsyschk -i 4 nodeA nodeB
```

3. To check management domain compatibility between HMC node *nodeH* and lpar node *nodeA*, run the following command:

```
ctsyschk -C m nodeH nodeA
```

4. To check the RMC connection between nodes *nodeA* and *nodeB*, run the following command:

```
ctsyschk -s {u/a} nodeA nodeB
```

5. To check MTU compatibility between nodes *nodeA* and *nodeB* for making peer-domain, run the following command:

```
ctsyschk -m -C p nodeA nodeB
```

6. To check *RMC*, *HAGS* and *HATS* ports for availability on the local node, run the following command:

```
ctsyschk -p a -C p
```

7. To check whether nodes *nodeA* and *nodeB* can form Peer Domain, run the following command:

```
ctsyschk -p a -C p
```

cttracecfg Command

Purpose

Changes the trace configuration and the spool area configuration dynamically.

Syntax

To change the trace configuration:

```
cttracecfg -T [ -l | -a | -u | -r ] [ -n section_name ] [ -p pattern ] [ -d dir ] [ -s size ]
```

```
[ -o on | off ] [ -h ]
```

To configure the spool area management:

```
cttracecfg -S [ -l | -a | -u | -r ] [ -n section_name ] [ -d dir ] [ -i interval ]
```

```
[ -t retention_days ] [ -c max_size ] [ -o on | off ] [ -h ]
```

Description

The **cttracecfg** command is used to turn on or off the trace spooling dynamically or to configure the cleanup activity on the spooling directory.

The **ctracecfg** command can be run with the **-T** flag to work on trace configuration (to enable or disable trace spooling) or with the **-S** flag to work on spool configuration (to clean up the spool directory).

Trace configuration

You can change the trace configuration by using the **cttracecfg** command. The trace configuration changes are dynamically picked by the Reliable Scalable Cluster Technology (RSCT) daemons and the required changes are applied to the daemon's trace configuration.

A reserved section called `default` represents the default values for the following attributes if these attributes are not defined in the trace configuration sections:

Attribute	Description
spooling	Specifies whether trace spooling is enabled or disabled.
tracesize	Specifies the total trace size.
dest	Specifies the spool destination directory.

You can overwrite the default behavior of the trace spooling by using a specialized section for a trace file. In the specialized section, you can change the **spooling**, **tracesize**, and **dest** attributes to change the daemon's trace behavior.

You can perform the following operations on trace configuration sections:

- Query or list all the trace sections.
- Add a section.
- Change a section.
- Delete a section.

Spool area management

You can change the spool area management policies by using the **cttracecfg** command and by using one of the following methods:

- Enable or disable the cleanup activity on the spool area.
- Change the cleanup interval of the spool area.
- Change the number of retention days of the spooled files.
- Change the maximum allowed size of the spool directory.

You can perform the following operations on spool area management sections:

- Query or list all the spool area management sections.
- Add a spool area management section.
- Change a spool area management section.
- Delete a spool area management section.

Note: The name of the spool area management section must start with the `spoolarea_` string.

Flags

Trace configuration flags

Flag	Description
-T	Designates the **cttracecfg** command to work on dynamic tracing sections.
-l	Lists the trace configuration sections.
-a	Adds a trace configuration section.
-u	Updates a trace configuration section.
-r	Deletes a trace configuration section.
-n *section_name*	Specifies a particular section in the configuration file.
-p *pattern*	Specifies the pattern of the trace file directory.

Flag	Description
-o [**on** | **off**]	Turns on or off the trace spooling mechanism. The valid values of this flag are as follows: **on** Enables the trace spooling mechanism and copies the files to the spool directory. **off** Disables the trace spooling mechanism.
-d *dir*	Specifies the destination directory path.
-s *size*	Specifies the size of the trace in bytes.
-h	Displays the usage information for this command.

Spool area management flags

Flag	Description
-S	Designates the **cttracecfg** command to work on trace spool area management sections.
-l	Lists the spool area management sections.
-a	Adds a spool area management section.
-u	Updates a spool area management section.
-r	Deletes a spool area management section.
-n *section_name*	Specifies a particular section in the configuration file.
-o [**on** | **off**]	Removes the old trace files from the spool directory. The valid values of this flag are as follows: **on** Removes the old trace files. **off** Does not remove the old trace files.
-d *dir*	Specifies the destination directory path.
-i *interval*	Specifies the cleanup interval in hours.
-t *retention_days*	Specifies the number of retention days for a spooled file.
-c *max_size*	Specifies the maximum allowed capacity of the trace spool area in MB.
-h	Displays the usage information for this command.

Exit Status

0
The command completed successfully.

1
An error occurred.

Examples

1. To query all the dynamic trace sections, type the following command:

```
cttracecfg -T -l
```

2. To query the default trace section, type the following command:

```
cttracecfg -T -l -n default
```

3. To query the dynamic trace section `section_test`, type the following command:

```
cttracecfg -T -l -n section_test
```

4. To configure the trace spooling mechanism for the resource monitoring and control (RMC) daemon that has a trace size of 2 MB and a destination directory path `/data/trc`, type the following command:

```
cttracecfg -T -a -n RMCD -p "/var/ct/.*/log/mc/.*" -d "/data/trc" -s 2097152 -o on
```

5. To add a spool area management section in the `/data/trc` directory such that the directory is checked every 12 hours and the spooled files are retained for 14 days before removing the spooled files, type the following command:

```
cttracecfg -S -a -n spoolarea_data -d /data/trc -i 12 -t 14 -o on
```

6. To delete the trace files from the trace spool area `/data/trc` if the spool area exceeds 50 MB size, type the following command. Also, the spool directory must be checked every 12 hours.

```
cttracecfg -S -a -n spoolarea_data -d "/data/trc" -i 12 -c 50 -o on
```

Location

/opt/rsct/bin/cttracecfg
Contains the **cttracecfg** command.

Files

/var/ct/cfg/trace.conf
Contains the trace configuration and spool area configuration.

cu Command

Purpose

Connects directly or indirectly to another system.

Syntax

To Establish a Connection Using a Modem

cu [**-d**] [**-h**] [**-m**] [**-T***Seconds*] [**-n**] [**-s***Speed*] [**-t**] [**-e** | **-o**] *TelephoneNumber*

To Specify the Name of a Device for a Connection

cu [**-d**] [**-h**] [**-m**] [**-T***Seconds*] [**-s***Speed*] [**-e** | **-o**] **-l***Line*

To Specify a System Name for a Connection

cu [**-d**] [**-h**] [**-m**] [**-T***Seconds*] [**-e** | **-o**] *SystemName*

Description

The **cu** command is a Basic Networking Utilities (BNU) command that connects one system to a terminal connected to either a UNIX system or other system. The connection can be established over a hardwired line or over a telephone line using a modem.

Once the connection is established, a user can be logged in on both systems at the same time, executing commands on either one without dropping the BNU communication link. If the remote computer is also running under UNIX, the user can transfer ASCII files between the two systems.

After issuing the **cu** command from the local system, the user must press the Enter key and then log in to the remote system. After making the connection, the **cu** command runs as two concurrent processes: the transmit process reads data from standard input and, except for lines beginning with a ~ (tilde), passes that data to the remote terminal.

The receive process accepts data from the remote system and, except for lines beginning with a ~, passes it to standard output. Internally, the program accomplishes this by initiating an output diversion to a file on the local system when a line from the remote system begins with ~> (tilde, greater than). The trailing ~> marks the end of the diversion. To control input from the remote system so the buffer is not overrun, the **cu** command uses an automatic **DC3/DC1** (Ctrl-Q/Ctrl-S) protocol.

The **cu** command can be used to connect multiple systems, and commands can then be executed on any of the connected systems. For example, the user can issue the **cu** command on system X to connect to system Y, and then issue the **cu** command on system Y to connect to system Z. System X is then the local computer, and systems Y and Z are remote computers.

The user can execute commands on system Z by logging in and issuing the command. Commands can be executed on system X by prefixing the command with a single tilde (**~***Command*) and on system Y by prefixing the command with two tildes (**~~***Command*). In general, one tilde causes the specified command to be executed on the original local computer, and two tildes cause the command to be executed on the next system on which the **cu** command was issued.

For example, once the multiple systems are connected, the user can execute the **uname -n** command (to display the node name) on systems Z, X, and Y as follows:

```
$ uname -n
Z
$ ~!uname -n
X
$ ~~!uname -n
Y
```

Notes:

1. The **cu** command does not do integrity checking on data it transfers.
2. Data fields with special **cu** characters may not be transmitted properly.
3. The exit code is 0 for normal exit, otherwise, -1.

In addition to issuing regular commands on the remote system, the user can issue special **cu** command subcommands, which are preceded by a **~** (tilde). Use these subcommands to issue commands on the local system and to perform tasks such as transferring files between two UNIX systems. As soon as the user enters the **~!**, **~$**, **~%**, **~l**, or **~t** subcommand, the system displays the name of the local computer in a format similar to the following:

```
~[SystemName]/%
```

The user then enters the subcommand to be executed on the local computer.

Flags

Item	Description
-d	Prints diagnostic traces.
-e	Designates that even parity is to be generated for data sent to the remote system.
-h	Emulates local echo, supporting calls to other systems that expect terminals to be set to half-duplex mode.

Item	Description
-l*Line*	Specifies the name of a device to be used as the line of communication between the local and the remote system. This can be used to override the search that would otherwise take place for the first available line with the right speed. When the **-l** flag is used without the **-s** flag, the speed of the *Line* is taken from the **Devices** file(s) (by default, the **/etc/uucp/Devices** file). When the **-l** and **-s** flags are used together, the **cu** command searches the **Devices** file(s) to check whether the requested speed is available for the specified line. If so, the connection is made at the requested speed; otherwise, an error message is printed, and the call is not made. The specified device is generally a hardwired asynchronous line (for example, **/dev/tty2**), in which case the *TelephoneNumber* parameter is not required. If the specified device is associated with a modem, a telephone number must be provided. Using this flag with the *SystemName* parameter rather than with *TelephoneNumber* parameter does not give the desired result. Under ordinary circumstances, the user should not have to specify the transmission speed or a line or device. The defaults set when BNU is installed should be sufficient.
-m	Instructs the **cu** command to ignore modem control signal data carrier detect (DCD).
-n	For added security, prompts the user to provide the telephone number to be dialed, rather than taking it from the command line.
-o	Designates that odd parity is to be generated for data sent to the remote system.
-s*Speed*	Specifies the rate at which data is transmitted to the remote system (300, 1200, 2400, 4800, 9600, or 19200 baud).The default value is **Any** speed, which instructs the system to use the rate appropriate for the default (or specified) transmission line. The order of the transmission lines is specified in the BNU **Devices** file(s) (by default, the **/etc/uucp/Devices** file). Most modems operate at 300, 1200, or 2400 baud, while most hardwired lines are set to 1200 baud or higher. When transferring data such as a file between a local and a remote system, a speed of 300 baud may occasionally be needed. The lower baud rate results in less interference on the line.
-t	Used to dial an ASCII terminal that has been set to autoanswer. Appropriate mapping of carriage-return to carriage-return line feed pairs is set.
-T*Seconds*	Specifies the maximum number of seconds to wait before timing out. The default is 45 seconds. **Note:** You can also enter `WAIT=n` before any send string in the **Dialers** file. Where n is the number of seconds to wait before timing out.

Parameters

Item	Description
SystemName	The name of the remote system, recognized by BNU, with which a connection is established. A system name can be used rather than a telephone number; in that case, the **cu** command obtains an appropriate hardwired line or telephone number from the BNU **Systems** file(s) (by default, the **/etc/uucp/Systems** file). System names must be ASCII characters only. **Note:** Do not use the *SystemName* flag with the **-l** flag and the **-s** flag. If you do, the **cu** command connects to the first available line for the requested system name, ignoring the specified line and speed.

Item	Description
TelephoneNumber	The telephone number used to establish a remote connection using a modem. This entry can be either a local or a long-distance telephone number.

Subcommands

The **cu** command transmit process interprets lines beginning with a **~** (tilde) in the following ways:

Item	Description
~!	Returns the user to an interactive shell on the local system. Toggle between the local and remote systems using **~!** (remote to local) and Ctrl-D (local to remote).
~%break	Transmits a break sequence to the remote system. The break can also be specified as **~%b**.
~%cd *DirectoryName*	Changes the directory on the local system from the current directory to the directory specified by the *DirectoryName* variable.
~%debug	Toggles the **-debug** flag on or off; this can also be specified as **~%d**.
~%nostop	Toggles between **DC3/DC1** input control protocol and no input control. This is useful in case the remote system is one that does not respond properly to the **DC3** and **DC1** characters.
~%put *From* [*To*]	Copies the *From* file on the local system to the *To* file on the remote system. If the *To* variable is omitted, the local file is copied to the remote system under the same file name. As each block of the file is transferred, consecutive single digits are displayed on the terminal screen. Only ASCII files can be transferred using this subcommand. The use of the **~%put** subcommand requires the **stty** command and the **cat** command on the remote system. It also requires that the current erase and kill characters on the remote system be identical to these current control characters on the local system. Backslashes are inserted at appropriate places in the transmitted data. There is an artificial slowing of transmission by the **cu** command during the **~%put** operation so that loss of data is unlikely.
~%take *From* [*To*]	Copies the *From* file on the remote system to the *To* file on the local system. If the *To* variable is omitted, the remote file is copied to the local system under the same file name. As each block of the file is transferred, consecutive single digits are displayed on the terminal screen. Only ASCII files can be transferred using this subcommand. The use of the **~%take** subcommand requires the **echo** command and the **cat** command on the remote system. Also, **stty tabs** mode should be set on the remote system if tabs are to be copied without expansion to spaces.
~.	Logs the user off the remote computer and then terminates the remote connection. Usually the connection terminates when you log off the remote computer. However, with some types of interconnection hardware, it may be necessary to use a **~.** to terminate the conversation after the normal logoff sequence has been used.
~!*Command*	Executes, on the local system, the command denoted by the *Command* variable.
~$*Command*	Runs, on the local system, the command denoted by the *Command* variable, then sends the command's output to the remote system for execution.

Item	Description
~l	Prints the values of the **TERMIO** structure variables for the remote communication line. This is useful for debugging.
~t	Prints the values of the **TERMIO** structure variables for the user's terminal. This is useful for debugging.
~~*String*	Sends the string denoted by the *String* variable to the remote system.

Examples

The following are examples of connecting to a remote system.

1. To connect to a remote system, enter:

   ```
   cu venus
   ```

   In this example, you are connected to the remote system `venus`. System `venus` must be listed in one of the local **Systems** files (by default, the **/etc/uucp/Systems** file or one of the **Systems** files listed for the **cu** command in the **/etc/uucp/Sysfiles** file).
2. To dial a remote system and set the baud rate, enter:

   ```
   cu  -s1200 9=12015558391
   ```

   In this example, you dial a remote system whose telephone number is `1-201-555-8391`, where dialing 9 is required to get an outside dial tone. The baud rate is set to `1200`.
3. To log in to a system connected by a hardwired line asynchronous line, enter:

   ```
   cu  -l /dev/tty2
   ```

   The **cu** command contacts the system connected to the `tty2` device.
4. To dial a remote system with a specified line and a specific speed, enter:

   ```
   cu  -s 1200  -l tty3
   ```

   The command contacts the system connected to the `tty3` device, using a speed of `1200` baud.
5. To dial a remote system using a specific line associated with a modem, enter:

   ```
   cu  -l cul4 9=12015558391
   ```

   In this example, you dial a remote system whose telephone number is `1-201-555-8391`, where dialing 9 is required to get an outside dial tone. The **cu** command uses the modem connected to the `cul4` device.

1. To display the contents of a file after logging in to the remote system, enter:

   ```
   ~!pg /usr/msg/memos/file10
   ```

   The **~!** subcommand executes the **pg** command on the local system, displaying the contents of the `file10` file in the `/usr/msg/memos` directory on the local system.
2. To copy a file from the local system to the remote system without changing the name of the file, enter:

   ```
   ~%put /home/amy/file
   ```

   The `/home/amy/file` file is copied from the local system to the remote system without changing the name of the file.
3. To copy a file from the local system to the remote system and change the file name, enter:

   ```
   ~%put /home/amy/file /home/amy/tmpfile
   ```

   The `/home/amy/file` file is copied from the local system to the remote system and the file name changed to `/home/amy/tmpfile`.

4. To copy a file from the remote system to the local system without changing the name of the file, enter:

   ```
   ~%take /home/jeanne/test1
   ```

   The `/home/jeanne/test1` file is copied from the remote system to the local system without changing the name of the file.
5. To copy a file from the remote system to the local system and change the file name, enter:

   ```
   ~%take /home/jeanne/test1 /usr/dev/jeanne/tmptest
   ```

   In this example, the `/home/jeanne/test1` file is copied from the remote system to the local system and the file name changed to `/usr/dev/jeanne/tmptest`.

Files

Item	Description
/etc/locks	Prevents multiple use of device.
/usr/bin/cu	Specifies the path name of the **cu** command.
/bin/cu	Specifies a symbolic link to the **/usr/bin/cu** command.
/etc/uucp/Devices	Contains information about available links.
/etc/uucp/Dialcodes	Contains dialing code abbreviations.
/etc/uucp/Dialers	Controls initial handshaking on a link.
/etc/uucp/Permissions	Contains access permission codes.
/etc/uucp/Systems	Lists accessible remote systems.
/etc/uucp/Sysfiles	Specifies alternate files to be used as **Systems**, **Devices**, and **Dialers** files.

curt Command

Purpose

Generates CPU utilization report from a trace.

Syntax

curt -i *inputfile* [-o *outputfile*] [-n *gensymsfile*] [-m *trcnmfile*] [-a *pidnamefile*] [-f *timestamp*] [-l *timestamp*] [**-r** PURR] [-ehpstP] [-@ {ALL | *WparList*}]

Description

The **curt** command takes an AIX trace file as input and produces a number of statistics related to processor (CPU) utilization and process/thread/pthread activity. The command will work with both uniprocessor and multiprocessor AIX traces if the processor clocks are properly synchronized.

The AIX trace file which is gathered using the **trace** command should contain at least the trace events (trace hooks) listed below. These are the events **curt** looks at to calculate its statistics:

```
   HKWD_KERN_SVC, HKWD_KERN_SYSCRET, HKWD_KERN_FLIH, HKWD_KERN_SLIH,
   HKWD_KERN_SLIHRET, HKWD_KERN_DISPATCH, HKWD_KERN_RESUME, HKWD_KERN_IDLE,
   HKWD_SYSC_FORK, HKWD_SYSC_EXECVE, HKWD_KERN_PIDSIG, HKWD_SYSC_EXIT
   HKWD_SYSC_CRTHREAD, HKWD_KERN_INITP, HKWD_NFS_DISPATCH, HKWD_CPU_PREEMPT,
   HKWD_DR, HKWD_KERN_PHANTOM_EXTINT, HKWD_RFS4_VOPS, HHKWD_RFS4_VFSOPS, HKWD_RFS4_MISCOPS,
HKWD_RFS4,
   HKWD_KERN_HCALL, HKWD_WPAR,
   HKWD_PTHREAD_VPSLEEP, HKWD_PTHREAD_GENERAL
```

This means that, if you specify the **-j** flag on your **trace** command, you must include these numbers for **curt**:

```
-j
100,101,102,103,104,106,10C,119,134,135,139,200,210,215,38F,419,465,47F,488,489,48A,48D,492,4C9,
605,609
```

Or, you can use **-J curt** instead.

To get the PTHREAD hooks into the trace, you must execute your `pthread` application using the instrumented `libpthreads.a`. One way to cause that to happen is to perform the following three steps before starting your application (KornShell syntax):

1. mkdir /temp.lib; cd /temp.lib
2. ln -s /usr/ccs/lib/perf/libpthreads.a
3. export LIBPATH=$PWD:$LIBPATH

Putting the instrumented library directory in LIBPATH is necessary to activate the user `pthread` instrumentation; the `temp.lib` directory can be put anywhere.

Flags

Item	Description
-i *inputfile*	Specifies the input AIX trace file to be analyzed.
-o *outputfile*	Specifies the output file (default is stdout).
-n *gensymsfile*	Specifies a names file produced by **gensyms**.
-m *trcnmfile*	Specifies a names file produced by **trcnm**.
-a *pidnamefile*	Specifies a PID to process name mapping file.
-f *timestamp*	Starts processing trace at *timestamp* seconds.
-l *timestamp*	Stops processing trace at *timestamp* seconds.
-r PURR	Uses the PURR register to calculate CPU times.
-e	Outputs elapsed time information for system calls and pthread calls.
-h	Displays usage text (this information).
-p	Outputs detailed process information.
-s	Outputs information about errors returned by system calls.
-t	Outputs detailed thread information.
-P	Outputs detailed pthread information.

Item	Description
-@	Controls the addition of workload partition information to a curt report. You can use the **-@** flag in one of the following forms: **-@** Outputs a summary of workload partitions. The summary includes the processor usage for workload partitions in various execution modes. In addition, WPAR names are shown for listed processes summarizing the processor usage by processes, threads, or pthreads. **-@ All** Outputs reports for the system and all of the workload partitions. The reports are delimited by three lines containing WPAR names or SYSTEM for the overall system. **-@ *WparList*** Outputs reports for the workload partitions specified by the *WparList* parameter, which is a comma-separated list of WPAR names. The reports are delimited by three lines containing WPAR names.

If the **trace** process name table is not accurate, or if more descriptive names are desired, use the **-a** flag to specify a PID to process name mapping file. This is a file with lines consisting of a process ID (in decimal) followed by a space followed by an ASCII string to use as the name for that process.

If the input AIX-trace file is created with the **-n** flag specified, curt will use that address/name table to resolve System Call and Slih addresses to names *if* you do not specify a **-m** or a **-n** flag on the curt command line.

If the input AIX-trace file is created in a workload partition, the **curt** command prints a WPAR report. The **-@** flag is not allowed in this case.

Report Contents

The curt report includes the following information:

curt and Trace Information

The first lines in the curt report give the time when the curt program was executed and the command line used to invoke curt. Following that is this information about the AIX trace file being processed by **curt**: name, size, creation date, and the command used to gather the trace file.

The line `PURR was used to calculate CPU times` is printed if the **-r** PURR option was used and the trace file includes the PURR register.

System Summary

The first major section of the report is the System Summary. This section describes the time spent by the system as a whole (all processors) in various execution modes. These modes are as follows:

APPLICATION
The sum of times spent by all processors in User (non-privileged) mode.

SYSCALL
The sum of times spent by all processors doing System Calls. This is the portion of time that a processor spends executing in the kernel code providing services directly requested by a user process.

HCALL
The sum of times spent by all processors doing Hypervisors Calls. This is the portion of time that a processor spends executing in the hypervisor code providing services directly requested by the kernel.

KPROC
The sum of times spent by all processors executing kernel processes other than the IDLE process and NFS processes. This is the portion of time that a processor spends executing specially created dispatchable processes which only execute kernel code.

NFS
The sum of times spent by all processors executing NFS operations. NFS operations begin with RFS_DISPATCH_ENTRY and end with RFS_DISPATCH_EXIT subhooks for NFS V2/V3. NFS operations begin with `start` and end with `done` or `done error` for NFS V4.

FLIH
The sum of times spent by all processors in FLIHs (first level interrupt handlers).

SLIH
The sum of times spent by all processors in SLIHs (second level interrupt handlers).

DISPATCH
The sum of times spent by all processors in the AIX dispatch code. This sum includes the time spent in dispatching all threads (i.e. it includes the dispatches of the IDLE process).

IDLE DISPATCH
The sum of times spent by all processors in the AIX dispatch code where the process being dispatched was the IDLE process. Because the DISPATCH category includes the IDLE DISPATCH category's time, the IDLE DISPATCH category's time is not separately added to calculate either CPU(s) busy time or TOTAL (see below).

CPU(s) busy time
The sum of times spent by all processors executing in application, syscall, kproc, flih, slih, and dispatch modes.

IDLE
The sum of times spent by all processors executing the IDLE process.

TOTAL
The sum of CPU(s) busy time and IDLE. This number is referred to as "total processing time."

The column labeled `processing total time (msec)` gives the total time (in milliseconds) for the corresponding processing category. The column labeled `percent total time` gives the processing total time as a percentage of the TOTAL processing total time. The column labeled `percent busy time` gives the processing total time as a percentage of the CPU(s) busy time processing total time. The `Avg. Thread Affinity` is the probability that a thread was dispatched to the same processor that it last executed on.

The Total Physical CPU time (msec) is the real time the CPU(s) were running (not preempted). The Physical CPU percentage gives the Physical CPU(s) Time as a percentage of total time.

Note: In a WPAR report, the system summary information is labeled "WPAR summary".

System Application Summary

Following the System Summary is the System Application Summary, which describes the time spent in User mode in details. This section describes the time spent by all processes (on all processors) executing various parts of libpthreads.

PTHREAD
The sum of times spent by all pthreads in traced libpthreads operations.

PDISPATCH
The sum of times spent by all pthreads in the libpthreads dispatch code.

PIDLE
The sum of times spent by all pthreads in libpthreads vp_sleep code.

OTHER
The sum of time spent by all threads in user mode outside traced libpthreads operations.

APPLICATION time
The sum of times spent by all processors in user mode.

The column labeled `processing total time (msec)` gives the total time in milliseconds for the corresponding processing category. The column labeled `percent total time` gives the processing total time as a percentage of the TOTAL processing total time of System Summary. The column labeled `percent application time` gives the processing total time as a percentage of the APPLICATION processing total time. The `Avg. Pthread Affinity` is the probability that a pthread was dispatched to the same thread that it last executed on.

Note: In a WPAR report, the system application summary information is labeled "WPAR application summary".

s Summary

The WPARs Summary of the report is generated when you specify the **-@** flag. The following system and system application information for workload partitions, shown as column headings in the summary, describes the time spent in all of the workload partitions in details:

appli
Percent of the total process time that was spent by the WPAR in user mode (non-privileged).

syscall
Percent of the total process time that was spent by the WPAR performing system calls.

hcall
Percent of the total process time that was spent by the WPAR performing hypervisor calls.

kproc
Percent of the total process time that was spent by the WPAR running kernel processes calls.

nfs
Percent of the total process time that was spent by the WPAR running NFS operations.

flih
Percent of the total process time that was spent by the WPAR in the first-level interrupt handlers.

slih
Percent of the total process time that was spent by the WPAR in the second-level interrupt handlers.

total
Percent of the total process time that was spent by the WPAR.

total(msec)
The sum of processor time, in milliseconds, used by the WPAR.

WPAR
The WPAR name.

Note: The WPARs Summary is generated only in an overall system report.

Per Processor Summary

Following the System Application Summary is the Per Processor Summary, which is essentially the same information but broken down on a processor by processor basis. In the description given for the System Summary, the phrase "sum of times spent by all processors" can be replaced by "time spent by this processor". The Total number of process dispatches refers to how many times AIX dispatched any non-IDLE process on this processor, while Total number of idle dispatches gives the count of IDLE process dispatches.

The `Total Physical CPU time (msec)` is the real time the processor was running (not preempted). The `Physical CPU percentage` gives the Physical CPU Time as a percentage of total time.

`Physical processor affinity` is the probability that a logical processor was dispatched on the same physical processor that it last executed on. Total number of preemptions is the number of times the virtual processor was redispatched on a physical CPU.

`Total number of H_CEDE` is the number of H_CEDE hypervisor call done by this processor; `with preeemption` indicates the number of H_CEDE calls resulting in preemption.

`Total number of H_CONFER` is the number of H_CONFER hypervisor call done by this processor; `with preeemption` indicates the number of H_CONFER calls resulting in preemption.

Note: A per processor summary is not generated in a WPAR report.

Per Processor Application Summary

Following each Processor Summary is the Per Processor Application Summary, which is essentially the same information as System Application Summary but broken down on a processor by processor basis.

The `Total number of pthread dispatches` refers to how many times libpthreads dispatched any pthread on this processor, while `Total number of pthread idle dispatches` gives the count of calls to vp_sleep.

Note: A per processor application summary is not generated in a WPAR report.

Application Summary

The second major section of the report is the Application Summary. The first part of this section summarizes the total system processing time on a per-thread basis (by Tid). For each thread, identified by Process ID (and name if available) and Thread ID, the summary gives the total application (same as APPLICATION above) and syscall (same as SYSCALL above) processing time in milliseconds and as the percentage of the total system processing time for all processors in the trace. In addition, the summary gives the sum of those two times, both as raw time, and as a percentage of the total processing time.

The second part of this section gives the same information on a per-process ID (by Pid) basis. The third part of this section gives the same information on a per-process name (by process type) basis.

The fourth part of this section gives similar information for kernel process threads (Kproc Summary). Since most kprocs provide a specific kernel service, the total processing time is split into two categories, operation and kernel, which loosely correspond to syscall and application for a process which always runs in kernel code. Each kproc thread is identified by name, Process ID, Thread ID and type of kproc if known. The kproc types are listed and described in a table immediately following this summary.

The fifth part of this section is the Pthread Process Summary. This section gives the total application time on multi-threaded Process (by Pid). For each process, identified by Process ID (and name if available), the summary gives the total application, pthread and other processing time in milliseconds and as the percentage of the total application time for all processors in the trace.

All five sections of the Summary are presented in sorted order from most combined processing time to least.

In all five sections of an Application Summary, the WPAR name is added to identify the thread or process if you specify the **-@** flag.

Note: Pids and Tids (Process and Thread IDs) are always given in decimal.

System Calls Summary

The third major section of the report is the System Calls Summary. This section summarizes the processing time spent in system calls. For each system call (SVC), identified by kernel address (and name if available), the summary gives the number of times the SVC was called and the total processor time for all calls in milliseconds and as a percentage of total system processing time for all processors in the trace. In addition, the summary gives the average, minimum and maximum times for one call to the SVC. If the **-e** flag is specified, the summary gives the total elapsed time for all calls to the SVC and the average, minimum and maximum elapsed times for one call. Elapsed time is the wall-clock time from when the process starts executing the SVC in kernel mode until the process resumes executing in application mode. The Summary is presented in sorted order from most total processor time to least. If the **-s** flag is specified, the summary gives the number of times each error code (errno) was returned by each System Call.

The second part of this section is the Pending System Calls Summary. This part lists the System Calls which have started but not completed. The time that is given is included in the SYSCALL time for the system and the various processors and is included in the syscall time for the pthread, thread and process which issued the SVC, but is not included in the processing time for the system call in the first part of this section. The pending call is also not included in the count given in the first part of this section.

Note:

1. System call addresses are always given in hexadecimal. Pids and Tids are always given in decimal.
2. WPAR names are added in a System Calls Summary to identify threads or processes if you specify the **-@** flag.

System Hypervisor Calls Summary

If there is hypervisor activity in the trace, an additional section is inserted at this point of the report. This major section of the report is called `Hypervisor Calls Summary`. This section summarizes the processing time spent in hypervisor calls. For each Hypervisor call (HCALL), identified by name (and kernel address), the summary gives the number of times the HCALL was called and the total processor time for all calls in milliseconds and as a percentage of total system processing time for all processors in the trace. In addition, the summary gives the average, minimum and maximum times for one call to the HCALL. If the **-e** flag is specified, the summary gives the total elapsed time for all calls to the HCALL and the average, minimum, and maximum elapsed times for one call. Elapsed time is the wall-clock time between the start and end of an hypervisor call. The summary is presented in sorted order from most total processor time to least.

The second part of this section is called `Pending Hypervisor Calls Summary`. This part lists the Hypervisor Calls which have started but not completed. The time that is given is included in the HCALL time for the system and the various processors and is included in the hypervisor time for the pthread, thread, and process which issued the HCALL, but is not included in the processing time for the hypervisor call in the first part of this section. The pending call is also not included in the count given in the first part of this section.

Note:

1. Hypervisor call addresses are always given in hexadecimal. Pids and Tids are always given in decimal.
2. WPAR names are added in a System Hypervisor Calls Summary to identify the threads or processes if you specify the **-@** flag.

Pthread Calls Summary

The fourth major section of the report is the Pthread Calls Summary. This section summarizes the processing time spent in called pthread routines. For each pthread routine, identified by name, the summary gives the number of times the pthread routine was called and the total processor time for all calls, in milliseconds and as a percentage of total system processing time, for all processors in the trace. In addition, the summary gives the average, minimum and maximum times for one call to the pthread routine. If the **-e** flag is specified, the summary gives the total elapsed time for all calls to the pthread routine and the average, minimum and maximum elapsed times for one call. Elapsed time is the wall-clock time from when the process starts executing the pthread routine until the process exits the libpthreads code. The Summary is presented in sorted order from most total processor time to least.

The second part of this section is the Pending Pthread Calls Summary. This part lists the Pthread Calls which have started but not completed.

Note: WPAR names are added in a Pthread Calls Summary to identify threads or processes if you specify the **-@** flag.

System NFS Calls Summary

This major section of the report is the System NFS Calls Summary. This section summarizes the processing time spent in NFS operations. For each NFS operation, identified by operation name and NFS version, the summary gives the number of times the operation was called and the total processor time for all calls in milliseconds and as a percentage of total NFS operation time for all operations with the same NFS version. In addition, the summary gives the average, minimum and maximum times for one call to the operation. If the **-e** flag is specified, the summary gives the total elapsed time for all calls to the operation and the average, minimum and maximum times for one call. The total elapsed time is also given as a percentage of total NFS operation elapsed time for all operations with the same NFS version. Elapsed time is the wall-clock time from the operation dispatch entry hook until the operation dispatch exit hook. In all cases, the summary gives the count of operation calls as a percentage of total NFS operation calls for all operations with the same NFS version. The Summary is presented in numerical

order of the operation codes. The operations are presented in order of NFS Version. For NFS V4, the server operations are listed before the client operations.

The System NFS Calls Summary is followed by the Pending NFS Calls Summary. This part lists the NFS calls which have started but not completed. The time that is given is included in the NFS time for the system and the various processors and is included in the operation time for the thread and process which issued the NFS call, but is not included in the processing time for the NFS operation in the first part of this section. The pending call is also not included in the count given in the first part of this section.

Note: WPAR names are added in a System NFS Calls Summary to identify threads or processes if you specify the **-@** flag.

Flih Summary

The fifth major section of the report is the Flih Summary. This section summarizes the amount of time spent in first level interrupt handlers (Flih). The first part of the summary gives the total number of entries to each Flih in the trace, as well as the total processor time for all executions of the Flih by all processors in milliseconds. In addition, the summary gives the average, minimum and maximum times for one execution. Each Flih is identified by a system-defined Flih type and a corresponding Flih name, if known.

The second part is the same information broken down on a processor by processor basis. It is possible that not all Flihs which occurred on the system will have occurred on each processor, so the Global Flih list may not be the same as the Flih list for each processor.

The second part of this section may include the Pending Flih Summary. This is a list of the Flihs which have started but not completed. The time that is given is included in the FLIH time for the system and the affected processor, but is not included in the processing time for the Flih in both parts of this section. The pending Flih is also not included in the counts given in both parts of this section.

Slih Summary

The fifth major section of the report is the Slih Summary. This section summarizes the amount of time spent in second level interrupt handlers (Slih). The first part of the summary gives the total number of entries to each Slih in the trace, as well as the total processor time for all executions of the Slih by all processors in milliseconds. In addition, the summary gives the average, minimum and maximum times for one execution. Each Slih is identified by kernel address and Slih function or module name, if known.

The second part is the same information broken down on a processor by processor basis. It is possible that not all Slihs which occurred on the system will have occurred on each processor, so the Global Slih list may not be the same as the Slih list for each processor.

The second part of this section may include the Pending Slih Summary. This is a list of the Slihs which have started but not completed. The time that is given is included in the SLIH time for the system and the affected processor, but is not included in the processing time for the Slih in both parts of this section. The pending Slih is also not included in the counts given in both parts of this section.

Detailed Process Information

This section of the report is produced when the **-p** flag is specified. It gives detailed information about each process found in the trace. This information is as follows:

- The Process ID (Pid) for that process, the process name if known, and the WPAR name if you specify the **-@** flag.
- A count and a list of the Thread IDs (Tids) for that process.
- A count and a list of Pthread IDs (Ptid) for that process, if any.
- The time spent in application (user) mode, system call mode, and hypervisor mode is shown. For kprocs, the time spent in kernel mode and operation mode is shown instead.
- The detail of time spent in application mode, time spent in pthread operations, time spent in libpthreads dispatch, and time spent in vp_sleep. This is printed only if there are any Ptids for the process.

- Information on what Pthread calls were made by pthreads of this process. For NFS kprocs, information on which NFS Calls were made by threads of this process is shown instead. The **-e** flag also affects this output.
- Information on what hypervisor calls were made by threads of this process. The **-e** flag also affects this output.
- Information on what system calls were made by threads of this process. The **-e** flag also affects this output.

The processes are presented in sorted order from most combined application and syscall processing time to least.

Detailed Thread Information

This section of the report is produced when the **-t** flag is specified. It gives detailed information about each thread found in the trace. This information is as follows:

- The Thread ID (Tid) and Process ID (Pid) for that thread, the process name if known, and the WPAR name if you specify the **-@** flag.
- The time spent in application (user) mode, system call mode, and hypervisor call mode is shown. For kprocs, the time spent in kernel mode and operation mode is shown instead.
- Information on which system calls were made by this thread, including information on errors returned by the system calls if the **-s** flag was specified. For NFS kproc threads, information on which NFS Calls were made by this thread is shown instead. The **-e** flag also affects this output.
- Information on which hypervisor calls were made by this thread. The **-e** flag also affects this output.
- The processor affinity is the probability that, for any dispatch of the thread, the thread was dispatched to the same processor that it last executed on.
- The Dispatch Histogram shows the number of times the thread was dispatched to each CPU in the system.
- The total number of times the thread was dispatched (not including redispatches described in 7 below).
- The number of redispatches due to interrupts being disabled indicates that the same thread which just ran was dispatched again because that thread has set the interrupt mask to INTMAX. This is shown only if nonzero.
- The average dispatch wait time is the average elapsed time since the thread was last undispatched (i.e. average elapsed time since the thread last stopped executing).
- How many times each type of Flih occurred while this thread was executing. Some of these types may be caused by the thread (such as DSI or ISI) while other types (such as IO) are can occur when this thread just happens to be running and are not necessarily caused by the thread itself.

The threads are presented in sorted order from most combined application and syscall processing time to least.

Detailed Pthread Information

This section of the report is produced when the **-P** flag is specified. It gives detailed information about each pthread found in the trace. This information is as follows:

- The Pthread ID (Ptid) and Process ID (Pid) for that pthread, the process name if known, and the WPAR name if you specify the **-@** flag.
- The time spent in application (user) mode, kernel mode, and hypervisor mode is shown.
- Application time detail: time spent in pthread calls, pthread dispatch, vp_sleep (pthread idle), and other application time.
- Information on what system calls were made by this pthread, including information on errors returned by the system calls if the **-s** flag was specified. The **-e** flag also affects this output.
- Information on what hypervisor calls were made by this pthread. The **-e** flag also affects this output.
- Information on what Pthread calls were made by this pthread. The **-e** flag also affects this output.

- The processor affinity is the probability that, for any dispatch of the pthread, the pthread was dispatched to the same processor that it last executed on.
- The Dispatch Histogram for thread shows the number of times the pthread was dispatched to each CPU in the system.
- The total number of times the pthread was dispatched (not including redispatches described in 9 below).
- The number of redispatches due to interrupts being disabled indicates that the same pthread which just ran was dispatched again because that pthread has set the interrupt mask to INTMAX. This is shown only if non-zero.
- The average dispatch wait time is the average elapsed time since the pthread was last undispatched by the kernel dispatcher (that is, average elapsed time since the pthread last stopped executing).
- The thread affinity is the probability that, for any dispatch of the pthread, the pthread was dispatched to the same thread that it last executed on.
- The Dispatch Histogram for pthread shows the number of times the pthread was dispatched to each thread in the system.
- The total number of times the pthread was dispatched in libpthreads.
- The average dispatch wait time is the average elapsed time since the thread was last undispatched by the libpthreads dispatcher (that is, the average elapsed time since the thread last stopped executing).
- How many times each type of Flih occurred while this thread was executing. Some of these types may be caused by the thread (such as DSI or ISI) while other types (such as IO) are can occur when this thread just happens to be running and are not necessarily caused by the thread itself.

The pthreads are presented sorted by Pid-Ptid.

Files

Item	Description
/usr/bin/curt	Contains the **curt** command. Located in the **bos.perf.tools** fileset.

custom Command

Purpose

Enables users to customize X applications.

Syntax

custom [**-h** | **-e** *Browser* | [**-s** *ResourceFile*] [*Application*]]

Description

The **custom** command starts the customizing tool, which is used to customize various aspects of applications.

The customizing tool can change the look of an application. It provides a user-friendly way to add resource values to your **.Xdefaults** file. *Resources* are customizable items such as colors, fonts, and other attributes that allow you to customize resources of a client application. Each application has its own set of unique resources, which are listed in an **app-custom** file. The customizing tool describes the resources available for modification for an application and the possible resource values you can select.

Flags

Item	Description
-h	Provides command line help.
-e *Browser*	Calls one of the standalone browsers. Valid values for *Browser* are **color**, **font**, **cursor**, and **picture**.
-s *ResourceFile*	Specifies the resource file from which to load and save resource settings. If the **-s** flag is not specified, the default is to load the values from the resource database stored in the **RESOURCE_MANAGER** property on the X server. If this database does not exist, then **$HOME/.Xdefaults** is loaded.

Most standard X Toolkit command-line options are understood by the **custom** command.The following table lists the standard command-line options:

Standard Command-Line Options in **custom** command	
Option	**Information**
-bg	**Resource** *background **Value** Next argument **Sets** Background color
-background	**Resource** *background **Value** Next argument **Sets** Background color
-bd[1]	**Resource** *borderColor **Value** Next argument **Sets** Border color
-bordercolor[1]	**Resource** *borderColor **Value** Next argument **Sets** Color of border
-bw	**Resource** .borderWidth **Value** Next argument **Sets** Width of border in pixels

Standard Command-Line Options in **custom** command *(continued)*	
Option	**Information**
-borderWidth	**Resource** .borderWidth **Value** Next argument **Sets** Width of border in pixels
-display	**Resource** .display **Value** Next argument **Sets** Server to use
-fn[2]	**Resource** *font **Value** Next argument **Sets** Font name
-font[2]	**Resource** *font **Value** Next argument **Sets** Font name
-fg	**Resource** *foreground **Value** Next argument **Sets** Foreground color
-foreground	**Resource** *foreground **Value** Next argument **Sets** Foreground color
-geometry	**Resource** .geometry **Value** Next argument **Sets** Size and position

Standard Command-Line Options in **custom** command *(continued)*	
Option	**Information**
-iconic	**Resource** .iconic **Value** On **Sets** Start as an icon
-name	**Resource** .name **Value** Next argument **Sets** Name of application
-reverse	**Resource** *reverseVideo **Value** On **Sets** Reverse video
-rv	**Resource** *reverseVideo **Value** On **Sets** Reverse video
+rv	**Resource** *reverseVideo **Value** Off **Sets** No Reverse video
-selection- Timeout	**Resource** .selection-Timeout **Value** Next argument **Sets** Selection timeout
-synchronous	**Resource** *synchronous **Value** On **Sets** Synchronous debug mode

Standard Command-Line Options in **custom** command *(continued)*	
Option	**Information**
+synchronous	**Resource** *synchronous **Value** Off **Sets** Synchronous debug mode
-title	**Resource** .title **Value** Next argument **Sets** Title of application
-xrm	**Resource** value of argument **Value** Next argument **Sets** Depends on argument
-xnllanguage	**Resource** .xnlLanguage **Value** Next argument **Sets** Locale

Note:

1. These options often have no visible effect on AIXwindows applications if the AIXwindows Window Manager is running.
2. Motif applications do not generally respond to these options.
3. Resources beginning with an* (asterisk) set the resource of every widget in the application to the same value.
4. Resources that begin with a . (period) set the resources of only the application's top-level Shell widget.

Parameters

Item	Description
Application	Specifies the name or class of the application to customize.

Examples

1. To start the customizing tool and use prompts to choose the application to customize, type the following:

```
custom
```

2. To start the customizing tool to modify the **app-defaults** file of the **xcalc** application, type the following:

```
custom -s
/usr/lib/X11/app-defaults/XCalc xcalc
```

Resources

The customizing tool has the following application resources:

Item	Description
listOfApps	This resource is used to display the application names on the starting dialog. The application name and corresponding **app-custom** file must be listed in pairs with the following syntax: `Application:app-custom` `[,Application:app-custom]...` For example: `Custom.listOfApps:` `xclock:XClock,custom:Custom` You can specify a maximum of 100 applications.
colorEditor*rgbtxtPath	This resource specifies the full path name of the **rgb.txt** file that the X server uses to define named colors. The default value is **/usr/lib/X11/rgb.txt**, which is correct for an X server running on a display that is directly attached to your system.
windowSearchDepth	The customizing tool must determine the top-level shell window of the application. It startswith the root window and conducts a recursive search to a depth of three windows by default. This default can be changed using the **windowSearchDepth** resource.
timeout	The Instant Changes button is grayed out until communication with the application is established. The amount of time to wait for the application to contact the customizing tool is controlled by the **Custom*timeout** resource.
resourceFile	The resource file is where your resource changes are saved. The default is **$HOME/.Xdefaults**. The **-s** flag allows the user to override this value.

Item	Description
appCustomPath	This resource specifies where the customizing tool is to look for the **app-custom** file. The **appCustomPath** string consists of a series of possible file names separated by colons. Within each name, the following values can be substituted: **%N** Name of the **app-custom** file (usually the same as the class name of the application). **%T** **"app-custom"** **%L** Locale in which **custom** is running. **%l** Language part of the locale. **%t** Territory part of the locale. **%c** Codeset part of the locale. **%:** A : (colon). **%%** A % (percent sign). **$*envvar*** Value of the named environment variable. **${*envvar*}** Value of the named environment variable. **$$** A $ (dollar sign). The default value of **appCustomPath** is as follows: **$HOME/%L/%T/%N:** **$HOME/%T/%N:** **/usr/lib/X11/%L/%T/%N:** **/usr/lib/X11/%T/%N**
topEditHighlight, **bottomEditHighlight**, **foregroundEditHighlight**, **backgroundEditHighlight**	The Browser button is highlighted when a browser is called and unhighlighted when a browser is canceled. These resources set the highlight color for the top shadow, bottom shadow, foreground, and background of the Browser button.
pictureEditor*editor	You can edit the bitmap or pixmap by pressing the Edit Picture button on the Pictures browser window. The editor is a separate application that exists on your system. It is called on your behalf. The **Custom*pictureEditor*editor** resource determines which editor commands to choose from. This resource accepts a list of commands separated by \n's (backslash 'n's). The first command that identifies an existing program that the user has permission to execute is used. The file name in the Chosen Picture text field is passed as a parameter to the editor when it is invoked. The default setting for this resource is:

Item	Description

```
Custom*pictureEditor*editor:
/usr/dt/bin/dticon  -f
\n
/usr/lib/X11/bitmap
```

Note: The default editor, **/usr/dt/bin/dticon** only exists if the Common Desktop Environment (CDE) is installed. It edits both bitmaps (monochrome images) and pixmaps (color images). The **dticon** command accepts bitmaps stored in either the X Pixmap Version 2 Enhanced (XPM2) format which was used by the X Desktop (**xdt**) application shipped in AIXwindows Version 1.2.5, or X Pixmap Version 3 (XPM3) - a new XPG3 compliant format used by CDE. However, it requires pixmap images be stored in the XPM3 format. CDE has documented tools that can convert pixmaps from the XPM2 to the XPM3 format.

The **/usr/bin/X11/bitmap** command is an unsupported sample program that accepts bitmaps in either the XPM2 or XPM3 formats. It does not support pixmap editing. Be sure that the Bitmap app-defaults file has been installed in the **/usr/lib/X11/app-defaults** directory before invoking the **bitmap** command. If not, issue the following command in the **/usr/lpp/X11/Xamples/programs/bitmap** directory:

```
xmkmf;
make install
```

The following object names (and their class names) can be used to customize this tool:

```
custom (Custom)
  startupDialog_popup (XmDialogShell)
     startupDialog (XmSelectionBox)
  helpDialog_popup (XmDialogShell)
     helpDialog (XmForm)
  saveDialog_popup (XmDialogShell)
     saveDialog (XmSelectionBox)
  colorEditor_popup (XmDialogShell)
     colorEditor (XibmColorEditor)
  fontEditor_popup (XmDialogShell)
     fontEditor (XibmFontEditor)
  pictureEditor_popup (XmDialogShell)
     pictureEditor (XibmPictureEditor)
  cursorEditor_popup (XmDialogShell)
     cursorEditor (XibmCursorEditor)
  selectmanyEditor_popup (XmDialogShell)
     selectmanyEditor (XibmSelectManyEditor)
  filenameEditor_popup (XmDialogShell)
     filenameEditor (XmFileSelectionBox)
  mainWindow (XmMainWindow)
     menubar (XmRowColumn)
     form (XmForm)
        appClassLabel (XmLabel)
        appClass (XmLabel)
        groupMenuLabel (XmLabel)
        groupMenu (XmRowColumn)
        scrolledGroup (XmScrolledWindow)
           scrolledGroupForm (XmForm)
               (XmLabelGadget)
               TypeField (XmTextField)
               TypeButton (XmPushButton)
```

where *Type* can be one of the color, font, picture, cursor, selectmany, filename, selectone, string, or number data type values.

Exit Status

This command returns the following exit values:

Item	Description
0	Indicates successful completion.
>0	Indicates an error occurred.

Files

Item	Description
/usr/bin/X11	Is the path from which you run the **custom** command once the custom package is installed.
/usr/lib/X11/app-custom	Contains information about resources for individual applications.
/usr/lib/X11/*locale***/app-custom**	Contains information about resources for individual applications that is translated for specific locales.
/usr/lib/X11/app-defaults/Custom	Contains default settings for the Customizing Tool.
/usr/lib/X11/*locale***/app-defaults/Custom**	Contains default settings for the Customizing Tool in locales that require special settings.

cut Command

Purpose

Helps split the lines of a file.

Syntax

cut { **-b** *List* [**-n**] | **-c** *List* | **-f** *List* [**-s**] [**-d** *Character*] } [*File* ...]

Description

The **cut** command cuts bytes, characters, or fields from each line of a file and writes these bytes, characters, or fields to standard output. If you do not specify the *File* parameter, the **cut** command reads standard input.

You must specify either the **-b**, **-c**, or **-f** flag. The *List* parameter is a comma-separated, blank-separated, or hyphen-separated list of integer numbers (in increasing order). The hyphen separator indicates ranges. The following entries are some example *List* parameters which could refer to bytes, characters, or fields:

```
1,4,7
1-3,8
-5,10
3-
```

where `-5` is a short form for the first through fifth and `3-` is a short form for the third through last.

If using the **cut** command on fields, the length of the fields specified by the *List* parameter can vary from field to field and line to line. The position of the field delimiter character, such as a tab character, determines the length of a field.

You can also use the **grep** command to make horizontal cuts through a file and the **paste** command to put the files back together. To change the order of columns in a file, use the **cut** and **paste** commands.

Flags

Item	Description
-b *List*	Specifies byte positions. These byte positions ignore multibyte character boundaries unless the **-n** flag is also specified.
-c *List*	Specifies character positions. For example, if you specify `-c 1-72`, the **cut** command writes out the first 72 characters in each line of the file.
-d *Character*	Uses the character specified by the *Character* variable as the field delimiter when you specify the **-f** flag. You must put quotation marks around characters with special meaning to the shell, such as the space character.
-f *List*	Specifies a list of fields assumed to be separated in the file by a delimiter character, which is by default the tab character. For example, if you specify `-f 1,7`, the **cut** command writes out only the first and seventh fields of each line. If a line contains no field delimiters, the **cut** command passes them through intact (useful for table subheadings), unless you specify the **-s** flag.
-n	Suppresses splitting of multibyte characters. Use only with the **-b** flag. If the last byte of a character falls within the range denoted by the *List* variable of the **-b** flag, the character is written; otherwise, the character is excluded.
-s	Suppresses lines that do not contain delimiter characters. Use only with the **-f** flag.

Exit Status

This command returns the following exit values:

Item	Description
0	All input files were output successfully.
>0	An error occurred.

Examples

1. To display several fields of each line of a file, enter:

   ```
   cut  -f 1,5 -d : /etc/passwd
   ```

   This displays the login name and full user name fields of the system password file. These are the first and fifth fields (`-f 1,5`) separated by colons (`-d :`).

   For example, if the **/etc/passwd** file looks like this:

   ```
   su:*:0:0:User with special privileges:/:/usr/bin/sh
   daemon:*:1:1::/etc:
   bin:*:2:2::/usr/bin:
   sys:*:3:3::/usr/src:
   adm:*:4:4:System Administrator:/var/adm:/usr/bin/sh
   pierre:*:200:200:Pierre Harper:/home/pierre:/usr/bin/sh
   joan:*:202:200:Joan Brown:/home/joan:/usr/bin/sh
   ```

   The **cut** command produces:

   ```
   su:User with special privileges
   daemon:
   bin:
   sys:
   adm:System Administrator
   pierre:Pierre Harper
   joan:Joan Brown
   ```

2. To display fields using a blank separated list, enter:

```
cut -f "1 2 3" -d : /etc/passwd
```

The **cut** command produces:

```
su:*:0
daemon:*:1
bin:*:2
sys:*:3
adm:*:4
pierre:*:200
joan:*:202
```

Files

Item	Description
/usr/bin/cut	Contains the **cut** command.

cxref Command

Purpose

Creates a C and C++ program cross-reference listing.

Syntax

cxref [**-c**] [**-o** *File*] [**-q***Option*] [**-s**] [**-t**] [**-w** *Number*] [[**-D** *Name* [*=Definition*]] [**-I** *Directory*] [**-U** *Name*]] ... [**-Nd***Number*] [**-Nl***Number*] [**-Nn***Number*] [**-Nt***Number*] *File* ...

Description

The **cxref** command analyzes C and C++ program *File*s and creates a cross-reference table, using the **cpp** command to include **#define** directives in its symbol table. It writes to standard output a listing of all symbols in each file processed, either separately or in combination (see the **-c** flag). The formal parameters in a function definition are always listed; but if a function is only prototyped and not defined, the parameters are not listed. When a reference to a symbol is that symbol's declaration, an * (asterisk) precedes it.

Flags

Item	Description
-c	Displays a combined listing of the cross-references in all input files.
-o *File*	Directs the output to the specified *File*.
-s	Does not display the input file names.
-t	Makes the listing 80 columns wide.
-w *Number*	Makes the listing *Number* columns wide, where *Number* is a decimal integer greater than or equal to 51. If *Number* is less than 51, the listing will be 80 columns wide.
-**Nd***Number*	Changes the dimension table size to *Number*. The default is 2000.
-**Nl***Number*	Changes the number of type nodes to *Number*. The default is 8000.
-**Nn***Number*	Changes the symbol table size to *Number*. The default is 1500.
-**Nt***Number*	Changes the number of tree nodes to *Number*. The default is 1000.

In addition, the **cxref** command recognizes the following flags of the **cpp** command (macro preprocessor):

Item	Description
-D *Name*[=*Definition*]	Defines *Name* as in a **#define** directive. The default definition is 1.
-I *Directory*	Looks first in directory, then looks in the directories on the standard list for **#include** files with names that do not begin with a slash (/) (see the **cpp** command).
-U *Name*	Removes any initial definition of *Name*, where *Name* is a reserved symbol predefined by the preprocessor.
-q*Option*	Pass **-q***Option* to the preprocessor. For example, `-qmbcs` sets multibyte mode specified by the current locale, and `-qidirfirst` modifies the search order for files included with the **#include file_name** directive.

Examples

To provide a combined cross-reference listing of `stdin1.c` and `stdin2.c`, making the output 80 columns wide, enter:

```
cxref  -c  -t stdin1.c stdin2.c > output
```

Files

Item	Description
/usr/ccs/lib/xpass	Special version of C compiler first-pass.
/usr/ccs/bin/cxref	Contains the **cxref** command.

d

The following AIX commands begin with the letter d.

dacinet Command

Purpose

Administers security on TCP ports in CAPP/EAL4+ configuration.

Syntax

dacinet **aclflush**

dacinet **aclclear** *Service* | *Port*

dacinet **acladd** *Service* | [-] *addr* [/*prefix_length*] [u : *user* | *uid* | g : *group* | *gid*]

dacinet **acldel** *Service* | [-] *addr* [/*prefix_length*] [u : *user* | *uid* | g : *group* | *gid*]

dacinet **aclls** *Service* | *Port*

dacinet **setpriv** *Service* | *Port*

dacinet **unsetpriv** *Service* | *Port*

dacinet **lspriv**

Description

The **dacinet** command is used to administer security on TCP ports. See the Subcommands section for details of the various functions of **dacinet**.

Subcommands

Item	Description
acladd	Adds ACL entries to the kernel tables that hold access control lists used by the **dacinet** command. The syntax of the parameters for the **acladd** subcommand follow: [**-**]*addr*[**/***length*][**u**:*user*|*uid*| **g**:*group*|*gid*] The parameters are defined as follows: ***addr*** A DNS host name or an IPv4 (or IPv6) address. A "-" before the address means that this ACL entry is used to deny access rather than to allow access. ***length*** Indicates that *addr* is to be used as a network address rather than host address, with its first *length* bits taken from *addr*. **u:*user*|*uid*** Optional user identifier. If the *uid* is not specified, all users on the specified host or subnet are given access to the service. If supplied, only the specified user is given access. **g:*group*|*gid*** Optional group identifier. If the *gid* is not specified, all users on the specified host or subnet are given access to the service. If supplied, only the specified group is given access.

Item	Description
aclclear	Clears the ACL for specified service or port.
acldel	Deletes ACL entries from the kernel tables that hold access control lists used by the **dacinet** command. The **dacinet acldel** subcommand deletes an entry from an ACL only if it is issued with parameters that exactly match the ones that were used to add the entry to the ACL. The syntax of the parameters for the **acldel** subcommands is as follows: [**-**]*addr*[**/***length*][**u**:*user*|*uid*| **g**:*group*|*gid*] The parameters are defined as follows: ***addr*** A DNS host name or an IPv4 (or IPv6) address. A "-" before the address means that this ACL entry is used to deny access rather than to allow access. ***length*** Indicates that *addr* is to be used as a network address rather than host address, with its first *length* bits taken from *addr*. **u:*user*|*uid*** Optional user identifier. If the *uid* is not specified, all users on the specified host or subnet are given access to the service. If supplied, only the specified user is given access. **g:*group*|*gid*** Optional group identifier. If the *gid* is not specified, all users on the specified host or subnet are given access to the service. If supplied, only the specified group is given access.
aclflush	Clears all the ACLs defined in the system, rendering all TCP ports inaccessible to connection requests except from the root user on the host. It also clears privileged ports such that any process can bind to any port above 1024.
aclls	Lists the ACL for the specified service or port. The **dacinet aclls 0** lists the default ACL. For authentication processing, from a logical perspective, the default ACL is appended to the ACL for the service. If no entry on the ACL matches the user who is attempting a connection to the service, access is denied. If one or more entries exist, the first one on the list with a *user*|*group*@*host*|*subnet* that matches the connection requester determines the user's ability to connect to the service. It is thus possible to deny a service to a member of a group that has access to the service merely by adding a deny entry for that member before you add the allow entry for the group.
lspriv	Lists all the privileged services or ports that are not permanently privileged (that is, it lists only privileged services with port numbers above 1024).
setpriv	Makes the specified service or port privileged such that only a process with superuser privileges might bind to the port and offer a service on that port. Ports below 1024 are ignored as they are permanently privileged.
unsetpriv	Makes the specified service or port unprivileged such that any process might bind to it. Any process might also bind to any port in the current ephemeral port range, regardless of whether that port is marked as privileged.

Files

Item	Description
`/usr/sbin/dacinet`	Contains the **dacinet** command.

dadmin Command

Purpose

Used to query and modify the status of the DHCP server.

Syntax

dadmin [**-?**] [**-v**] [**-h** *Hostname*] [**-n** *interval*] [**-f**] **-d** *IpAddress* | [**-x**] **-i** | [**-x**] **-s** | **-t** **on**|**off**|*Value* | **-q** *IpAddress* | **-r** *IpAddress* | **-p** *IpAddress* | **-c** *ClientId*

Description

The **dadmin** command enables the DHCP administrator to query and modify the state of DHCP server databases. It gives the administrator the ability to query the DHCP server, locally or remotely, for the status of an IP address, query for a pool of IP addresses, query for a client, delete an IP address mapping, refresh the server, and change the server's tracing level.

The **dadmin** command is compatible with an earlier version of DHCP servers to list their IP address status and refresh.

When querying for an IP address information, the **dadmin** command returns the IP address's status. And depending on the IP address's status, the **dadmin** command might return the lease duration, start lease time, last leased time, whether the server supports DNS, a record update for this IP address, and the client identifier that is mapped to this IP address.

When querying for a client information, the **dadmin** command returns the client's IP address and IP address status, the last time the client was given any IP address, the host name and domain name that are used by the client, whether the server supports DNS, and a record update for this IP address.

When you modify the server tracing level, the **dadmin** command sets and returns the server tracing level in the form of a tracing mask. This mask represents a bit string where each bit represents whether a specific log item is being traced by the server (see "DHCP Server Configuration File" in the online documentation). From least significant to most significant order, these log items are LOG_NONE, LOG_SYSERR, LOG_OBJERR, LOG_PROTOCOL and LOG_PROTERR (same value), LOG_WARN, AND LOG_CONFIG (same value), LOG_EVENT, and LOG_PARSEERR (same value), LOG_ACTION, LOG_INFM, LOG_ACNTING, LOG_STAT, LOG_TRACE, LOG_START, and LOG_RTRACE.

Note: LOG_START cannot be disabled. It implies a mask range from 0x0800 through 0x1FFF.

Flags

Item	Description
-c *ClientId*	Returns the status for a specific client that might be known to the DHCP server. *ClientId* represents the client identifier that a DHCP client used to identify itself, or the field can either be specified as hexadecimal characters only, or in the TYPE-STRING representation that is used by the DHCP server.
-d *IpAddress*	Deletes the lease information that is associated with IP address *IpAddress*. As a result, the address is moved to the FREE state and be available for binding again.
-f	To be used with the **-d** flag. The **-f** flag forces the deletion of the address without any prompting. Deletes the lease information that is associated with IP.
-h *Hostname*	Used to specify the destination DHCP server. *Hostname* can either be a name or IP address.
-i	Reinitializes the DHCP server. This flag signals the server to sync its databases and restarts by rereading the configuration file.
-n *interval*	Displays server statistics, summaries, and any requested intervals.

Item	Description
-p *IpAddress*	Returns the status of each address in a subnet. *IpAddress* is used to identify the subnet to a list.
-q *IpAddress*	Returns the status of a specific IP address.
-r *IpAddress*	Puts the IP address in the `Free` state.
-s	Returns the status of each address in the DHCP server's configured pools.
-t on|off| *Value*	Changes the tracing level of the DHCP server. Trace values are reported in a hexadecimal format that represents the tracing mask in use on the server. *Value* can be specified as either a decimal or hexadecimal format. The keywords **on** and **off** enable or disable a single bit at a time in the tracing mask.
-v	Runs the command in verbose mode.
-x	Use Version 1 of the **dadmin** protocol. The **-x** flag is used to connect to previous release DHCP servers and is only valid for the **-i** and **-s** flags. Follow with 6 when you connect with DHCPv6 server.
-?	Displays the usage syntax.

Exit Status

Item	Description
0	Successful completion.
>0	An error occurred.

Security

To secure connections from the **dadmin** clients, the DHCP server allows connections only from the server itself or from remote systems that are included in the superuser's `.rhosts` file. To prevent ordinary users from modifying the DHCP server's address mappings, the administrator must ensure that the execution of the **dadmin** command is limited to the proper users on those systems that are allowed access.

Files

Item	Description
`/usr/sbin/dadmin`	Contains the **dadmin** command.

date Command

Purpose

Displays or sets the date or time.

Syntax

To Set the Date and Time as Root User

/usr/bin/date [**-n**] [**-u**] [*Date*] [**+***FieldDescriptor* ...]

To Display the Date and Time

/usr/bin/date [**-u**] [**+***FieldDescriptor* ...]

To adjust the Time in Seconds as root User

/usr/bin/date [**-a** [**+** | **-**]*sss*[*.fff*]

Description

Attention: Do not change the date when the system is running with more than one user.

The **date** command writes the current date and time to standard output if called with no flags or with a flag list that begins with a **+** (plus sign). Otherwise, it sets the current date. Only a root user can change the date and time. The **date** command prints the usage message on any unrecognized flags or input.

The following formats can be used when you set the date with the *Date* parameter:

- *mmddHHMM*[*YYyy*]
- *mmddHHMM*[*yy*]

The variables to the *Date* parameter are defined as follows:

Item	Description
mm	Specifies the month number.
dd	Specifies the number of the day in the month.
HH	Specifies the hour in the day by using a 24-hour clock.
MM	Specifies the minute number.
YY	Specifies the first 2 digits of the year. **Note:** If you do not specify the first 2 digits of the year, values in the range 70 - 99 refer to the 20th century, 1970 - 1999 inclusive. Similarly, values in the range 00 - 37 refer to years in the 21st century, 2000 - 2037 inclusive.
yy	Specifies the last 2 digits of the year. **Note:** The **date** command accepts a 4-digit year as input. For example, if a 4-digit year is specified, the **date** command tries to set the year to *YYyy* and fails for values that are out of range (less than 1970 and greater than 2105). For years in the range 2038 - 2105, specify the year in the *yyyy* format.

The current year is used as the default value when the year is not specified. The system operates in Coordinated Universal Time (CUT).

If you follow the **date** command with a **+** (plus sign) and a field descriptor, you can control the output of the command. You must precede each field descriptor with a **%** (percent sign). The system replaces the field descriptor with the specified value. Enter a literal % as %% (two percent signs). The **date** command copies any other characters to the output without change. The **date** command always ends the string with a new-line character.

Flags

Item	Description
-a [**+** | **-**] *sss*[*.fff*]	Slowly adjusts the time by *sss.fff* seconds (*fff* represents fractions of a second). This adjustment can be positive or negative. The system's clock is speeded up or slowed down until it is drifted by the number of seconds specified.
-n	Does not set the time globally on all systems in a local area network that have their clocks that are synchronized.
-u	Displays or sets the time in Coordinated Universal Time (CUT).

Field Descriptors

Item	Description
%a	Displays the locale's abbreviated weekday name.
%A	Displays the locale's full weekday name.
%b	Displays the locale's abbreviated month name.
%B	Displays the locale's full month name.
%c	Displays the locale's appropriate date and time representation (default).
%C	Displays the first 2 digits of the four-digit year as a decimal number (00-99). A year is divided by 100 and truncated to an integer.
%d	Displays the day of the month as a decimal number (01-31). In a two-digit field, a 0 is used as leading space fill.
%D	Displays the date in the format equivalent to **%m/%d/%y**.
%e	Displays the day of the month as a decimal number (1-31). In a two-digit field, a blank space is used as leading space fill.
%h	Displays the locale's abbreviated month name (a synonym for **%b**).
%H	Displays the hour (24-hour clock) as a decimal number (00-23).
%I	Displays the hour (12-hour clock) as a decimal number (01-12).
%j	Displays the day of year as a decimal number (001-366).
%k	Displays the 24-hour-clock hour clock as a right-align, space-filled number (0 - 23).
%m	Displays the month of year as a decimal number (01-12).
%M	Displays the minutes as a decimal number (00-59).
%n	Inserts a new-line character.
%p	Displays the locale's equivalent of either AM or PM.
%r	Displays 12-hour clock time (01-12) using the AM-PM notation; in the POSIX locale, it is equivalent to **%I:%M:%S %p**.
%S	Displays the seconds as a decimal number (00 - 59).
%s	Displays the number of seconds since January 1, 1970, Coordinated Universal Time (CUT).
%t	Inserts a <tab> character.
%T	Displays the 24-hour clock (00-23) in the format equivalent to `HH:MM:SS`.
%u	Displays the weekday as a decimal number in the range 1-7 (Sunday = 7). Refer to the **%w** field descriptor.
%U	Displays week of the year (Sunday as the first day of the week) as a decimal number [00 - 53]. All days in a new year that precede the first Sunday are considered to be in week 0.
%V	Displays the week of the year as a decimal number in the range 01-53 (Monday is used as the first day of the week). If the week that contains January 1 has four or more days in the new year, then it is considered week 01. Otherwise, it is week 53 of the previous year.
%w	Displays the weekday as a decimal number in the range 0-6 (Sunday = 0). Refer to the **%u** field descriptor.
%W	Displays the week number of the year as a decimal number (00-53) counting Monday as the first day of the week.
%x	Displays the locale's appropriate date representation.
%X	Displays the locale's appropriate time representation.
%y	Displays the last 2 numbers of the year (00-99).

Item	Description
%Y	Displays the four-digit year as a decimal number.
%Z	Displays either the time-zone name or time-zone offset as applicable. No characters are displayed, if the time-zone is not determined.
%%	Displays a % (percent sign) character.

Modified Field Descriptors

The **%E** and **%O** field descriptors can be modified to indicate a different format or specification, as described in LC_TIME Category for the Locale Definition Source File Format in *Files Reference*. If the corresponding keyword (see the `era`, `era_year`, `era_d_fmt`, and `alt_digits` keywords) is not specified or not supported for the current locale, the unmodified field descriptor value is used.

Item	Description
%Ec	Displays the locale's alternative appropriate date and time representation.
%EC	Displays the name of the base year (or other time period) in the locale's alternative representation.
%Ex	Displays the locale's alternative date representation.
%EX	Displays the locale's alternative time representation.
%Ey	Displays the offset from the **%EC** field descriptor (year only) in the locale's alternative representation.
%EY	Displays the full alternative year representation.
%Od	Displays the day of the month using the locale's alternative numeric symbols.
%Oe	Displays the day of the month using the locale's alternative numeric symbols.
%OH	Displays the hour (24-hour clock) using the locale's alternative numeric symbols.
%OI	Displays the hour (12-hour clock) using the locale's alternative numeric symbols.

Item	Description
%Om	Displays the month using the locale's alternative numeric symbols.
%OM	Displays minutes using the locale's alternative numeric symbols.
%OS	Displays seconds using the locale's alternative numeric symbols.
%Ou	Displays the weekday as a number in the locale's alternative representation (Monday=1).
%OU	Displays the week number of the year using the locale's alternative numeric symbols. Sunday is considered the first day of the week.
%OV	Displays the week number of the year using the locale's alternative numeric symbols. Monday is considered the first day of the week.
%Ow	Displays the weekday as a number in the locale's alternative representation (Sunday =0).
%OW	Displays the week number of the year using the locale's alternative numeric symbols. Monday is considered the first day of the week.
%Oy	Displays the year (offset from %C) in alternative representation.

Exit Status

This command returns the following exit values:

Item	Description
0	The date was written successfully.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display current date and time, enter:

   ```
   date
   ```

2. To set the date and time, enter:

   ```
   date 0217142590
   ```

   For a system that uses CST as its time zone, this command sets the date and time to Sat Feb 17 14:25:00 CST 1990.

   Note: You must have root authority to change the date and time.

3. To display the date and time in a specified format, enter:

   ```
   date +"%r %a %d %h %y (Julian Date: %j)"
   ```

   This command displays the date that is shown in Example 2 as:

   ```
   02:25:03 PM Fri 17 Feb 90 (Julian Date: 048)
   ```

Environment Variables

The following environment variables affect the execution of the **date** command.

Item	Description
LANG	Determines the locale to use when both *LC_ALL* and the corresponding environment variable (beginning with *LC_*) do not specify a locale.
LC_ALL	Determines the locale to be used to override any values for locale categories that are specified by the setting of *LANG* or any environment variable beginning with *LC_*.
LC_CTYPE	Determines the locale for the interpretation of sequences of bytes of text data as characters (for example, single versus multibyte character in an argument).
LC_MESSAGES	Determines the language in which messages are to be written.
LC_TIME	Determines the contents of date and time strings that are written by the **date** command.
NLSPATH	Determines the location of message catalogs for the processing of *LC_MESSAGES*.
TZ	Specifies the time zone in which the time and date are written, unless the **-u** flag is specified. If the *TZ* variable is not set and the **-u** flag is not specified, an unspecified system default time zone is used.

dbts Command

Purpose

Debugs a thin server.

Syntax

dbts [**-v**] *ThinServer*

Description

The **dbts** command starts a thin server in the debug mode. The command checks if the thin server was previously started in the debug mode by searching for a debug boot image that is created for the thin server. If none is found, the common image that the thin server is using is cloned and a debug boot image is created from the clone to allow the thin server to boot into debug mode. The debug boot image clone uses the following naming convention:

```
{COSI name}_{thin server name}-debug
```

After the thin server is finished using the debug common image, the **swts** command must be run to switch the thin server to a different common image. The **rmcosi** command removes the debug common image that is created from the **dbts** command. The **dbts** command can run on either a NIM master or a thin server.

Flags

Item	Description
-v	Enables verbose debug output while the **dbts** command runs.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Security

Access Control: You must have root authority to run the **dbts** command.

Examples

1. To debug boot a thin server named `lobo` that is using a common image named `cosi1`, enter:

   ```
   dbts lobo
   ```

   A debug boot image named `cosi1_lobo-debug` is created to boot `lobo` into debug mode.

Location

`/usr/sbin/dbts`

Files

Item	Description
/etc/niminfo	Contains variables that are used by NIM.

dbx Command

Purpose

Provides an environment to debug and run programs.

Syntax

dbx [**-a** *ProcessID*] [**-B** *DebugFile*] [**-c** *CommandFile*] [**-I** *Directory*] [**-E** *DebugEnvironment*] [**-p** *oldpath=newpath:.../ pathfile*] [**-u**] [**-F**] [**-L**] [**-r**] [**-x**] [**-v**] [**-C** *CoreFile* | *ObjectFile* [*CoreFile*]]

Description

The **dbx** command provides a symbolic debug program for C, C++, and Fortran programs, allowing you to carry out the following operations:

- Examine object and core files.
- Provide a controlled environment for running a program.
- Set breakpoints at selected statements or run the program one line at a time.
- Debug using symbolic variables and display them in their correct format.

The *ObjectFile* parameter is an object (executable) file produced by a compiler. Use the **-g** (generate symbol table) flag when compiling your program to produce the information the **dbx** command needs.

Note: The **-g** flag of the **cc** command must be used when the object file is compiled. If the **-g** flag is not used or if symbol references are removed from the **xcoff** file with the **strip** command, the symbolic capabilities of the **dbx** command are limited. In addition, do not use the -O compiler option to optimize an executable that you plan to debug with **dbx**. Optimization rearranges the code and compromises the debug data, further limiting the value of debugging the executable program with the **dbx** command.

If the **-c** flag is not specified, the **dbx** command checks for a **.dbxinit** file in the user's **$HOME** directory. It then checks for a **.dbxinit** file in the user's current directory. If a **.dbxinit** file exists in the current directory, that file overrides the **.dbxinit** file in the user's **$HOME** directory. If a **.dbxinit** file exists in the user's **$HOME** directory or current directory, that file subcommands run at the beginning of the debug session. Use an editor to create a **.dbxinit** file.

If *ObjectFile* is not specified, then **dbx** asks for the name of the object file to be examined. The default is **a.out**. If the **core** file exists in the current directory or a *CoreFile* parameter is specified, then **dbx** reports the location where the program faulted. Variables, registers, and memory held in the core image might be examined until execution of *ObjectFile* begins. At that point the **dbx** debug program prompts for commands.

The **-B** flag is used to specify an alternative object file or a separate .stab file containing debug information about startup. The alternative object file can be specified only while attaching to a process. The debug information is read from this alternate object file or the .stab debug file instead of the disk copy of the running process. This alternate object file must be the unstripped copy of the original object file; otherwise, it will be ignored. Use the **-B** flag when the size of the debug section is large. Use the stripped copy of the object file while running and an unstripped copy while debugging. The .stab debug file can be generated through the **–bstabsplit** linker option. If the **-B** flag is not specified for a stabsplit executable the **dbx** command will try to acquire the corresponding .stab file from the executable directory.

Expression Handling

The **dbx** program can display a wide range of expressions. You can specify expressions in the **dbx** debug program with C syntax, with some Fortran extensions.

The following operators are valid in the debug program:

Item	Description
***** (asterisk) or **^** (caret)	Denotes indirection or pointer dereferencing.
[ ] (brackets) or **( )** (parentheses)	Denotes subscript array expressions.
. (period)	Use this field reference operator with pointers and structures. This operator makes the C operator -> (arrow) unnecessary, although it is allowed.
& (ampersand)	Gets the address of a variable.
.. (two periods)	Separates the upper and lower bounds when specifying a subsection of an array. For example: **n[1..4]**.

The following types of operations are valid in expressions in the debug program:

Item	Description
Algebraic	**=**, **-**, *****, **/** (floating division), **div** (integral division), **mod**, **exp** (exponentiation)
Bitwise	**-**, **I**, **bitand**, **xor**, **~**. **<<**, **>>**
Logical	**or**, **and**, **not**, **II**, **&&**
Comparison	**<**, **>**, **<=**, **>=**, **< >** or **!=**, **=** or **==**
Other	**(typename),sizeof**

Logical and comparison expressions are allowed as conditions in **stop** and **trace**.

Expression types are checked. You override an expression type by using a renaming or casting operator. The three forms of type renaming are *Typename(Expression)*, *Expression*|*Typename*, and *(Typename) Expression*. The following is an example where the *x* variable is an integer with value 97:

```
(dbx) print x
97
(dbx) print char (x), x \ char, (char) x, x
'a' 'a' 'a' 97
```

Command Line Editing

The **dbx** command provides a command line editing feature similar to the features provided by Korn Shell. **vi** mode provides **vi-like** editing features, while **emacs** mode gives you controls similar to **emacs**.

These features can be turned on by using **dbx** subcommand **set -o** or **set edit**. To turn on vi-style command line editing, you would type the subcommand **set edit vi** or **set -o vi**.

You can also use the **EDITOR** environment variable to set the editing mode.

The **dbx** command saves the history of commands, which are entered in the command line, in the **.dbxhist** history file. If the **DBXHISTFILE** environment variable is not set, the **$HOME/.dbxhist** history file is used.

By default, **dbx** saves the text of the last 128 commands entered. The **DBXHISTSIZE** environment variable can be used to increase this limit.

Flags

Item	Description
-a *ProcessID*	Attaches the debug program to a process that is running. To attach the debug program, you need authority to send signals to this process. Use the **ps** command to determine the process ID. If you have permission, the **dbx** program interrupts the process using the `ptrace` system call to send a SIGTRAP signal to the process, which cannot ignore the SIGTRAP signal. It then determines the full name of the object file, reads in the symbolic information, and prompts for commands.
-B*DebugFile*	This flag allows you to specify an alternative debug file on startup.
-c *CommandFile*	Runs the **dbx** subcommands in the file before reading from standard input. The specified file in the **$HOME** directory is processed first; then the file in the current directory is processed. The command file in the current directory overrides the command file in the **$HOME** directory. If the specified file does not exist in either the **$HOME** directory or the current directory, a warning message is displayed. The **source** subcommand can be used once the **dbx** program is started.
-C *CoreFile*	Analyzes the core file without specifying the object file. In this case, the **dbx** command uses the object file mentioned in the core file if it exists in the current directory and matches with the core file. Otherwise, it proceeds further without the object file. This flag is ignored if you use it after the **-r** flag or the **-a** flag.
-E *DebugEnvironment*	Specifies the environment variable for the debug program.
-p *oldpath=newpath:...| pathfile*	Specifies a substitution for library paths when examining core files or attaching to a process, in the format *oldpath=newpath*. The *oldpath* variable specifies the value to be substituted (as stored in the core file or the loader section of the process when attaching). The *newpath* variable specifies what it is to be replaced with. The *oldpath* variable and *newpath* variable can be complete paths, partial paths, relative paths, or absolute paths. Multiple substitutions are separated by colons. Alternatively, the **-p** flag might specify the name of a file from which mappings in the previously described format are to be read. Only one mapping per line is allowed when mappings are read from a file. If you use the **-p** flag when attaching to a process, the debug information is read from the substituted path files. The path files must match the running copy of the library.
-F	Can be used to turn off the lazy read mode and make the **dbx** command read all symbols at startup time. By default, lazy reading mode is on: it reads only required symbol table information about initiation of **dbx** session. In this mode, **dbx** does not read local variables and types whose symbolic information is not read. Therefore, commands such as **whereis i** might not list all instances of the local variable **i** in every function.
-L	Keep linkage symbols.

Item	Description
-I *Directory*	(Uppercase i) Includes directory specified by the *Directory* variable in the list of directories searched for source files. The default is to look for source files in the following directories: • The directory the source file was located in when it was compiled. This directory is searched only if the compiler placed the source path in the object. • The current directory. • The directory where the program is currently located.
-r	Runs the object file immediately. If it terminates successfully, the **dbx** debug program is exited. Otherwise, the debug program is entered and the reason for termination is reported. **Note:** Unless **-r** is specified, the **dbx** command prompts the user and waits for a command.
-u	Causes the **dbx** command to prefix file name symbols with an @ (at sign). This flag reduces the possibility of ambiguous symbol names.
-v	Causes the **dbx** command to skip the validity checking of the core file. This flag allows you to analyze the valid sections of the core file even if some sections are not valid.
-x	Prevents the **dbx** command from stripping _ (trailing underscore) characters from symbols originating in Fortran source code. This flag allows **dbx** to distinguish between symbols which are identical except for an underscore character, such as xxx and xxx_.

Examples

1. The following example explains how to start the **dbx** debug program simultaneously with a process. The example uses a program called **samp.c**. This C program is first compiled with the **-g** flag to produce an object file that includes symbolic table references. In this case, the program is named **samp**:

   ```
   $ cc -g samp.c -o samp
   ```

   When the program **samp** is run, the operating system reports a bus error and writes a core image to your current working directory as follows:

   ```
   $ samp
   Bus Error - core dumped
   ```

   To determine the location where the error occurred, enter:

   ```
   $ dbx samp
   ```

   The system returns the following message:

   ```
   dbx version 3.1
   Type 'help' for help.
   reading symbolic information . . . [
   using memory image in core]
     25   x[i] = 0;
   (dbx) quit
   ```

2. This example explains how to attach **dbx** to a process. This example uses the following program, **looper.c**:

   ```
   main()
   {
   ```

```
        int i,x[10];

        for (i = 0; i < 10;);
}
```

The program never terminates because **i** is never incremented. Compile **looper.c** with the **-g** flag to get symbolic debugging capability:

```
$ cc -g looper.c -o looper
```

Run **looper** from the command line and perform the following steps to attach **dbx** to the program while it is running:

a. To attach **dbx** to **looper**, you must determine the process ID. If you did not run **looper** as a background process, you must have another Xwindow open. From this Xwindow, enter:

   ```
   ps -u UserID
   ```

   where *UserID* is your login ID. All active processes that belong to you are displayed as follows:

   ```
   PID     TTY      TIME    COMMAND
   68      console   0:04    sh
   467     lft3     10:48    looper
   ```

   In this example the process ID associated with **looper** is 467.

b. To attach **dbx** to **looper**, enter:

   ```
   $ dbx -a 467
   ```

   The system returns the following message:

   ```
   Waiting to attach to process 467 . . .
   Successfully attached to /tmp/looper.
   dbx is initializing
   Type 'help' for help.
   reading symbolic information . . .

   attached in main at line 5
   5     for (i = 0; i < 10;);
   (dbx)
   ```

   You can now query and debug the process as if it was originally started with **dbx**.

3. To add directories to the list of directories to be searched for the source file of an executable file **objfile**, you can enter:

   ```
   $dbx -I /home/user/src -I /home/group/src
   objfile
   ```

   The **use** subcommand might be used for this function once **dbx** is started. The **use** command resets the list of directories, whereas the **-I** flag adds a directory to the list.

4. To use the **-r** flag, enter:

   ```
   $ dbx -r samp
   ```

   The system returns the following message:

   ```
   Entering debug program . . .
   dbx version 3.1
   Type 'help' for help.
   reading symbolic information . . .
   bus error in main at line 25
     25   x[i] = 0;
   (dbx) quit
   ```

   The **-r** flag allows you to examine the state of your process in memory even though a core image is not taken.

5. To specify the environment variables for the debug program, enter:

```
dbx -E LIBPATH=/home/user/lib -E LANG=Ja_JP objfile
```

6. To specify alternative object file and libraries while attaching to the process, enter:

```
dbx –a 467 –B debug_samp –p /usr/lib/=./dir/debug_libs/
```

7. To specify the separated debug file at startup, enter:

```
 dbx –B /usr/debug_samp.stab debug_samp
```

dbx Subcommands

Note: The subcommands can be used only while running the **dbx** debug program.

Item	Description
/	Searches forward in the current source file for a pattern.
?	Searches backward in the current source file for a pattern.
addcmd	Adds the **dbx** subcommands to the specified event numbers.
alias	Creates aliases for the **dbx** subcommands.
assign	Assigns a value to a variable.
attribute	Displays information about all or selected attributes objects.
call	Runs the object code associated with the named procedure or function.
case	Changes how the **dbx** debug program interprets symbols.
catch	Starts trapping a signal before that signal is sent to the application program.
clear	Removes all stops at a particular source line.
cleari	Removes all breakpoints at an address.
condition	Displays information about all or selected condition variables.
cont	Continues application program execution from the current stopping point until the program finishes or another breakpoint is encountered.
corefile	Displays high-level data about a core file.
coremap	Displays the mapping of a particular address space region.
delcmd	Deletes **dbx** subcommands associated with the specified event number.
delete	Removes the traces and stops corresponding to the specified event numbers and `tskip` counts for a thread.
detach	Continues execution of application and exits the debug program.
disable	Disables the traces and stops corresponding to the specified event numbers.
display memory	Displays the contents of memory.
down	Moves the current function down the stack.
dump	Displays the names and values of variables in the specified procedure.
edit	Starts an editor on the specified file.
enable	Enables the traces and stops corresponding to the specified event numbers.
fd	Displays file descriptor information.
file	Changes the current source file to the specified file.

Item	Description
frame	Changes the current function to the function corresponding to the specified stack frame number.
func	Changes the current function to the specified procedure or function.
goto	Causes the specified source line to be the next line run.
gotoi	Changes the program counter address.
handler	Displays information about pthreads atfork or cancelation cleanup handlers.
help	Displays help information for **dbx** subcommands or topics.
ignore	Stops trapping a signal before that signal is sent to the application program.
kthread	Displays information about kernel threads.
limitbp	Limits the number of times that a breakpoint can be run.
list	Displays lines of the current source file.
listi	Lists instructions from the application program.
malloc	Displays information about the program usage of the malloc subsystem.
map	Displays information about load characteristics of the application.
move	Changes the next line to be displayed.
multproc	Enables or disables multiprocess debugging.
mutex	Displays information about all or selected mutexes.
next	Runs the application program up to the next source line.
nexti	Runs the application program up to the next machine instruction.
onceblock	Displays information about once blocks.
plugin	Invokes a plug-in subcommand or displays the names of available plug-ins.
pluginload	Loads a plug-in.
pluginunload	Unloads a plug-in.
print	Prints the value of an expression or runs a procedure and prints the return code of that procedure.
printbp	Prints the number of times that a breakpoint is run.
proc	Displays information about the process.
prompt	Changes the **dbx** command prompt.
quit	Stops the **dbx** debug program.
registers	Displays the values of all general-purpose registers, system-control registers, floating-point registers, and the current instruction register.
rerun	Begins execution of an application with the previous arguments.
resource	Displays information about resources owned or waited on by pthreads.
return	Continues running the application program until a return to the specified procedure is reached.
rwlock	Displays information about the rwlocks.
run	Begins running an application.
screen	Opens an Xwindow for **dbx** command interaction.
set	Defines a value for a **dbx** debug program variable.

Item	Description
sh	Passes a command to the shell to be run.
skip	Continues running the application program from the current stopping point.
source	Reads **dbx** subcommands from a file.
status	Prints the details about a breakpoint. It also displays the active trace, **stop** subcommands, and the remaining thread `tskip` counts.
step	Runs one source line.
stepi	Runs one machine instruction.
stophwp	Sets a hardware watchpoint stop.
stop	Stops running the application program.
stopi	Sets a stop at a specified location.
thdata	Displays thread-specific data.
thread	Displays and controls threads.
tls	Displays TLS initialization template information.
tm_status	Displays and interprets the value that is stored in the *$texasr* variable.
tnext	Runs a thread up to the next source line.
tnexti	Runs a thread up to the next machine instruction.
trace	Prints tracing information.
tracehwp	Sets a hardware watchpoint trace.
tracei	Turns on tracing.
tskip	Skips breakpoints for a thread.
tstep	Runs a thread for one source line.
tstepi	Runs a thread for one machine instruction.
tstop	Sets a source-level breakpoint stop for a thread.
tstophwp	Sets a thread-level hardware watchpoint stop.
tstopi	Sets an instruction-level breakpoint stop for a thread.
ttrace	Sets a source-level trace for a thread.
ttracehwp	Sets a thread-level hardware watchpoint trace.
ttracei	Sets an instruction-level trace for a thread.
unalias	Removes an alias.
unset	Deletes a variable.
up	Moves the current function up the stack.
use	Sets the list of directories to be searched when looking for source files.
whatis	Displays the declaration of application program components.
where	Displays a list of active procedures and functions.
whereis	Displays the full qualifications of all the symbols whose names match the specified identifier.
which	Displays the full qualification of the specified identifier.

/ Subcommand

/ [*RegularExpression* [**/**]]

The **/** subcommand searches forward in the current source file for the pattern specified by the *RegularExpression* parameter. Entering the **/** subcommand with no arguments causes **dbx** to search forward for the previous regular expression. The search wraps around the end of the file.

Examples

1. To search forward in the current source file for the number 12, enter:

```
/ 12
```

2. To repeat the previous search, enter:

```
/
```

See the **?** (search) subcommand and the **regcmp** subroutine.

? Subcommand

? [*RegularExpression* [**?**]]

The **?** subcommand searches backward in the current source file for the pattern specified by the *RegularExpression* parameter. Entering the **?** subcommand with no arguments causes the **dbx** command to search backwards for the previous regular expression. The search wraps around the end of the file.

Examples

1. To search backward in the current source file for the letter z, enter:

```
?z
```

2. To repeat the previous search, enter:

```
?
```

See the **/** (search) subcommand and the **regcmp** subroutine.

addcmd Subcommand

addcmd { *Number*... | **all** } "*commands_string*"

The **addcmd** subcommand adds **dbx** subcommands to the specified event. This specified event is run whenever the breakpoint, tracepoint, or watchpoint corresponding to the event is executed. The **dbx** subcommands can be specified through the "*commands_string*" parameter, which is a group of dbx subcommands separated by a semicolon (;). The event to which the **dbx** subcommands are to be added can be specified through the *Number* parameter, or the **dbx** subcommands can be added to all events by using the `all` flag.

Flags

Item	Description
all	Adds **dbx** subcommands to all the events.

Examples

1. To add the `where` subcommand to event number 1, enter:

```
addcmd 1 "where"
```

2. To add the `registers` subcommand to event number 2, enter:

```
addcmd 2 "registers"
```

3. To add the `where` and `registers` subcommands to event number 3, enter:

```
addcmd 3 "where;registers"
```

See **clear** subcommand, the **delcmd** subcommand, the **delete** subcommand, **disable** subcommand, **enable** subcommand, the **stop** subcommand, the **status** subcommand, and the **trace** subcommand. Also see Setting and Deleting Breakpoints in in *General Programming Concepts: Writing and Debugging Programs*.

alias Subcommand

alias [*Name* [[(*Arglist*)] *String* | *Subcommand*]]

The **alias** subcommand creates aliases for **dbx** subcommands. The *Name* parameter is the alias being created. The *String* parameter is a series of **dbx** subcommands that, after the execution of this subcommand, can be referred to by *Name*. If the **alias** subcommand is used without parameters, it displays all current aliases.

Examples

1. To substitute `rr` for `rerun`, enter:

   ```
   alias rr rerun
   ```

2. To run the two subcommands `print n` and `step` whenever `printandstep` is typed at the command line, enter:

   ```
   alias printandstep "print n; step"
   ```

3. The `alias` subcommand can also be used as a limited macro facility. For example:

   ```
   (dbx) alias px(n) "set $hexints; print n; unset $hexints"
   (dbx) alias a(x,y) "print symname[x]->symvalue._n_n.name.Id[y]"
   (dbx) px(126)
   0x7e
   ```

   In this example, the alias px prints a value in hexadecimal without permanently affecting the debugging environment.

assign Subcommand

assign *Variable*=*Expression*

The **assign** subcommand assigns the value specified by the *Expression* parameter to the variable specified by the *Variable* parameter.

Examples

1. To assign a value of 5 to the x variable, enter:

   ```
   assign x = 5
   ```

2. To assign the value of the y variable to the x variable, enter:

   ```
   assign x =  y
   ```

3. To assign the character value `'z'` to the z variable, enter:

   ```
   assign  z  =  'z'
   ```

4. To assign the boolean value `false` to the logical type variable B, enter:

   ```
   assign  B  =  false
   ```

5. To assign the `"Hello World"` string to a character pointer Y, enter:

   ```
   assign  Y  =  "Hello  World"
   ```

6. To disable type checking, set the **dbx** debug program variable `$unsafeassign` by entering:

```
set $unsafeassign
```

See Displaying and Modifying Variables.

attribute Subcommand

attribute [*AttributeNumber* ...]

The **attribute** subcommand displays information about the user thread, mutex, or condition attributes objects defined by the *AttributeNumber* parameters. If no parameters are specified, all attributes objects are listed.

For each attributes object listed, the following information is displayed:

Item	Description
attr	Indicates the symbolic name of the attributes object, in the form $a*AttributeNumber*.
obj_addr	Indicates the address of the attributes object.
type	Indicates the type of the attributes object; this value can be thr, mutex, or cond for user threads, mutexes, and condition variables respectively.
state	Indicates the state of the attributes object. This value can be valid or inval.
stack	Indicates the stacksize attribute of a thread attributes object.
scope	Indicates the scope attribute of a thread attributes object. This value determines the contention scope of the thread, and defines the set of threads with which it must contend for processing resources. The value can be sys or pro for system or process contention scope.
prio	Indicates the priority attribute of a thread attributes object.
sched	Indicates the schedpolicy attribute of a thread attributes object. This attribute controls scheduling policy, and can be fifo , rr (round robin), or other.
p-shar	Indicates the process-shared attribute of a mutex or condition attribute object. A mutex or condition is process-shared if it can be accessed by threads belonging to different processes. The value can be yes or no.
protocol	Indicates the protocol attribute of a mutex. This attribute determines the effect of holding the mutex on a threads priority. The value can be no_prio, prio, or protect.
clock	Indicates the clock attribute of a condition attribute object. This attribute determines which clock must be used when a thread that waits for the condition variable as specified a timeout. The value can be realtime or monotonic.

Notes:

1. The **print** subcommand of the **dbx** debug program recognizes symbolic attribute names, and can be used to display the status of the corresponding object.
2. The available attributes depend on the implementation of POSIX options.

Examples

1. To list information about all attributes, enter:

   ```
   attribute
   ```

   The output is similar to:

   ```
   attr   obj_addr   type  state  stack   scope    prio
   sched p-shar
   $a1   0x200035c8  mutex valid                              no
   $a2   0x20003628  cond  valid                              no
   $a3   0x200037c8  thr   valid  57344    sys      126 other
   ```

```
$a4   0x200050f8  thr   valid  57344    pro      126 other
```

2. To list information about attributes 1 and 3, enter:

```
attribute 1 3
```

The output is similar to:

```
attr   obj_addr   type  state  stack   scope    prio
sched p-shar
$a1   0x200035c8  mutex valid                              no
$a3   0x200037c8  thr   valid  57344    sys      126 other
```

See the **condition** subcommand, **mutex** subcommand, **print** subcommand, and **thread** subcommand for the **dbx** command.

Also, see Creating Threads, Using Mutexes, and Using Condition Variables in *General Programming Concepts: Writing and Debugging Programs*.

call Subcommand

call *Procedure* ([*Parameters*])

The **call** subcommand runs the procedure specified by the *Procedure* parameter. The return code is not printed. If any parameters are specified, they are passed to the procedure being run.

Note: The `call` subcommand cannot be used to call functions that take vector parameters.

Example

To call a command while running the **dbx** command, enter:

```
(dbx) call printf("hello")
hello
```

printf returns successfully.

case Subcommand

case [**default** | **mixed** | **lower** | **upper**]

The **case** subcommand changes how the **dbx** debug program interprets symbols. The default handling of symbols is based on the current language. If the current language is C, C++, or undefined, the symbols are not folded; if the current language is Fortran, the symbols are folded to lowercase. Use this subcommand if a symbol needs to be interpreted in a way not consistent with the current language.

Entering the **case** subcommand with no parameters displays the current case mode.

Flags

Item	Description
default	Varies with the current language.
mixed	Causes symbols to be interpreted as they actually appear.
lower	Causes symbols to be interpreted as lowercase.
upper	Causes symbols to be interpreted as uppercase.

Examples

1. To display the current case mode, enter:

```
case
```

2. To instruct **dbx** to interpret symbols as they actually appear, enter:

```
case mixed
```

3. To instruct **dbx** to interpret symbols as uppercase, enter:

```
case upper
```

See Folding Variables to Lowercase and Uppercase.

catch Subcommand

catch [*SignalNumber* | *SignalName*]

The **catch** subcommand starts the trapping of a specified signal before that signal is sent to the application program. This subcommand is useful when the application program being debugged handles signals such as interrupts. The signal to be trapped can be specified by number or by name using either the *SignalNumber* or the *SignalName* parameter, respectively. Signal names are case insensitive, and the **SIG** prefix is optional. If the *SignalNumber* and the *SignalName* parameters are not specified, all signals are trapped by default except the **SIGHUP**, **SIGCLD**, **SIGALARM**, and **SIGKILL** signals. If no arguments are specified, the current list of signals to be caught is displayed.

Examples

1. To display a current list of signals to be caught by the **dbx** command, enter:

```
catch
```

2. To trap signal SIGALARM, enter:

```
catch SIGALARM
```

See the **ignore** subcommand and Handling Signals.

clear Subcommand

clear *SourceLine*

The **clear** subcommand removes all stops at a particular source line. The *SourceLine* parameter can be specified in two formats:

- As an integer
- As a file name string followed by a : (colon) and an integer

Examples

To remove breakpoints set at line 19, enter:

```
clear 19
```

The **cleari** subcommand and **delete** subcommand. Also, see Setting and Deleting Breakpoints in in *General Programming Concepts: Writing and Debugging Programs*.

cleari Subcommand

cleari *Address*

The **cleari** subcommand clears all the breakpoints at the address specified by the *Address* parameter.

Examples

1. To remove a breakpoint set at address 0x100001b4, enter:

```
cleari 0x100001b4
```

2. To remove a breakpoint set at the main() procedure address, enter:

```
cleari &main
```

See the **clear** subcommand, the **delete** subcommand, and Setting and Deleting Breakpoints in in *General Programming Concepts: Writing and Debugging Programs*.

condition Subcommand

condition [**wait** | **nowait** | *ConditionNumber* ...]

The **condition** subcommand displays information about one or more condition variables. If one or more *ConditionNumber* parameters are given, the **condition** subcommand displays information about the specified condition variables. If no flags or parameters are specified, the **condition** subcommand lists all condition variables.

The information listed for each condition is as follows:

Item	Description
`cv`	Indicates the symbolic name of the condition variable, in the form $c*ConditionNumber*.
`obj_addr`	Indicates the memory address of the condition variable.
`clock`	Indicates the clock attribute of the condition variable.
`num_wait`	Indicates the number of threads waiting on the condition variable.
`waiters`	Lists the user threads which are waiting on the condition variable.

> **Note:** The **print** subcommand of the **dbx** debug program recognizes symbolic condition variable names, and can be used to display the status of the corresponding object.

Flags

Item	Description
wait	Displays condition variables which have waiting threads.
nowait	Displays condition variables which have no waiting threads.

Examples

1. To display information about all condition variables, enter:

   ```
   condition
   ```

2. To display information about all condition variables which have waiting threads, enter:

   ```
   condition  wait
   ```

3. To display information about the condition variable 3, enter:

   ```
   condition 3
   ```

   The output is similar to:

   ```
   cv      obj_addr     num_wait  waiters
   $c3     0x20003290         0
   ```

See the **attribute** subcommand, **mutex** subcommand, **print** subcommand, and **thread** subcommand.

Also, see Using Condition Variables in *General Programming Concepts: Writing and Debugging Programs*.

cont Subcommand

cont [*SignalNumber* | *SignalName*]

The **cont** subcommand continues the execution of the application program from the current stopping point until either the program finishes or another breakpoint is reached. If a signal is specified, either by the number specified in the *SignalNumber* parameter or by the name specified in the *SignalName*

parameter, the program continues as if that signal is received. Signal names are not case-sensitive and the **SIG** prefix is optional. If no signal is specified, the program continues as if it was not stopped.

Examples

1. To continue program execution from current stopping point, enter:

```
cont
```

2. To continue program execution as though it received the signal SIGQUIT, enter:

```
cont SIGQUIT
```

See the **detach** subcommand for the **dbx** command, the **goto** subcommand for the **dbx** command, the **next** subcommand for the **dbx** command, the **skip** subcommand for the **dbx** command, the **step** subcommand for the **dbx** command.

corefile Subcommand

The **corefile** subcommand displays information from the header of a core file, including the executable name, core file format version information, flags indicating which data is available, the signal that caused the crash, and the execution mode of the process that dumped core.

coremap Subcommand

coremap [*stack* | *data* | *sdata* | *mmap* | *shm* | *loader*]

The **coremap** subcommand displays the mapping of a particular address space region. If you do not specify the region name, the **coremap** subcommand displays all available mappings.

Examples

1. To display the mapping of a shared memory region, enter:

```
coremap shm
```

2. To display the mapping of a memory mapped region, enter:

```
coremap mmap
```

3. To display the mappings of all of the regions described by the loader entries, enter:

```
coremap loader
```

4. To display all of the available mappings, enter:

```
coremap
```

See the **corefile** subcommand.

delcmd Subcommand

delcmd *EventNumber* { *Number...* | **all** }

The **delcmd** subcommand removes the **dbx** subcommands associated with the specified event. The **dbx** subcommands to be removed can be specified through *Number* parameters, or all **dbx** subcommands associated with the specified event can be removed by using the `all` flag. The *EventNumber* parameter specifies the event from which the **dbx** subcommands are to be removed.

Flags

Item	Description
all	Removes all the **dbx** subcommands associated with the specified event.

Examples

1. To remove all the **dbx** subcommands from event number 2, enter:

```
delcmd 2 all
```

2. To remove **dbx** subcommand number 1 from event number 3, enter:

```
delcmd 3 1
```

3. To remove **dbx** subcommands numbers 1 and 2 from event number 2, enter:

```
delcmd 2 1 2
```

See the **addcmd** subcommand, **clear** subcommand, the **delete** subcommand, **disable** subcommand, **enable** subcommand, the **stop** subcommand, the **status** subcommand, and the **trace** subcommand. Also see Setting and Deleting Breakpoints in in *General Programming Concepts: Writing and Debugging Programs*.

delete Subcommand

delete { *Number* ... | **all** | `tskip` [**for $t***threadnumber*]}

The **delete** subcommand removes traces and stops from the application program and `tskip` counts for a thread. The traces and stops to be removed can be specified through the *Number* parameters, or all traces and stops can be removed by using the `all` flag. Use the **status** subcommand to display the numbers associated by the **dbx** debug program with a trace or stop.

The remaining `tskip` count, which was set using the `tskip` subcommand for a thread, can be deleted using the `tskip` flag. Use the `status` subcommand to display the remaining thread `tskip` counts. If no thread is specified, the current thread is used.

Flag

Item	Description
all	Removes all traces and stops.
for $t *threadnumber*	Specifies the thread number.

Examples

1. To remove all traces and stops from the application program, enter:

```
delete all
```

2. To remove traces and stops for event number 4, enter:

```
delete 4
```

3. To remove the `tskip` count for thread 3, enter:

```
delete tskip for $t3
```

4. To remove the `tskip` count for the current thread, enter:

```
delete tskip
```

See the **clear** subcommand, the **cleari** subcommand, the **status** subcommand, the `tskip` subcommand, and Setting and Deleting Breakpoints in in *General Programming Concepts: Writing and Debugging Programs*.

detach Subcommand

detach [*SignalNumber* | *SignalName*]

The **detach** subcommand continues the execution of the application program and exits the debug program. A signal can be specified either by:

- Name, using the *SignalName* parameter

- Number, using the *SignalNumber* parameter

  Signal names are not case-sensitive and the **SIG** prefix is optional.

  If a signal is specified, the program continues as if it received that signal. If no signal is specified, the program continues as if no stop occurred.

Examples

1. To continue execution of the application and exit **dbx**, enter:

```
detach
```

2. To exit **dbx** and continue execution of the application as though it received signal SIGREQUEST, enter:

```
detach SIGREQUEST
```

See Using the **dbx** Debug Program.

disable Subcommand

disable { *Number* ... **all** }

The **disable** subcommand disables traces and stops associated with debug events. The traces and stops to be disabled can be specified through the *Number* parameters, or all traces and stops can be disabled by using the **all** flag. Use the **status** subcommand to display the event numbers associated by the **dbx** debug program with a trace or stop.

Flags

Item	Description
all	Removes all traces and stops.

Examples

1. To disable all traces and stops from the application program, type:

```
disable all
```

2. To disable traces and stops for event number 4, type:

```
disable 4
```

For more information, see **enable subcommand**, **delete subcommand**, and **status subcommand**.

Also, see Setting and Deleting Breakpoints in *General Programming Concepts: Writing and Debugging Programs*.

display memory Subcommand

{ *Address,Address/* | *Address/* [*Count*] } [*Mode*] [**>***File*]

The **display memory** subcommand, which does not have a keyword to initiate the command, displays a portion of memory controlled by the following factors:

The range of memory displayed is controlled by specifying either:

- Two *Address* parameters, where all lines between those two addresses are displayed,

  OR
- One *Address* parameter where the display starts and a *Count* that determines the number of lines displayed from *Address.*

Specify symbolic addresses by preceding the name with an & (ampersand). Addresses can be expressions made up of other addresses and the operators + (plus sign), - (minus sign), and * (indirection). Any expression enclosed in parentheses is interpreted as an address.

- The format in which the memory is displayed is controlled by the *Mode* parameter. The default for the *Mode* parameter is the current mode. The initial value of *Mode* is **X**. The possible modes include:

Item	Description
b	Prints a byte in octal.
c	Prints a byte as a character.
d	Prints a short word in decimal.
D	Prints a long word in decimal.
Df	Prints a double-precision decimal float number.
DDf	Prints a quadruple-precision decimal float number.
f	Prints a single-precision real number.
g	Prints a double-precision real number.
h	Prints a byte in hexadecimal.
Hf	Prints a single-precision decimal float number.
i	Prints the machine instruction.
lld	Prints an 8-byte signed decimal number.
llu	Prints an 8-byte unsigned decimal number.
llx	Prints an 8-byte unsigned hexadecimal number.
llo	Prints an 8-byte unsigned octal number.
o	Prints a short word in octal
O	Prints a long word in octal.
p	Prints the address/pointer in hexadecimal.
q	Prints an extended-precision floating-point number.
s	Prints a string of characters terminated by a null byte.
x	Prints a short word in hexadecimal.
X	Prints a long word in hexadecimal.

Flag

Item	Description
>*File*	Redirects output to the specified file.

Examples

1. To display one long word of memory content in hexadecimal starting at the address `0x3fffe460`, enter:

```
0x3fffe460 / X
```

2. To display two bytes of memory content as characters starting at the variable y address, enter:

```
&y / 2c
```

3. To display the sixth through the eighth elements of the Fortran character string a_string, enter:

```
&a_string + 5, &a_string + 7/c
```

See Examining Memory Addresses in *General Programming Concepts: Writing and Debugging Programs*.

down Subcommand

down [*Count*]

The **down** subcommand moves the current function down the stack *Count* number of levels. The current function is used for resolving names. The default for the *Count* parameter is one.

Examples

1. To move one level down the stack, enter:

```
down
```

2. To move three levels down the stack, enter:

```
down 3
```

See the **up** subcommand, the **where** subcommand, and Displaying a Stack Trace in *General Programming Concepts: Writing and Debugging Programs*.

dump Subcommand

dump [*Procedure* | "PATTERN"] [**>***File*]

The **dump** subcommand displays the names and values of all variables in the specified procedure or those variables that match with the specified pattern. If the *Procedure* parameter is a period (**.**), then all active variables are displayed. If the *Procedure* nor "PATTERN" parameter is specified, the current procedure is used. The "PATTERN" parameter is a wildcard expression with the `*`, `?`, and `[]` meta-characters. When "PATTERN" is used, it displays all the matching symbols in the global space (from all the procedures). If the **>***File* flag is used, the output is redirected to the specified file.

Flags

Item	Description
>*File*	Redirects output to the specified file.

Examples

1. To display names and values of variables in the current procedure, enter:

```
dump
```

2. To display names and values of variables in the **add_count** procedure, enter:

```
dump add_count
```

3. To display names and values of variables starting from the character `s`, enter:

```
dump "s*"
```

4. To redirect names and values of variables in the current procedure to the **var.list** file, enter:

```
dump > var.list
```

See Displaying and Modifying Variables in *General Programming Concepts: Writing and Debugging Programs*.

edit Subcommand

edit [*Procedure* | *File*]

The **edit** subcommand starts an editor on the specified file. The file might be specified through the *File* parameter or by specifying the *Procedure* parameter, where the editor is started on the file containing that procedure. If no file is specified, the editor is started on the current source file. The default is the **vi** editor. Override the default by resetting the **EDITOR** environment variable to the name of the required editor.

Examples

1. To start an editor on the current source file, enter:

```
edit
```

2. To start an editor on the `main.c` file, enter:

```
edit main.c
```

3. To start an editor on the file containing the `do_count()` procedure, enter:

```
edit do_count
```

See the **list** subcommand, the **vi** or **vedit** command.

enable Subcommand

enable { *Number* ... **all** }

The **enable** subcommand enables traces and stops associated with debug events. The traces and stops to be enabled can be specified through the *Number* parameters, or all traces and stops can be enabled by using the **all** flag. Use the **status** subcommand to display the event numbers associated by the **dbx** debug program with a trace or stop.

Flags

Item	Description
all	Removes all traces and stops.

Examples

1. To enable all traces and stops from the application program, type:

```
enable all
```

2. To enable traces and stops for event number 4, type:

```
enable 4
```

For more information, see **disable subcommand**, **delete subcommand**, **status subcommand**.

Also, see Setting and Deleting Breakpoints in *General Programming Concepts: Writing and Debugging Programs*.

fd Subcommand

fd [**raw**] [*start* [*end*]]

The **fd** subcommand displays file descriptor information. Using the **raw** option causes output to be displayed in raw hex format. Other optional arguments include *start* and *end* indices. If no index is given, then information about all available file descriptors is displayed. Use of one index displays a single file descriptor; two an inclusive range.

Examples

1. To view information about all file descriptors in hex, type:

```
fd raw
```

2. To view information about file descriptors in the range of 3 to 5, type:

```
fd 3 5
```

file Subcommand

file [*File*]

The **file** subcommand changes the current source file to the file specified by the *File* parameter; it does not write to that file. The *File* parameter can specify a full path name to the file. If the *File* parameter does not specify a path, the **dbx** program tries to find the file by searching the use path. If the *File* parameter is not specified, the **file** subcommand displays the name of the current source file. The **file** subcommand also displays the full or relative path name of the file if the path is known.

Examples

1. To change the current source file to the `main.c` file, enter:

```
file main.c
```

2. To display the name of the current source file, enter:

```
file
```

See the **func** subcommand. Also, see Changing the Current File or Procedure and Displaying the Current File in *General Programming Concepts: Writing and Debugging Programs*.

frame Subcommand

frame [*num*]

The **frame** subcommand changes the current function to the function corresponding to the specified stack frame number *num*. The current function is used for resolving names. The numbering of the stack frames starts from the currently active function stack frame (the function frame that is currently active is always numbered `0`). If there are *n* frames, the frame of the `main` function is numbered `n-1`. When no frame number is specified, information about the function associated with the current frame is displayed.

Examples

1. To move to frame number 2, enter:

```
frame 2
```

2. To display the current function on the stack, enter:

```
frame
```

See the **up** and **down** subcommands. Also, see Changing the Current File or Procedure and Displaying a Stack Trace in *General Programming Concepts: Writing and Debugging Programs*.

func Subcommand

func [*Procedure*]

The **func** subcommand changes the current function to the procedure or function specified by the *Procedure* parameter. If the *Procedure* parameter is not specified, the default current function is displayed. Changing the current function implicitly changes the current source file to the file containing the new function; the current scope used for name resolution is also changed.

Examples

1. To change the current function to the `do_count` procedure, enter:

```
func do_count
```

2. To display the name of the current function, enter:

```
func
```

See the **file** subcommand. Also, see Changing the Current File or Procedure in *General Programming Concepts: Writing and Debugging Programs*.

goto Subcommand

goto *SourceLine*

The **goto** subcommand causes the specified source line to be run next. Normally, the source line must be in the same function as the current source line. To override this restriction, use the **set** subcommand with the **$unsafegoto** flag.

Example

To change the next line to be executed to line 6, enter:

```
goto 6
```

See the **cont** subcommand, the **gotoi** subcommand, and the **set** subcommand.

gotoi Subcommand

gotoi *Address*

The **gotoi** subcommand changes the program counter address to the address specified by the *Address* parameter.

Example

To change the program counter address to address `0x100002b4`, enter:

```
gotoi 0x100002b4
```

See the **goto** subcommand.

handler Subcommand

handler { **atfork** | **cancel_cleanup** [**all** | *pthread id*] }

The **handler** subcommand displays information about atfork or cancelation cleanup handlers registered using **pthread_atfork**, and **pthread_cleanup_push**, respectively. Using the *atfork* option, the names of routines registered as `pre`, `parent` and `child` atfork handlers are displayed (with their respective arguments in the case of non-posix compliant atfork handlers). The **cancel_cleanup** option causes display of all registered cancelation cleanup handlers, with an optional *pthread id* parameter specifying a particular pthread, or **all** specifying all pthreads. If none is given, then the cancelation cleanup handlers for the current pthread are displayed, if there are any.

Examples

1. To view information about all registered atfork handlers, type:

   ```
   handler atfork
   ```

2. To view information about any registered cancelation cleanup handlers for the current pthread, type:

   ```
   handler cancel_cleanup
   ```

3. To view information about any registered cancelation cleanup handlers for the pthread object referred to as `$t2`, type:

   ```
   handler cancel_cleanup 2
   ```

help Subcommand

help [*Subcommand* | *Topic*]

The **help** subcommand displays help information for **dbx** subcommands or topics, depending upon the parameter you specify. Entering the **help** subcommand with the *Subcommand* parameter displays the syntax statement and description of the specified subcommand. Entering the **help** subcommand with the *Topic* parameter displays a detailed description of the specified topic. You do not need to provide the entire topic string with the **help** subcommand. The **dbx** program can recognize the topic if you provide a substring starting from the beginning of the topic. The following topics are available:

Item	Description
startup	Lists **dbx** startup options.
execution	Lists **dbx** subcommands related to program execution.
breakpoints	Lists **dbx** subcommands related to breakpoints and traces.
files	Lists **dbx** subcommands for accessing source files.
data	Lists **dbx** subcommands for accessing program variables and data.
machine	Lists descriptions of **dbx** subcommands for machine-level debugging.
environment	Lists **dbx** subcommands for setting **dbx** configuration and environment.
threads	Lists **dbx** subcommands for accessing thread-related objects.
expressions	Describes **dbx** expression syntax and operators.
scope	Describes how **dbx** resolves names from different scopes.
set_variables	Lists **dbx** debug variables with a usage description.
usage	Lists common **dbx** subcommands with brief descriptions.

Examples

1. To list all available **dbx** subcommands and topics, enter:

```
help
```

2. To display the description of the **dbx** subcommand **list**, enter:

```
help list
```

3. To display the description of the **dbx** topic **set_variables**, enter:

```
help set_variables
```

ignore Subcommand

ignore [*SignalNumber* | *SignalName*]

The **ignore** subcommand stops the trapping of a specified signal before that signal is sent to the application program. This subcommand is useful when the application program being debugged handles signals such as interrupts.

The signal to be trapped can be specified by:

- Number, with the *SignalNumber* parameter
- Name, with the *SignalName* parameter

Signal names are not case-sensitive. The **SIG** prefix is optional.

If the *SignalNumber* and the *SignalName* parameters are specified, all signals except the **SIGHUP**, **SIGCLD**, **SIGALRM**, and **SIGKILL** signals are trapped by default. The **dbx** debug program cannot ignore the **SIGTRAP** signal if it comes from a process outside of the debugger. If no arguments are specified, the list of currently ignored signals are displayed.

Example

To cause **dbx** to ignore alarm clock time-out signals sent to the application program, enter:

```
ignore alrm
```

See the **catch** subcommand. Also, see Handling Signals in *General Programming Concepts: Writing and Debugging Programs*.

kthread Subcommand

kthread [**raw**] [**info** | **ru**] [*tid*]

The **kthread** subcommand displays information about kernel threads. Using the **raw** option causes all output to be displayed in hex, regardless of whether it can be displayed in a more human-readable format. Using no arguments, summary information about all kernel threads is printed. Supplying a numeric thread ID causes the **dbx** command to show information about a single thread. The **info** option produces more detailed output about a thread, from the user thread structure. Use of the **ru** option displays the `ti_ru` data member, which contains resource usage information.

For more information about user threads, see **thread subcommand**.

Examples

1. To find information about the thread that is currently running, you must first obtain information about all threads by typing the following on the command line:

   ```
   kthread
   ```

   Threads that were running (or runnable) just before the **dbx** command stopped the process are marked with an asterisk. Choose the correct thread ID based on the output and type:

   ```
   kthread info tid
   ```

2. To view resource information in hex about all threads, type:

   ```
   kthread raw ru
   ```

limitbp Subcommand

limitbp (*bp1*, *Limit*) [(*bp2*, [**+**] *Limit*) ...]

The **limitbp** subcommand instructs the **dbx** command to stop running the debug program only when the breakpoint is executed a specified number of times. If the '**+**' character precedes the limit, the limit of that event is changed to the sum of the limit that is specified in the subcommand and the count of the number of times that the event is already executed. That is, the **dbx** command stops running the debug program when the breakpoint is about to be executed for the specified *Limit* after the **limitbp** subcommand is already run.

Examples

1. To instruct the **dbx** command to stop execution of the debug program when breakpoint 1 is about to be executed the 10th time, enter:

   ```
   limitbp (1, 10)
   ```

2. To instruct the **dbx** command to stop execution of the debug program when either breakpoint 1 is about to be executed the 15th time or breakpoint 2 is about to be executed the 20th time or both, enter:

   ```
   limitbp (1, 15) (2, 20)
   ```

3. To instruct the **dbx** command to stop execution of the debug program when breakpoint 1 is about to be executed the 20th time after the **limitbp** subcommand was run, enter:

   ```
   limitbp (1, +20)
   ```

list Subcommand

list [*Procedure* | *SourceLine-Expression* [**,***SourceLine-Expression*] | **at** *Address*]

The **list** subcommand displays a specified number of lines of the source file. The number of lines displayed are specified in one of following ways:

- By specifying a procedure using the *Procedure* parameter.

In this case, the **list** subcommand displays lines starting a few lines before the beginning of the specified procedure and until the list window is filled.

- By specifying a starting and ending source line number using the *SourceLine-Expression* parameter.

  The *SourceLine-Expression* parameter must consist of a valid line number followed by an optional + (plus sign), or - (minus sign), and an integer. In addition, a *SourceLine* of $ (dollar sign) might be used to denote the current line number; a *SourceLine* of @ (at sign) might be used to denote the next line number to be listed.

- By specifying the *$listwindow* internal **dbx** variable.

  If the **list** subcommand is used without parameters, the number of lines specified by the *$listwindow* variable are printed, beginning with the current source line. To change the default number of lines, set the *$listwindow* variable to the required number of lines. The *$listwindow* variable is a special debug program variable. Initially, the *$listwindow* variable is set to 10.

If the second source line is omitted, only the first line is printed.

All lines from the first line number specified to the second line number specified, inclusive, are then displayed.

When you specify an address after the **at** parameter in the **list** subcommand, the **list** subcommand displays the source lines that correspond to the specified address. Address can be specified as a decimal or hexadecimal unsigned integer, or a mnemonic that corresponds to registers, such as *$iar*, *$tfiar*, and *$tfhar* or debug variables.

Examples

1. To list the lines 1 through 10 in the current file, enter:

   ```
   list 1,10
   ```

2. To list 10, or `$listwindow`, lines around the `main` procedure, enter:

   ```
   list main
   ```

3. To list 11 lines around the current line, enter:

   ```
   list $-5,$+5
   ```

4. You can use simple integer expressions involving addition and subtraction in *SourceLineExpression* expressions. For example:

   ```
   (dbx) list $
   4 {

   (dbx) list 5
   5 char i = '4';

   (dbx) list sub
   23 char *sub(s,a,k)
   24 int a;
   25 enum status k;  .  .  .

   (dbx) move
   25
   (dbx) list @ -2
   23 char *sub(s,a,k)
   ```

5. You can display the source lines that correspond to a specific address. For example:

   ```
   (dbx) r
   [1] stopped in main at line 5
       5 int i, sum = 0;

   (dbx) list at $iar
   source file: "tt.c"
       5 int i, sum = 0;
       6 int last = 0;
       7
   ```

```
    8 scanf("%d", &last);
    9
    10 for ( i = 1; i &lt;=last; i++ ) {
    11 sum += i;
    12 }
    13 printf("sum = %d\n", sum);
    14

(dbx) list at ($iar+16)
source file: "tt.c"
    8 scanf("%d", &last);
    9
    10 for ( i = 1; i <= last; i++ ) {
    11 sum += i;
    12 }
    13 printf("sum = %d\n", sum);
    14
    15 return 0;
    16 }
```

See the **edit** subcommand, the **listi** subcommand, and the **move** subcommand. Also, see Displaying the current file in *General Programming Concepts: Writing and Debugging Programs*.

listi Subcommand

listi [*Procedure* | **at** *SourceLine* | *Address* [**,** *Address*]]

The **listi** subcommand displays a specified set of instructions from the source file. The instructions displayed are specified by:

- Providing the *Procedure* parameter, where the **listi** subcommand lists instructions from the beginning of the specified procedure until the list window is filled.
- Using the **at** *SourceLine* flag, where the **listi** subcommand displays instructions beginning at the specified source line and continuing until the list window is filled. The *SourceLine* variable can be specified as an integer or as a filename string followed by a : (colon) and an integer.
- Specifying a beginning and ending address using the *Address* parameters, where all instructions between the two addresses, inclusive, are displayed.

If the **listi** subcommand is used without flags or parameters, the next **$listwindow** instructions are displayed. To change the current size of the list window, use the **set $listwindow=***Value* subcommand.

Disassembly Modes

The **dbx** program can disassemble instructions for either the POWER family or PowerPC architecture. In the default mode, the **dbx** program displays the instructions for the architecture on which it is running.

The **$instructionset** and **$mnemonics** variables of the **set** subcommand for the **dbx** command allow you to override the default disassembly mode. For more information, see the **set** subcommand for the **dbx** command.

Flag

Item	Description
at *SourceLine*	Specifies a starting source line for the listing.

Examples

1. To list the next 10, or `$listwindow`, instructions, enter:

   ```
   listi
   ```

2. To list the machine instructions beginning at source line `10`, enter:

   ```
   listi at 10
   ```

3. To list the machine instructions beginning at source line 5 in file `sample.c`, enter:

   ```
   listi at "sample.c":5
   ```

4. To list the instructions between addresses `0x10000400` and `0x10000420`, enter:

```
listi 0x10000400, 0x10000420
```

See the **list** subcommand and the **set** subcommand. Also, see Debugging at the Machine Level with dbx in *General Programming Concepts: Writing and Debugging Programs*.

malloc Subcommand

malloc [> *File*]

The **malloc** subcommand with no options prints out a list of enabled options and allocation policies as well as a statistical summary of malloc usage since process startup.

malloc [**allocation** [{ *address* | *size* | *heap* | *pid* | *tid* | *time* } { "<" | "==" | ">" "!=" | “~=”]} *Value*]] [> *File*]

The **allocation** option to the **malloc** subcommand displays a sorted list of all the allocations currently held by the process. Using an optional `attribute RELOP value` argument allows for a more narrow selection of active allocations.

malloc [**freespace** [{ *address* | *size* | *heap* } { "<" | "==" | ">" | "!=" | “~=”]} *Value*]] [> *File*]

The **freespace** option to the **malloc** subcommand displays a sorted list of all the free space available in the process heap. Using an optional `attribute RELOP value` argument allows for a more narrow selection of free space nodes.

Note: ~= operator can be used only with address option. This operator is used to fetch the free space or allocation node to which the specified address belongs to.

malloc *address*

The **malloc** subcommand with address displays the nodes details of the address, the address need not be a starting address of an allocated or free node.

Flags

Item	Description
> *File*	Redirects output to the specified file.

For more information, see System Memory Allocation Using the malloc Subsystem in *General Programming Concepts: Writing and Debugging Programs*.

map Subcommand

map { [*Format*] [**entry** *ModuleNumber* [, *ModuleNumber*] | *Address* | *SymbolName*] [**for $t***threadnumber*] [**>** *File*] }

The **map** subcommand displays characteristics for loaded portions of the application. This information can include the module name, member name, text origin, text end, text length, data origin, data end, data length, TLS data origin, TLS data end, TLS data length, and file descriptor for each loaded module. The entries to be displayed can be specified in the following ways:

- By specifying a single entry using the *ModuleNumber* parameter.
- By specifying a range of entries using two comma-separated *ModuleNumber* parameters.
- By specifying an address to be resolved to a loaded module using the *Address* parameter.
- By specifying a symbol name to be resolved to a loaded module using the *SymbolName* parameter.

When called without one of the above specifications, the map subcommand displays information for all loaded portions of the application.

The *Format* argument specifies the output mode for the loaded module descriptions. The following list contains possible values for the *Format* argument:

Item	Description
abbr	Specifies the abbreviated output mode, which consists of a single line for each loaded module containing the entry number, module name, and optional member name for that module.
normal	Specifies the normal output mode, which consists of the entry number, module name, member name, text origin, text length, data origin, data length, and file descriptor for each loaded module. If the loaded module has TLS data, the TLS data origin and TLS data length are also displayed.
raw	Specifies the raw output mode, which consists of a single unformatted line for each module containing the following space-separated fields: entry number, module name with optional member name, text origin, text end, text length, data origin, data end, data length, and file descriptor. If the loaded module has TLS data, the TLS data origin, TLS data end, and TLS data length are also displayed.
verbose	Specifies the verbose output mode, which consists of the entry number, module name, member name, text origin, text end, text length, data origin, data end, data length, and file descriptor for each loaded module. If the loaded module has TLS data, the TLS data origin, TLS data end, and TLS data length are also displayed.

If no *Format* parameter is specified, the **dbx** command uses the value of the **$mapformat** internal variable. If no *Format* parameter is specified and **$mapformat** is unset, the **dbx** command displays loaded module information in normal mode.

The TLS data information of the specified thread is displayed if the loaded module has TLS data. If no thread is specified, the current thread is used.

Flags

Item	Description
> *File*	Redirects output to the specified file.
entry *ModuleNumber* [, *ModuleNumber*]	Specifies the module or range of modules to be displayed.
for $t *threadnumber*	Specifies the thread number.

Examples

1. To list all loaded modules in abbreviated mode, type:

```
map abbr
```

2. To list loaded modules 3 through 5 in verbose mode, type:

```
map verbose entry 3,5
```

3. To list the loaded module that contains address `0x20001000`, type:

```
map 0x20001000
```

4. To list the loaded module that contains variable `example`, type:

```
map example
```

5. To list the loaded modules in normal mode with TLS data information of the modules for the thread 2, type:

```
map normal for $t2
```

For more information, see the **$mapformat** internal variable. See also, Debugging at the Machine Level with dbx in *General Programming Concepts: Writing and Debugging Programs*.

move Subcommand

move *SourceLine*

The **move** subcommand changes the next line to be displayed to the line specified by the *SourceLine* parameter. This subcommand changes the value of the @ (at sign) variable.

The *SourceLine* variable can be specified as an integer or as a file name string followed by a : (colon) and an integer.

Examples

1. To change the next line to be listed to line 12, enter:

```
move 12
```

2. To change the next line to be listed to line 5 in file `sample.c`, enter:

```
move "sample.c":5
```

See the **list** subcommand. Also, see Displaying the Current File in *General Programming Concepts: Writing and Debugging Programs*.

multproc Subcommand

multproc [**on** | **parent** | **child** | **off**]

The **multproc** subcommand specifies the behavior of the **dbx** debug program when forked and exceed processes are created. The **on** flag is used to specify that a new **dbx** session is created to debug the child path of a fork. The original **dbx** continues to debug the parent path. The **parent** and **child** flags are used to specify a single path of a fork to follow. All flags except **off** enable **dbx** to follow an exceed process. The **off** flag disables multiprocess debugging. If no flags are specified, the **multproc** subcommand returns the status of multiprocess debugging.

The **dbx** program uses the X Window System for multiprocess debugging. The **dbx** program opens as many windows as needed for multiprocessing. The title for each child window is the process ID (pid) of the child process. To switch between processes, use the X Window System handling techniques to activate the window where the **dbx** command session is displayed. If the system does not have the X Window System support, a warning message is issued when the debugger forks, and the **dbx** program continues debugging only the parent process. Multiprocess debugging can also be unsuccessful for the following reasons:

- The **dbx** program is not running in the X Window System environment.
- The X Window System is running but the **dbx** global **$xdisplay** variable is not set to a valid display name. The **$xdisplay** variable is initialized to the shell **DISPLAY** environment variable. The **set** *Name=Expression* **dbx** subcommand can be used to change the value of the display name.
- The **/tmp** directory does not allow read or write access to the debugging program. The **dbx** program requires a small amount of space in this directory when controlling an Xwindow environment.
- The system does not have enough resources to accommodate a new Xwindow.

If **$xdisplay** is set to a remote display, the user might not be able to see the newly created Xwindow. If the **$xdisplay** setting is not correct, the X Window System or other system resources report the cause of the failure.

The **dbx** program does not distinguish between different types of failures, but the following message is sent when the subcommand is not successful:

```
Warning: dbx subcommand multiproc fails. dbx
continued with multproc disabled.
```

The user-defined configuration of the newly created window can be defined under the **dbx_term** application name in the **.Xdefaults** file.

Flags

Item	Description
on	Enables multiprocess debugging.
off	Disables multiprocess debugging.

Examples

1. To check the status of multiprocess debugging, enter:

```
multproc
```

2. To enable multiprocess debugging, enter:

```
multproc on
```

3. To disable multiprocess debugging, enter:

```
multproc off
```

See the **screen** subcommand and the **fork** subroutine. Also, see Debugging Programs Involving Multiple Processes in *General Programming Concepts: Writing and Debugging Programs*.

mutex Subcommand

mutex [**lock** | **unlock** | **thnum** | **utid** | *MutexNumber* ...]

The **mutex** subcommand displays information about mutexes. If the *MutexNumber* parameter is given, the **mutex** subcommand displays information about the specified mutexes. If no flags or parameters are specified, the **mutex** subcommand displays information about all mutexes.

The information listed for each mutex is as follows:

Item	Description
`mutex`	Indicates the symbolic name of the mutex, in the form $m*MutexNumber*.
`type`	Indicates the type of the mutex: `non-rec` (non recursive), `recursi` (recursive) or `fast`.
`obj_addr`	Indicates the memory address of the mutex.
`lock`	Indicates the lock state of the mutex: `yes` if the mutex is locked, `no` if not.
`owner`	If the mutex is locked, indicates the symbolic name of the user thread which holds the mutex.
`blockers`	List the user threads which are blocked on this mutex variable.

Note: The **print** subcommand of the **dbx** debug program recognizes symbolic mutex names, and can be used to display the status of the corresponding object.

Flags

Item	Description
lock	Displays information about locked mutexes.
unlock	Displays information about unlocked mutexes.
thnum	Displays information about all the mutexes held by a particular thread.
utid	Displays information about all the mutexes held by a user thread whose user thread id matches the user thread id.

Examples

1. To display information about all mutexes, enter:

```
mutex
```

2. To display information about all locked mutexes, enter:

   ```
   mutex  lock
   ```

3. To display information about mutexes number four, five and six enter:

   ```
   mutex 4 5 6
   ```

   The output is similar to:

   ```
   mutex   obj_addr         type     lock owner  blockers
   $m4    0x20003274        non-rec   no
   $m5    0x20003280        recursi   no
   $m6    0x2000328a        fast      no
   ```

4. To display information about all the mutexes held by thread 1, enter:

   ```
   mutex thnum 1
   ```

5. To display information about all the mutexes held by a thread whose user thread id is 0x0001, enter:

   ```
   mutex utid 0x0001
   ```

See the **attribute** subcommand, the **condition** subcommand, the **print** subcommand, and the **thread** subcommand.

Also, see. Using Mutexes *General Programming Concepts: Writing and Debugging Programs*.

next Subcommand

next [*Number*]

The **next** subcommand runs the application program up to the next source line. The *Number* parameter specifies the number of times the **next** subcommand runs. If the *Number* parameter is not specified, **next** runs once only.

If you use the **next** subcommand in a multithreaded application program, all the user threads run during the operation, but the program continues execution until the running thread reaches the specified source line. If you want to step the running thread only, use the **set** subcommand to set the variable **$hold_next**. Setting this variable might result in deadlock since the running thread might wait for a lock held by one of the blocked threads.

Examples

1. To continue execution up to the next source line, enter:

   ```
   next
   ```

2. To continue execution up to the third source line following the current source line, enter:

   ```
   next 3
   ```

See the **cont** subcommand, **goto** subcommand, **nexti** subcommand, **set** subcommand, and the **step** subcommand.

nexti Subcommand

nexti [*Number*]

The **nexti** subcommand runs the application program up to the next instruction. The *Number* parameter specifies the number of times the **nexti** subcommand runs. If the *Number* parameter is not specified, **nexti** runs once only.

If you use the **nexti** subcommand in a multithreaded application program, all the user threads run during the operation, but the program continues execution until the running thread reaches the specified machine instruction. If you want to step the running thread only, use the **set** subcommand to set the variable **$hold_next**. Setting this variable might result in deadlock since the running thread might wait for a lock held by one of the blocked threads.

Examples

1. To continue execution up to the next machine instruction, enter:

```
nexti
```

2. To continue execution up to the third machine instruction following the current machine instruction, enter:

```
nexti 3
```

See the **gotoi** subcommand, **next** subcommand, **set** subcommand, and **stepi** subcommand. Also, see Running a Program at the Machine Level in *General Programming Concepts: Writing and Debugging Programs*.

onceblock Subcommand

onceblock [**uninit** | **done**]

The **onceblock** subcommand displays information about blocks of initialization code registered using the **pthread_once** routine. With no arguments, information about all registered once blocks are shown. The optional **uninit** and **done** flags display only the once blocks that either have not, or have already executed, respectively, while supplying a numeric once ID displays information for a single once block.

Note: For the **onceblock** subcommand to work while debugging a live process, the environment variable AIXTHREAD_ONCE_DEBUG must be set equal to ON. Likewise, if debugging a core file, if the variable was not on when the process ran, the **onceblock** subcommand is not able to obtain any information.

Examples

1. To find out if any once blocks are not yet executed, type:

```
onceblock uninit
```

plugin Subcommand

plugin [*Name* [*Command*]]

The **plugin** subcommand passes the command specified by the *Command* parameter to the plug-in specified by the *Name* parameter. If no parameters are specified, the names of all available plug-ins are displayed.

Examples

1. To list all available plug-ins, type:

```
plugin
```

2. To start the subcommand "help" of a plug-in named "sample", type:

```
plugin sample help
```

3. To start the subcommand "`interpret 0x20000688`" of a plug-in named "`xyz`", type:

```
plugin xyz interpret 0x20000688
```

See the **pluginload** subcommand and **pluginunload** subcommand. Also see Developing for the dbx Plug-in Framework in *General Programming Concepts*.

pluginload Subcommand

pluginload *File*

The **pluginload** subcommand loads the plug-in specified by the *File* parameter. The *File* parameter must specify a path to the plug-in.

Note: Because the default **dbx** command is a 64-bit process, you must use the 32-bit version of the **dbx** command, named **dbx32**, to load 32-bit plug-ins.

Examples

To load the plug-in named "`sample`" at "`/home/user/dbx_plugins/libdbx_sample.so`", type:

```
pluginload /home/user/dbx_plugins/libdbx_sample.so
```

See the **plugin** subcommand and **pluginunload** subcommand. Also see Developing for the dbx Plug-in Framework in *General Programming Concepts*.

pluginunload Subcommand

pluginunload *Name*

The **pluginunload** subcommand unloads the plug-in specified by the *Name* parameter.

Examples

To unload the plug-in named "`sample`", type:

```
pluginunload sample
```

See the **plugin** subcommand and **pluginload** subcommand. Also see Developing for the dbx Plug-in Framework in *General Programming Concepts*.

print Subcommand

print *Expression* ...

print *Procedure* ([*Parameters*])

The **print** subcommand does either of the following operations:

- Prints the value of a list of expressions, specified by the *Expression* parameters.
- Executes a procedure, specified by the *Procedure* parameter and prints the return value of that procedure. Parameters that are included are passed to the procedure.

Examples

1. To display the value of x and the value of y shifted left two bits, enter:

   ```
   print x, y << 2
   ```

2. To display the value returned by calling the `sbrk` routine with an argument of 0, enter:

   ```
   print sbrk(0)
   ```

See the **assign** subcommand, the **call** subcommand, and the **set** subcommand.

printbp Subcommand

printbp [*bp1*] [*bp2*] ... | **all**

The **printbp** subcommand instructs the **dbx** command to print the number of times that each of the breakpoints or all the subcommands were run and the details of the limit on the breakpoint, if a limit was set on it.

Examples

1. To instruct the **dbx** command to print the number of times that breakpoint 1 is run and the details of the limit set, enter:

```
printbp 1
```

2. To instruct the **dbx** command to print the number of times that breakpoints 1 and 2 are run and to limit the number of times that breakpoints 1 and 2 can be allowed to run if a limit is set on them, enter:

```
printbp 1, 2
```

3. To instruct the **dbx** command to print the number of times that all breakpoints are run and the details of a limit on any breakpoint, if applicable, enter:

```
printbp all
```

proc Subcommand

proc [**raw**] [**cred** | **cru** | **ru** | **sigflags** | **signal**]

The **proc** subcommand displays information about the process. Usage of the **raw** option causes output to be displayed in raw hex, rather than interpreting values in a more human-readable fashion. Using the **proc** subcommand with no additional arguments outputs general information about the process, as is stored in the user process data structure. The **cred** option displays contents of the `pi_cred` data member, which describes the credentials of the process. The **cru** and **ru** options display data members `pi_cru` and `pi_ru` respectively, which contain resource usage information. The **sigflags** and **signal** options display information relating to the current signal status and registered signal handlers, as contained within the `pi_sigflags` and `pi_signal` data members.

Examples

1. To view resource usage information for the current process (or core file) in raw hex, type:

```
proc raw ru
```

2. To view signal handler information, type:

```
proc signal
```

prompt Subcommand

prompt [**"***String***"**]

The **prompt** subcommand changes the **dbx** command prompt to the string specified by the *String* parameter.

Example

To change the prompt to dbx>, enter:

```
prompt "dbx>"
```

See Defining a New dbx Prompt in *General Programming Concepts: Writing and Debugging Programs*.

quit Subcommand

quit

The **quit** subcommand terminates all processes running in the **dbx** debugging session.

See the **detach** subcommand.

registers Subcommand

registers [ALL | *$t*threadnumber ...] [**>***File*]

The **registers** subcommand displays the values of general-purpose registers, system control registers, floating-point registers, vector registers, and the current instruction register.

- General-purpose registers are denoted by the **$r***Number* variable, where the *Number* parameter indicates the number of the register.

  Note: The register value might be set to the **0xdeadbeef** hexadecimal value. The **0xdeadbeef** hexadecimal value is an initialization value assigned to general-purpose registers at process initialization.
- Floating point registers are denoted by the **$fr***Number* variable. By default, the floating-point registers are not displayed. To display the floating-point registers, use the **unset $noflregs dbx** subcommand.
- Vector registers are denoted by the **$vr*Number*** variable. The `$novregs` internal variable controls whether vector registers are displayed. The `$novregs` variable is set by default, and vector registers are not displayed. When `$novregs` is not set, and vector registers are valid (either debugging a program on a vector capable processor, or analyzing a core file containing vector registers state), then all the vector registers are displayed (vr0–vr31, vrsave, vscr). Vector registers can also be referenced by type. For example, the `$vrNf` (float), `$vrNs` (short), and `$vrNc` (char) vector register variables can be used with the `print` and `assign` subcommands to display and set vector registers by type.
- Vector scalar registers are denoted by the **$vsr*Number*** variable. By default, the vector scalar registers are not displayed. Unset **$novsregs** variable to display the vector scalar registers whenever vector scalar registers are valid (either debugging a program on a vector scalar capable processor, or analyzing a core file containing vector scalar registers state). As vector scalar registers are a superset of legacy floating point registers and vector registers, the debug variable **$novsregs** , when unset, takes precedence over **$noflregs** and **$novsregs** , whenever vector scalar registers state is valid. The **registers** subcommand will then display the vector scalar registers with the legacy register aliases along with it in braces. The floating point register aliases correspond to the low 64-bits only. Vector scalar registers can also be referenced by type as similar to vector registers. For example, the **$vsrNf** (float), **$vsrNs** (short), **$vsrNc** (char), **$vsrNg** (double) and **$vsrNll** (long long) vector scalar register variables can be used with the print and assign subcommands to display and set vector scalar registers by type.
- In the multithreaded environment option ALL displays the register details for all available threads. The register details of individual threads are displayed by specifying the thread number along with registers subcommand. Using the registers subcommand with no options display the registers for the current thread.

Note: The **registers** subcommand cannot display registers if the current thread is in kernel mode.

Flag

Item	Description
>*File*	Redirects output to the specified file.

See the **set** subcommand and the **unset** subcommand. Also, see Using Machine Registers in *General Programming Concepts: Writing and Debugging Programs*.

Example

To display the register details of threads $t1, $t2 and $t3, enter:

```
registers $t1 $t2 $t3
```

See the **set** subcommand and the **unset** subcommand. Also, see Using Machine Registers in *General Programming Concepts: Writing and Debugging Programs*.

rerun Subcommand

rerun [*Arguments*] [**<** *File*] [**>** *File*] [**> >** *File*] [**2>** *File*] [**2> >** *File*] [**>&** *File*] [**> >&** *File*]

The **rerun** subcommand begins execution of the object file. The *Arguments* are passed as command-line arguments. If the *Arguments* parameter is not specified, the arguments from the last **run** or **rerun** subcommand are reused.

Flags

Item	Description
<*File*	Redirects input so that input is received from *File*.
>*File*	Redirects output to *File*.
> >*File*	Appends redirected output to *File*.
2>*File*	Redirects standard error to *File*.
2> >*File*	Appends redirected standard error to *File*.
>&*File*	Redirects output and standard error to *File*.
> >&*File*	Appends output and standard error to *File*.

See the **run** subcommand.

resource Subcommand

resource { **owner** | **waiter** } [**all** | *pthread id*]

The **resource** subcommand displays information about which resources pthreads currently hold or are waiting on. The first argument, which is required, indicates whether you are interested in viewing pthreads that own resources or are waiting for them. The second argument can be used to indicate all pthreads, or a specific one. If none is given, then only information relevant to the current pthread is displayed, if applicable.

Note: The **resource** subcommand is only useful for debugging processes that run with several debugging environmental variables set to ON. These include AIXTHREAD_MUTEX_DEBUG, AIXTHREAD_COND_DEBUG, AIXTHREAD_RWLOCK_DEBUG, AIXTHREAD_READ_OWNER and AIXTHREAD_WAITLIST_DEBUG. If these variables are not turned on while debugging a live process, or were not on when a debugger core file was generated, the **resource** subcommand will be able to retrieve less information or none at all. Because use of these features can degrade performance, it is recommended that they be activated only for debugging purposes.

Examples

1. To ascertain whether the current pthread holds any resources, type:

```
resource owner
```

2. To view which resources any pthreads are waiting on, type:

```
resource waiter all
```

return Subcommand

return [*Procedure*]

The **return** subcommand causes the application program to execute until a return to the procedure specified by the *Procedure* parameter is reached. If the *Procedure* parameter is not specified, execution ceases when the current procedure returns.

Examples

1. To continue execution to the calling routine, enter:

```
return
```

2. To continue execution to the `main` procedure, enter:

```
return main
```

rwlock Subcommand

rwlock [read | write | *RwlockNumber*....]

The **rwlock** subcommand displays information about rwlocks. If the *RwlockNumber* parameter is given, the **rwlock** subcommand displays information about the specified rwlocks. If no flags or parameters are specified, the **rwlock** subcommand displays information about all rwlocks.

The information for each **rwlock** is as follows:

Item	Description
rwl	Indicates the symbolic name of the rwlock, in the form $rw *RwlockNumber*.
flag_value	Indicates the flag value.
owner	Indicates the owner of the rwlock
status	Indicates who is holding the rwlock. The values are read (if held by reader), write (if held by writer), free (if free).
wsleep[#]	Indicates threads blocking in write. # indicates the total number of threads blocking in write.
rsleep[#]	Indicates threads blocking in read. # indicates the total number of threads blocking in read.

Note: The **print** subcommand of the **dbx** debug program recognizes symbolic rwlock names, and can be used to display the status of the corresponding object.

Flags

Item	Description
read	Displays information about all rwlocks whose status is in read mode.
write	Displays information about all rwlocks whose status is in write mode.

Examples

1. To display information about all rwlocks, enter:

   ```
   rwlock
   ```

   The output is similar to:

   ```
   rwl     flag_value    owner status
   $rwl        1         $t1    write
           rsleeps[     0]:
           wsleeps[     0]:
   ```

2. To display information about all rwlocks in write mode:

   ```
   rwlock write
   ```

   The output is similar to:

   ```
   rwl     flag_value    owner status
   $rwl        1         $t1    write
           rsleeps[     0]:
           wsleeps[     0]:
   ```

See the **attribute** subcommand, the **condition** subcommand, **mutex** subcommand, the **print** subcommand, and the **thread** subcommand

run Subcommand

run [*Arguments*] [**<***File*] [**>***File*] [**> >***File*] [**2>***File*] [**2> >***File*] [**>&***File*] [**> >&***File*]

The **run** subcommand starts the object file. The *Arguments* are passed as command-line arguments.

Flags

Item	Description
<*File*	Redirects input so that input is received from *File*.
>*File*	Redirects output to *File*.
2>*File*	Redirects standard error to *File*.
> >*File*	Appends redirected output to *File*.
2> >*File*	Appends redirected standard error to *File*.
>&*File*	Redirects output and standard error to *File*.
> >&*File*	Appends output and standard error to *File*.

Example

To run the application with the arguments `blue` and 12, enter:

```
run blue 12
```

See the **rerun** subcommand.

screen Subcommand

screen

The **screen** subcommand opens an Xwindow for the **dbx** command interaction. You continue to operate in the window in which the process originated.

The **screen** subcommand must be run while the **dbx** debug program is running in the X Window System environment. If the **screen** subcommand is issued in a non-Xwindow environment, the **dbx** program displays a warning message and resumes debugging as if the **screen** subcommand was not given. The **screen** subcommand can also be unsuccessful in the following situations:

- The **dbx** program is not running in the X Window System environment.
- The X Window System is running but the **dbx** global **$xdisplay** variable is not set to a valid display name. The **$xdisplay** variable is initialized to the **DISPLAY** environment variable. The **dbx** subcommand **set** *Name=Expression* changes the value of the display name.
- The X Window System is running, but the **TERM** environment variable is not set to a valid command name to start a new window.
- The **/tmp** directory does not allow read or write access to the program. The **dbx** program requires a small amount of space in this directory when the screen command is executed.
- System does not have enough resources to accommodate a new Xwindow.

The **dbx** program does not distinguish between different types of failures, but the program does send the following message:

```
Warning: dbx subcommand screen fails. dbx
continues.
```

If **$xdisplay** is set to a remote display, you might not be able to see the newly created Xwindow. If the **$xdisplay** setting is not correct, the X Window System or other system resources report the problem.

The user-defined configuration of the newly created window can be defined under the **dbx_term** application name in the **.Xdefaults** file.

Example

To open an Xwindow for **dbx** command interaction, enter:

```
screen
```

set Subcommand

set [*Variable*=*Expression*]

The **set** subcommand defines a value for the **dbx** debug program variable. The value is specified by the *Expression* parameter; the program variable is specified by the *Variable* parameter. The name of the variable must not conflict with names in the program being debugged. A variable is expanded to the corresponding expression within other commands. If the **set** subcommand is used without arguments, the variables currently set are displayed.

Variable	Description
$catchbp	Catches breakpoints during the execution of the next command.
$codepage	Specifies the code set to use for interpreting characters within the program. When specified with a valid code page, all characters are read from the specified code set and converted to the code set in use by the current environment.
$compact_bt_ident	Specifies the limit on the number of characters in the identifier names that can be printed in a stack trace. The specified limit must be positive integers in the range 4 - 128. If this variable is set without specifying any limit value, the default number of characters that can be printed is 8. If the variable is set and the identifier name is four or more characters longer than the specified limit, the **dbx** command prints the specified number of characters of the original identifier name in the stack trace followed by three periods (...). For example, if the identifier name is *variable_example*, which is 16 characters long, and the specified limit is 7, the identifier name is printed as `variabl...`. However, if the identifier name is *variable_1*, which is 10 characters long, and the specified limit is 7, the **dbx** command does not shorten the identifier name to seven characters followed by three periods. It is printed as `variable_1`.
$compact_bt_string	Specifies the limit on the number of characters in the function argument strings that can be printed in a stack trace. The specified limit must be positive integers in the range 4 - 128. If this variable is set without specifying any limit value, the default number of characters that can be printed is 8. If the variable is set and the length of the string is four or more characters longer than the specified limit, the **dbx** command prints the specified number of characters of the original string in the stack trace followed by three periods (...). For example, if the string is *string_example*, which is 14 characters long, and the specified limit is 5, the string is printed as `strin...`. However, if the string is *string_1*, which is 8 characters long, and the specified limit is 5, the **dbx** command does not shorten the string to five characters followed by three periods. It is printed as `string_1`.
$deferevents	Turns on the deferred events feature.
$display_address_name	Displays the member variable identifiers and the memory address that the identifier occupies when examining a set of memory addresses that are using the **dbx** command.
$expandunions	Displays values for each part of variant records or unions.
$frame	Uses the stack frame pointed to by the address designated by the value of **$frame** for doing stack traces and accessing local variables.
$hexchars	Prints characters as hexadecimal values.
$hexin	Interprets addresses in hexadecimal.
$hexints	Prints integers as hexadecimal values.
$hexstrings	Prints character pointers in hexadecimal.
$hold_next	Holds all threads except the running thread during the **cont**, **next**, **nexti**, and **step** subcommands. Setting this variable might result in deadlock since the running thread might wait for a lock held by one of the blocked threads.
$ignoreifhandler	Does not stop when your program receives a signal which has a registered handler.
$ignoreload	Does not stop when your program performs the **load**, **unload**, or **loadbind** subroutine.
$ignorenonbptrap	Does not stop when your program encounters a non-breakpoint trap instruction and has a registered **SIGTRAP** handler.

Variable	Description
$instructionset	Overrides the default disassembly mode. The following list contains possible values for the *Expression* parameter: **"default"** Specifies the architecture on which the **dbx** program is running. **"com"** Specifies the instruction set for the common intersection mode of the PowerPC and POWER family architectures. The **dbx** program defaults to POWER processor-based mnemonics. **"pwr"** Specifies the instruction set and mnemonics for the POWER family architecture. **"pwrx"** Specifies the instruction set and mnemonics for the POWER2 implementation of the POWER family architecture for AIX 5.1 and earlier. **"pwr6"** Specifies the instruction set and mnemonics for the POWER6 implementation of the PowerPC architecture. **"pwr7"** Specifies the instruction set and mnemonics for the POWER7 implementation of the PowerPC architecture. **"pwr8"** Specifies the instruction set and mnemonics for the POWER8 implementation of the PowerPC architecture. **"pwr9"** Specifies the instruction set and mnemonics for the POWER9 implementation of the PowerPC architecture. **"601"** Specifies the instruction set and mnemonics for the PowerPC 601 RISC Microprocessor for AIX 5.1 and earlier. **"603"** Specifies the instruction set and mnemonics for the PowerPC 603 RISC Microprocessor for AIX 5.1 and earlier. **"604"** Specifies the instruction set and mnemonics for the PowerPC 604 RISC Microprocessor. **"970"** Specifies the instruction set and mnemonics for the PowerPC 970 microprocessor. **"ppc"** Specifies the instruction set and mnemonics defined in the POWER processor-based architecture, excluding the optional instructions. These instructions are available in all POWER processor-based implementations except the PowerPC 601 RISC Microprocessor in AIX 5.1 and earlier. **"any"** Specifies any valid POWER processor-based or POWER family instruction. For instruction sets that overlap, the default is the POWER processor-based mnemonics. If no value is set for the *Expression* parameter, the **dbx** program uses the default disassembly mode.
$java	When set, also sets the following variables, placing the **dbx** command in a mode to debug Java applications. When unset, also unsets the following variables: **$ignorenonbptrap** Suppresses notification of trap instructions generated by the Java Just-In-Time (JIT) compiler.
$listwindow	Specifies the number of lines to list around a function and the number to list when the **list** subcommand is used without parameters. The default is 10 lines.
$mapaddrs	Starts mapping addresses. Unsetting **$mapaddrs** stops address mapping.

Variable	Description
$mapformat	Specifies the default output mode for the **map** subcommand. **"abbr"** Specifies the abbreviated output mode, which consists of a single line for each loaded module containing the entry number, module name, and optional member name for that module. **"normal"** Specifies the normal output mode, which consists of the entry number, module name, member name, text origin, text length, data origin, data length, and file descriptor for each loaded module. If the loaded module has TLS data, the TLS data origin and TLS data length are also displayed. **"raw"** Specifies the raw output mode, which consists of a single unformatted line for each module containing the following space-separated fields: entry number, module name with optional member name, text origin, text end, text length, data origin, data end, data length, and file descriptor. If the loaded module has TLS data, the TLS data origin, TLS data end, and TLS data length are also displayed. **"verbose"** Specifies the verbose output mode, which consists of the entry number, module name, member name, text origin, text end, text length, data origin, data end, data length, and file descriptor for each loaded module. If the loaded module has TLS data, the TLS data origin, TLS data end, and TLS data length are also displayed. If no value is set for the *Expression* parameter, the **dbx** program uses the normal output mode.
$mnemonics	Changes the set of mnemonics to be used by the **dbx** program when disassembling. **"default"** Specifies the mnemonics that most closely match the specified instruction set. **"pwr"** Specifies the mnemonics for the POWER family architecture. **"ppc"** Specifies the mnemonics defined in the POWER processor-based architecture book, excluding the optional instructions. If no value is set for the *Expression* parameter, the **dbx** program uses the mnemonics that most closely match the specified instruction set.
$noargs	Omits arguments from subcommands, such as *where*, *up*, *down*, and *dump*.
$noflregs	Omits the display of floating-point registers from the **registers** subcommand.
$novregs	Omits the display of vector registers from the **registers** subcommand.
$novsregs	Omits the display of vector scalar registers from the **registers** subcommand
$octint	Interprets addresses in octal.
$octints	Prints integers in octal.
$pretty	Displays complex C and C++ data structure (struts, unions, arrays) values in a *pretty printed* format with the **print** subcommand. **"on"** Specifies pretty printing with each value on its own line and with indentation to represent the static scope of each value. **"verbose"** Specifies pretty printing with each value on its own line and with qualified names to represent the static scope of each value. A qualified name consists of a dot-separated list of the outer blocks with which the value is associated. **"off"** Specifies pretty printing off. This value is the default.
$print_dynamic	Displays the dynamic type of the C++ objects with print / dump command. By default this variable is not set.
$repeat	Repeats the previous command if no command was entered.
$sigblock	Blocks signals to your program.
$show_vft	Displays Virtual Function Table while printing C++ objects with print / dump command. By default it is not set.
$stack_details	Displays the frame number and the register set for each active function or procedure displayed by the where subcommand.

Variable	Description
$stepignore	Controls how the **dbx** command behaves when the **step**/tstep subcommand runs on a source line that calls another routine for which no debugging information is available. This variable enables the **step**/tstep subcommand to step over large routines for which no debugging information is available. The following list contains possible values for the *Expression* parameter: **"function"** Performs the function of the **next**/tnext subcommand for the **dbx** command. This value is the default value. **"module"** Performs the function of the **next**/tnext subcommand if the function is in a load module for which no debug information is available (such as a system library). **"none"** Performs the function of the **stepi**/tstepi subcommand for the **dbx** command in the background until it reaches an instruction for which source information is available. At that point **dbx** displays where execution stopped.
$trace_good_transaction	Instructs the **dbx** command to display the following message every time a transactional memory (TM) transaction is completed successfully. `Process {PID} may have performed a transaction - $texasr, $tfiar, $tfhar are valid and may be inspected` Successful transactions are not reported because this variable is disabled by default.
$thcomp	When **$thcomp** is set, the information displayed by the thread command **th-** is shown in a compressed format.
$unsafeassign	Turns off strict type checking between the two sides of an **assign** statement. Even if the **$unsafeassign** variable is set, the two sides of an **assign** statement might not contain storage types of different sizes.
$unsafebounds	Turns off subscript checking on arrays.
$unsafecall	Turns off strict type checking for arguments to subroutines or function calls.
$unsafegoto	Turns off the **goto** subcommand destination checking.
$vardim	Specifies the dimension length to use when printing arrays with unknown bounds. The default value is 10.
$xdisplay	Specifies the display name for the X Window System, for use with the **multproc** subcommand or the **screen** subcommand. The default is the value of the shell **DISPLAY** variable.

The **$unsafe** variables limit the usefulness of the **dbx** debug program in detecting errors.

Examples

1. To change the default number of lines to be listed to 20, enter:

```
set $listwindow=20
```

2. To disable type checking on the assign subcommand, enter:

```
set $unsafeassign
```

3. To disassemble machine instructions for the POWER7 processor, enter:

```
set $instructionset="pwr7"
```

4. To display strings encoded in the IBM-eucCN code set, enter:

```
set $codepage="IBM-eucCN"
```

5. To specify a limit of four characters in the identifiers and a limit of twelve characters in the strings that are displayed in a stack trace, enter the following command:

```
set $compact_bt_ident=6
set $compact_bt_string=12
```

The stack trace that uses identifiers such as long_identifier, long_variable_name_str, and recursive_fun, and string such as this_is_a_really_long_string looks similar to the following output:

```
long_i...(a = 11, long_v... = "this_is_a_re..."), line 3 in "example.c"
recurs...(), line 13 in "example.c"
```

See the **unset** subcommand. Also, see Changing Print Output with Special Debug Program Variables in *General Programming Concepts: Writing and Debugging Programs*.

set edit [vi, emacs] or set -o [vi, emacs] Subcommand

The **set** subcommand with the **-o** or **edit** option might be used to turn on one of the line edit modes. If the **set-o vi** or **set edit vi** command is given, you are placed in the input mode of the *vi* line editor. If the **set -o emacs** or **set edit emacs** command is given, you are placed in the input mode of the *emacs* line editor.

Example

1. To turn on the vi line editor, enter:

   ```
   set-o vi
   ```

   or

   ```
   set edit vi
   ```

sh Subcommand

sh [*Command*]

The **sh** subcommand passes the command specified by the *Command* parameter to the shell for execution. The **SHELL** environment variable determines which shell is used. The default is the **sh** shell. If no argument is specified, control is transferred to the shell.

Examples

1. To run the `ls` command, enter:

   ```
   sh ls
   ```

2. To escape to a shell, enter:

   ```
   sh
   ```

3. To use the SHELL environment variable, enter:

   ```
   sh echo $SHELL
   ```

See Running Shell Commands from dbx in *General Programming Concepts: Writing and Debugging Programs*.

skip Subcommand

skip [*Number*]

The **skip** subcommand continues execution of the application program from the current stopping point. A number of breakpoints equal to the value of the *Number* parameter are skipped and execution then ceases when the next breakpoint is reached or when the program finishes. If the *Number* parameter is not specified, it defaults to a value of one.

Example

To continue execution until the second breakpoint is encountered, enter:

```
skip 1
```

Also see the **cont** subcommand.

source Subcommand

source *File*

The **source** subcommand reads **dbx** subcommands from the file specified by the *File* parameter.

Example

To read the **dbx** subcommands in the `cmdfile` file, enter:

```
source cmdfile
```

See Reading dbx Subcommands from a File in *General Programming Concepts: Writing and Debugging Programs*.

status Subcommand

status [more] [**>***File*]

The **status** subcommand displays all user-defined breakpoints, tracepoints, and watchpoints, in addition to the remaining thread `tskip` counts (set by using the `tskip` subcommand). If the more parameter is specified, the **status** subcommand also displays the **dbx** subcommands associated with the breakpoints, tracepoints, and watchpoints. The **status** subcommand lists enabled events with square brackets ([]) surrounding the event number, disabled events with periods (..) surrounding the event number, and deferred events with angle brackets (<>) surrounding the event number.

The > flag sends the output of the **status** subcommand to a file specified in the *File* parameter.

Flag

Item	Description
>*File*	Redirects output to *File*.

Examples

1. To display all user-defined breakpoints, tracepoints, and watchpoints, as well as the remaining thread `tskip` counts, type:

   ```
   status
   ```

   The output is similar to:

   ```
   [1] stop at 13
   [2] stop at 14
   .3. stop at 15
   .4. stop at 16
   [5] stop at 17 ( count = 0, limit = 3 )
   <6> stop at 18 if g > 10
   <7> stop in func

   Remaining tskip counts:
   tskip 2 for $t1
   tskip 1 for $t5
   ```

   In the example output, events 3 and 4 are disabled, and events 6 and 7 are deferred.

2. To display all user-defined breakpoints, tracepoints, and watchpoints with associated **dbx** subcommands, enter:

   ```
   status more
   ```

   The output is similar to:

   ```
          [1] stop at 13
              [1] where
          .2. stop at 14
              [1] where
              [2] registers
   ```

```
        <3> stop at 15 if g > 10
            [1] where; registers
```

See the addcmd subcommand, the **clear** subcommand, the **delete** subcommand, the delcmd subcommand, the tskip subcommand, the **stop** subcommand, and the **trace** subcommand for the **dbx** command.

Also, see Setting and Deleting Breakpoints in *General Programming Concepts: Writing and Debugging Programs*.

step Subcommand

step [*Number*]

The **step** subcommand runs source lines of the application program. Specify the number of lines to be executed with the *Number* parameter. If the *Number* parameter is omitted, it defaults to a value of 1.

If you use the **step** subcommand on a multithreaded application program, all the user threads run during the operation, but the program continues execution until the running thread reaches the specified source line. If you want to step the running thread only, use the **set** subcommand to set the variable **$hold_next**. Setting this variable might result in deadlock since the running thread might wait for a lock held by one of the blocked threads.

Note: Use the **$stepignore** variable of the **set** subcommand to control the behavior of the **step** subcommand. The **$stepignore** variable enables the **step** subcommand to step over large routines for which no debugging information is available.

Examples

1. To continue execution for one source line, enter:

   ```
   step
   ```

2. To continue execution for five source lines, enter:

   ```
   step 5
   ```

3. To prevent the **dbx** program from single-stepping the **printf** function, as illustrated in the following example code:

   ```
   60 printf ("hello world \n");
   ```

   enter:

   ```
   set $stepignore="function"; step
   ```

See the **cont** subcommand, the **goto** subcommand, the **next** subcommand, the **set** subcommand, and the **stepi** subcommand.

stepi Subcommand

stepi [*Number*]

The **stepi** subcommand runs instructions of the application program. Specify the number of instructions to be executed in the *Number* parameter. If the *Number* parameter is omitted, it defaults to one.

If used on a multithreaded application program, the **stepi** subcommand steps the running thread only. All other user threads remain stopped.

Examples

1. To continue execution for one machine instruction, enter:

```
stepi
```

2. To continue execution for 5 machine instructions, enter:

```
stepi 5
```

See the **gotoi** subcommand, the **nexti** subcommand, and the **step** subcommand.

stop Subcommand

stop { [*Variable*] [**at** *SourceLine* | **in** *Procedure* | **on load** ["*ModuleName*"]] [**if** *Condition*]} ["{ "*Limit*" }"]

The **stop** subcommand halts the application program when certain conditions are fulfilled. The program is stopped when:

- The *Condition* is true when the **if** *Condition* flag is used.
- The *Procedure* is called if the **in** *Procedure* flag is used.
- The *Variable* is changed if the *Variable* parameter is specified.
- The *SourceLine* line number is reached if the **at** *SourceLine* flag is used.

  The *SourceLine* variable can be specified as an integer or as a file name string followed by a : (colon) and an integer.
- The *ModuleName* loaded module is loaded or unloaded if the **on load** flag is used and the *ModuleName* parameter is specified.

  The optional *ModuleName* variable can be specified as a single module name, or as a module name paired with a member name in the format:

  ```
  ModuleName(MemberName)
  ```
- Any loaded module is loaded or unloaded if the **on load** flag is used and the *ModuleName* parameter is not specified.

You can set the *Limit* parameter to instruct the **dbx** command to ignore a condition for a specified number of times. In other words, the *Limit* parameter specifies the number of times that the specified condition must be fulfilled before the debug program execution is stopped.

After any of these commands, the **dbx** debug program responds with a message reporting the event it built as a result of your command. The message includes the event ID associated with your breakpoint along with an interpretation of your command. The syntax of the interpretation might not be the same as your command. For example:

```
stop in main
[1] stop in main
stop at 19 if x == 3
[2] stop at "hello.c":19 if x = 3
stop in func
<3> stop in func
stop g
<4> stop g
stop in getdata {3}
[5] stop in getdata ( count = 0, limit = 3 )
```

The numbers in square brackets ([]) are the event identifiers associated with the breakpoints. The **dbx** debug program associates event numbers with each **stop** subcommand. When the program is halted as the result of one of the events, the event identifier is displayed along with the current line to show which event caused the program to stop. The numbers in angle brackets (<>) are the event identifiers for the deferred events. A deferred event is an event without having any breakpoint, tracepoint, or watchpoint associated with it, and is created whenever the input command involves the symbols that are not currently loaded in the memory. A normal event displayed in square brackets ([]) is also converted into a deferred event whenever the corresponding module is unloaded. Whenever the module corresponding to the deferred event is loaded into the memory, the deferred event is converted into the normal event, and

the corresponding breakpoint, tracepoint, or watchpoint is created. The events that you create coexist with internal events that are created by the **dbx** command, so the event numbers might not always be sequential.

A limit can be associated with an event after its creation by using the **limitbp** subcommand. To view the limit associated with an event, the **printbp** subcommand can be used.

Use the **status** subcommand to view these numbers. You can redirect output from **status** to a file. Use the **delete** or **clear** subcommand to turn the **stop** subcommand off, or use the **enable** or **disable** subcommands. Use the **addcmd** subcommand to add **dbx** subcommands to the specified event number and the **delcmd** subcommand to delete the associated **dbx** subcommands from the specified event number.

In a multithreaded application program, all user threads are halted when any user thread hits a breakpoint. A breakpoint set on a source line or function is hit by any user thread which executes the line or function, unless you specify conditions (as in example 9). The following aliases specify the conditions automatically:

- **bfth**(*Function*, *ThreadNumber*)
- **blth**(*SourceLine*, *ThreadNumber*)

ThreadNumber is the number part of the symbolic thread name as reported by the **thread** subcommand (for example, 5 is the *ThreadNumber* for the thread name $t5). These aliases are actually macros which produce the expanded subcommands shown in the following example:

```
stopi at &Function if ($running_thread == ThreadNumber)
stop at SourceLine if ($running_thread == ThreadNumber)
```

Flags

Item	Description
at *SourceLine*	Specifies the line number.
if *Condition*	Specifies the condition, such as true.
in *Procedure*	Specifies the procedure to be called.
on load *ModuleName*	Specifies the loaded module to be monitored.

Examples

1. To stop execution at the first statement in the `main` procedure, enter:

   ```
   stop in main
   ```

2. To stop execution when the value of the x variable is changed on line 12 of the execution, enter:

   ```
   stop x at 12
   ```

3. To stop execution at line 5 in file `sample.c`, enter:

   ```
   stop at "sample.c":5
   ```

4. To check the value of x each time that the **dbx** command runs a subroutine within `func1`, enter:

   ```
   stop in func1 if x = 22
   ```

5. To check the value of x each time that the **dbx** command begins to run `func1`, enter:

   ```
   stopi at &func1 if x = 22
   ```

6. To stop the program when the value of *Variable* changes, enter:

   ```
   stop Variable
   ```

7. To stop the program whenever *Condition* evaluates to true, enter:

```
stop if (x > y) and (x < 2000)
```

8. The following example shows how to display active events and remove them:

```
status
[1] stop in main
[2] stop at "hello.c":19 if x = 3
delete 1
status
[2] stop at "hello.c":19 if x = 3
clear 19
status
(dbx)
```

   The **delete** command eliminates events by event identifier. The **clear** command deletes breakpoints by line number.

9. To place a breakpoint at the start of `func1` only when executed by thread `$t5`, enter one of the following equivalent commands:

```
stopi at &func1 if ($running_thread == 5)
```

   or

```
bfth(func1, 5)
```

10. To stop the program when any module is loaded or unloaded, enter:

```
stop on load
```

11. To stop the program whenever module `Module` is loaded or unloaded, enter:

```
stop on load "Module"
```

12. To stop the program whenever member `Member` of module `Module` is loaded or unloaded, enter:

```
stop on load "Module(Member)"
```

13. To stop the program in a function `getdata` when it is called the third time, enter:

```
stop in getdata {3}
```

See the addcmd subcommand, the **clear** subcommand, the **delete** subcommand, the delcmd subcommand, **disable** subcommand, **enable** subcommand, the **limitbp** subcommand, the **printbp** subcommand, the **status** subcommand, the **stopi** subcommand, and the **trace** subcommand. Also, see Setting and Deleting Breakpoints in *General Programming Concepts: Writing and Debugging Programs*.

stophwp Subcommand

stophwp *Address Size*

The **stophwp** subcommand sets a hardware watchpoint stop for the specified memory region. The program stops when the contents of the region change.

Notes:

1. The success of the **stophwp** subcommand is hardware dependent. This feature is available only on POWER630 and POWER4 onwards.
2. As a result of the hardware limitation of being able to set only a single watchpoint, an active watchpoint event acts as a conflict when attempting to create another hardware watchpoint event with **stophwp** and **tracehwp**. As such, the previous event must be deleted before creating a new one. Also, since the existence of an active software watchpoint (created by some invocations of the **stop** and **trace** subcommands) negate the performance gains of hardware watchpoints, these types of events also act as conflicts which must be deleted before creating a hardware watchpoint.

Example

1. To stop the program when the contents of the 4 byte memory region starting at address 0x200004e8 change, enter:

```
stophwp 0x200004e8 4
```

See the **tracehwp** subcommand.

stopi Subcommand

stopi { [*Address*] [**at** *Address* | **in** *Procedure*] [**if** *Condition*]}

The **stopi** subcommand sets a stop at the specified location:

- With the **if** *Condition* flag, the program stops when the condition true is specified.
- With the *Address* parameter, the program stops when the contents of *Address* change.
- With the **at** *Address* flag, a stop is set at the specified address.
- With the **in** *Procedure* flag, the program stops when the *Procedure* is called.

Flags

Item	Description
if *Condition*	Specifies the condition, such as true.
in *Procedure*	Specifies the procedure to be called.
at *Address*	Specifies the machine instruction address.

Examples

1. To stop execution at address `0x100020f0`, enter:

```
stopi at 0x100020f0
```

2. To stop execution when the contents of address `0x100020f0` change, enter:

```
stopi 0x100020f0
```

3. To stop execution when the contents of address 0x100020f0 are changed by thread `$t1`, enter:

```
stopi 0x200020f0 if ($running_thread == 1)
```

See the **stop** subcommand. Also, see Debugging at the Machine Level with dbx in *General Programming Concepts: Writing and Debugging Programs*.

thdata Subcommand

thdata [*$tthreadnumber* [all | *key1* ...] ...] | [all]

The **thdata** subcommand prints the thread-specific data that is associated with different keys, which are created by using the **pthread_key_create()** function. You can use the **thdata** subcommand in the following ways.

Command	Action
thdata [all]	Prints the thread-specific data that is associated with all the keys for all the available threads.
thdata $t1 [all]	Prints the thread-specific data that is associated with all the keys for the *$t1* thread.

Command	Action
thdata $t1 key1 key2	Prints the thread-specific data that is associated with the keys *key1* and *key2* for the *$t1* thread.
thdata $t1 key1 key2 $t2 key1	Prints the thread-specific data that is associated with the keys *key1* and *key2* for the *$t1* thread, and the thread-specific data that is associated with the key *key1* for the *$t2* thread.

Examples

1. To print the data associated to the current thread with all the available keys, enter:

```
(dbx) thdata $t1
Thread  :  1
    Key : 1 Data pointer : 0x200f7a28
    Key : 2  Data pointer : 0x200f7aa8
    Key : 3 Data pointer : 0x200f7ac4
(dbx)
```

2. To print the data associated to multiple threads and multiple keys, enter:

```
(dbx) thdata $t1 2 3 $t2
Thread  :  1
    Key : 2  Data pointer : 0x200f7aa8
    Key : 3 Data pointer : 0x200f7ac4
Thread : 2
    Key : 2  Data pointer : 0x200f7b24
    Key : 3 Data pointer : 0x200f7ba4
(dbx)
```

See Thread-Specific Data in General Programming Concepts: Writing and Debugging Programs

thread Subcommand

Display Selected Threads

thread { [**info**] [**-**] [*ThreadNumber* ...] } | **current** | **run** | **susp** | **term** | **wait**

Select an Individual Thread

thread current [-] *ThreadNumber*

Hold or Release Threads

thread { **hold** | **unhold** } [-] [*ThreadNumber* ...]

Help for the options displayed

thread { **help**}

The **thread** subcommand displays and controls user threads.

The first form of the **thread** subcommand can display information in two formats. If the **thread** subcommand is **th**, then the information displayed is in the first format. If the **thread** subcommand is **th -**, then the information displayed is in the second format. If no parameters are given, information about all user threads is displayed. If one or more *ThreadNumber* parameters are given, information about the corresponding user threads is displayed. When the **thread** subcommand displays threads, the current thread line is preceded by a **>**. If the running thread is not the same as the current thread, its line is preceded by a *****. The information displayed by the **thread** subcommand in both the formats is described below.

The information displayed by the **thread** subcommand in the first format is as follows:

Item	Description
`thread`	Indicates the symbolic name of the user thread, in the form `$t`*ThreadNumber*.

Item	Description
state-k	Indicates the state of the kernel thread (if the user thread is attached to a kernel thread). This can be run, wait, susp, or term, for running, waiting, suspended, or terminated.
wchan	Indicates the event on which the kernel thread is waiting or sleeping (if the user thread is attached to a kernel thread).
state-u	Indicates the state of the user thread. Possible states are running, blocked, or terminated.
k-tid	Indicates the kernel thread identifier (if the user thread is attached to a kernel thread).
mode	Indicates the mode (kernel or user) in which the user thread is stopped (if the user thread is attached to a kernel thread).
held	Indicates whether the user thread has been held.
scope	Indicates the contention scope of the user thread; this can be sys or pro for system or process contention scope.
function	Indicates the name of the user thread function.

The information displayed by the **thread** subcommand in the second format is given below. By default, for the **thread** subcommand **th -**, the information is displayed in the long form.

Item	Description
thread	Indicates the symbolic name of the user thread, in the form $t*ThreadNumber*.

Kernel thread-related information

Item	Description
tid	Indicates the user thread identifier (if the user thread is attached to a kernel thread).
pri	Indicates the priority of the kernel thread.
sched	Indicates the scheduling policy of the kernel thread. This value can be fif, oth, rr, for fifo, other, or round robin scheduling policies.
state	Indicates the state of the kernel thread (if the user thread is attached to a kernel thread). This value can be run, wait, susp, or zomb, for running, waiting, suspended, or zombie.

User thread-related information

Item	Description
tid	Indicates the user thread identifier.
pri	Indicates the priority of the userl thread.
sched	Indicates the scheduling policy of the user thread. This value can be fif, oth, rr, for fifo, other, or round-robin scheduling policies.
state	Indicates the state of the user thread. This value can be running, creating, suspended, blocked, runnable, or terminated.
state	Indicates the user state in hex.
flags	Indicates the values for pthread flags in hex.
wchan	Indicates the event on which the kernel thread is waiting or sleeping (if the user thread is attached to a kernel thread).
mode	Indicates the mode (kernel or user) in which the user thread is stopped (if the user thread is attached to a kernel thread).
held	Indicates whether the user thread is held.

Item	Description
scope	Indicates the contention scope of the user thread; this value can be sys or pro for system or process contention scope.
cancelation	**pending** Indicates whether cancelation is pending or not. **state** Indicates the mode and state of cancelation. If the cancelation is not pending and the state and mode are enabled and deferred respectively, then it is represented by **ed**, if cancelation state and mode is enabled and asynchronous, then it is represented by **ea**, and if mode is not enabled, then it is represented by **d**. If the cancelation is pending and the cancelation state and mode is enabled and deferred respectively, then it is represented by **ED**, if cancelation state and mode is enabled and asynchronous, then it is represented by **EA**, and if mode is not enabled, then it is represented by **D**.

Item	Description
joinable	Indicates whether the thread can be joined or not.
boosted	Indicates the boosted value of the thread.
function	Indicates the name of the user thread function.
cursig	Indicates the current signal value.

If the option set $thcomp is set, then the information is displayed in the compressed form as shown in the following example.

```
m           mode           (k)ernel (u)ser
k           k-state        (r)unning (w)aiting (s)uspended (z)ombie
u           u-state        (r)unning (R)unnable (s)uspended (t)erminated

                           (b)locked (c)reating
h           held           (yes) (n)o
s           scope          (s)ystem (p)rocess
c           cancellation   not pending:  (e)nabled & (d)eferred,
                                         (e)nabled & (a)sync, (d)isabled
                           pending    :  (E)nabled & (D)eferred,
                                         (E)nabled & (A)sync, (D)isabled
j           joinable       (yes) (n)o
b           boosted        value of boosted field in pthread structure
plk         kernel thread  (oth)er  (fif)o  (rr)-> round-robin
            policy
plu         user thread    (oth)er  (fif)o  (rr)-> round-robin
            policy
prk         kernel thread  hex number
            policy
pru         user thread    hex number
            policy
k-tid                      kernel thread id in hex
u-tid                      pthread id in hex
fl                         value of flags field in pthread structure in hex
sta                        value of state field in pthread structure in hex
cs                         value of the current signal
wchan                      event for which thread is waiting
function                   function name
```

The second form of the **thread** subcommand is used to select the current thread. The **print**, **registers**, and **where** subcommands of the **dbx** debug program all work in the context of the current thread. The **registers** subcommand cannot display registers if the current thread is in kernel mode.

The third form of the **thread** subcommand is used to control thread execution. Threads can be held using the **hold** flag, or released using the **unhold** flag. A held thread is not resumed until it is released.

Note: The **print** subcommand of the **dbx** debug program recognizes symbolic thread names, and can be used to display the status of the corresponding object.

Flags

Item	Description
current	If the *ThreadNumber* parameter is not given, displays the current thread. If the *ThreadNumber* parameter is given, selects the specified user thread as the current thread.
help	Displays all the information about the thread options that are shown when **th -** command is used.
hold	If the *ThreadNumber* parameter is not given, holds and displays all user threads. If one or more *ThreadNumber* parameters are given, holds and displays the specified user threads.
unhold	If the *ThreadNumber* parameter is not given, releases and displays all previously held user threads. If one or more *ThreadNumber* parameters are given, releases and displays the specified user threads.
info	If the *ThreadNumber* parameter is not given, displays a long format listing of all user threads. If one or more *ThreadNumber* parameters are given, displays a long format listing the specified user threads. All the previous flags take [-] option. If this option is given, then the thread information displayed is in the second format and in the long form unless the `set $thcomp` option is set.
run	Displays threads which are in the `run` state.
susp	Displays threads which are in the `susp` state.
term	Displays threads which are in the `term` state.
wait	Displays threads which are in the `wait` state.

Examples

1. To display information about threads that are in the wait state, enter:

   ```
   thread  wait
   ```

   The output is similar to:

   ```
   thread  state-k   wchan state-u   k-tid mode held scope function
    $t1     wait            running   17381   u   no   pro  main
    $t3     wait            running    8169   u   no   pro  iothread
   ```

2. To display information about several given threads, enter:

   ```
   thread 1 3 4
   ```

   The output is similar to:

   ```
   thread  state-k   wchan state-u   k-tid mode held scope function
    $t1     wait            running   17381   u   no   pro  main
    $t3     wait            running    8169   u   no   pro  iothread
   >$t4     run             running    9669   u   no   pro  save_thr
   ```

3. To make thread 4 the current thread, enter:

   ```
   thread  current 4
   ```

4. To hold thread number 2, enter:

   ```
   thread  hold 2
   ```

5. To display information about threads that are in the wait state, in the second format, enter:

```
thread wait -
```

The output is similar to:

```
thread m k u h s c  j b  kpl  upl  kpr upr k_tid   u_tid   fl  sta wchan  function
*$t1   u r w n p ed y 0  oth  oth  61   1  0043e5  000001  51  004        main
 $t3   u r w n p ed y 0  oth  oth  61   1  001fe9  000102  51  004        iothread
>$t4   u r r n p ed y 0  oth  oth  61   1  0025c5  000203  50  064        save_thr
```

6. To display information about several given threads in the second format, enter:

```
thread - 1 2 3
```

The output is similar to:

```
thread m k u h s c  j b  kpl  upl  kpr upr k_tid   u_tid   fl  sta wchan function
*$t1   u r w n p ed y 0  oth  oth  61   1  0043e5  000001  51  004       main
 $t3   u r w n p ed y 0  oth  oth  61   1  00fe9   000102  51  004       iothread
>$t4   u r r n p ed y 0  oth  oth  61   1  0025c5  000203  50  064       save_thr
```

See the **attribute** subcommand, the **condition** subcommand, the **mutex** subcommand, the **print** subcommand, the **registers** subcommand, and the **where** subcommand.

Also, see Creating Threads in *General Programming Concepts: Writing and Debugging Programs*.

tls Subcommand

tls map

The **tls** subcommand takes only one flag that it uses to display the TLS initialization template origin and length for each loaded TLS module.

tm_status Subcommand

```
tm_status
```

The **tm_status** subcommand displays the contents of the *$texasr* variable (transaction exception and summary register) and interprets the contents to determine the cause and nature of a transaction failure.

Example

To display and interpret the values that are stored in the *$texasr* variable, enter the following command:

```
(dbx) tm_status
```

An output that is similar to the following example is displayed:

```
REGISTER : TEXASR = 0x100000018C000001
Bit(s)   |Field                    |Meaning
_____________________________________________________________________________
0-7      Failure Code               TM_SIG_DELIVERED | Failed due to signal delivery
7        Failure Persistent         Failure is transient
31       Abort                      Execution of TM instruction caused Abort
32       Suspended                  Failure while in Suspended State
34-35    Privilege                  During Failure process-thread privilege state was 0
36       Failure                    Summary Failure recording has been performed
37       TFIAR (in)exact            TFIAR is exact
38       Rollback Only Transaction  non-ROT tbegin. initiated
52-63    Transaction Level          1
```

tnext Subcommand

tnext [*Number*]

The **tnext** subcommand runs the running thread up to the next source line. The *Number* parameter specifies the number of times the `tnext` subcommand runs. If the *Number* parameter is not specified, `tnext` runs once only. This subcommand can be started only on system-scope threads.

All the threads are run during this operation. To catch breakpoints during this operation, set the `$catchbp` **dbx** variable. If the `$catchbp` variable is set and a breakpoint is reached for another thread, the `tnext` subcommand is not repeated for the remaining number of times.

Examples

1. To continue execution of the running thread up to the next source line, enter:

```
tnext
```

2. To continue execution of the running thread up to the third source line following the current source line, enter:

```
tnext 3
```

See the **tnexti** subcommand. Also, see Debugging Programs Involving Multiple Threads in *General Programming Concepts: Writing and Debugging Programs*.

tnexti Subcommand

tnexti [*Number*]

The **tnexti** subcommand runs the running thread up to the next instruction. The *Number* parameter specifies the number of times the `tnexti` subcommand runs. If the *Number* parameter is not specified, `tnexti` runs once only. This subcommand can be started only on system-scope threads.

All the threads are run during this operation. To catch breakpoints during this operation, set the `$catchbp` **dbx** variable. If the `$catchbp` variable is set and a breakpoint is reached for another thread, the `tnexti` subcommand is not repeated for the remaining number of times.

Examples

1. To continue execution of the running thread up to the next machine instruction, enter:

```
tnexti
```

2. To continue execution of the running thread up to the third machine instruction following the current machine instruction, enter:

```
tnexti 3
```

See the **tnext** subcommand. Also, see Debugging Programs Involving Multiple Threads in *General Programming Concepts: Writing and Debugging Programs*.

trace Subcommand

trace [*SourceLine* | *Expression* **at** *SourceLine* | *Procedure* | [*Variable*] [**at** *SourceLine* | **in** *Procedure*] | **on load** *ModuleName*] [**if** *Condition*]

The **trace** subcommand prints tracing information for the specified procedure, function, source line, expression, or variable when the program runs. The *SourceLine* variable can be specified as an integer or as a file name string followed by a : (colon) and an integer. A condition can be specified. The **dbx** debug program associates a number with each **trace** subcommand. Use the **status** subcommand to view these numbers. Use the **delete** subcommand to turn tracing off. You can enable and disable traces using the **enable** and **disable** subcommands, respectively.

The **trace** subcommand can display tracing information when modules are loaded or unloaded by the debugged process. The optional *ModuleName* parameter can be specified as a single module name, or as a module name paired with a member name in the format:

```
ModuleName(MemberName)
```

If the **on load** flag is used without the *ModuleName* parameter, the **dbx** command traces the load and unload of all modules.

By default, tracing is process-based. To make a thread-based trace, specify the thread in a condition (as in example 8).

Flags

Item	Description
at *SourceLine*	Specifies the source line where the expression being traced is found.
if *Condition*	Specifies a condition for the beginning of the trace. The trace begins only **if** *Condition* is true.
in *Procedure*	Specifies the procedure to use to find the procedure or variable being traced.
on load *ModuleName*	Specifies the load module to be monitored.

Examples

1. To trace each call to the `printf` procedure, enter:

   ```
   trace printf
   ```

2. To trace each execution of line 22 in the `hello.c` file, enter:

   ```
   trace "hello.c":22
   ```

3. To trace changes to the *x* variable within the `main` procedure, enter:

   ```
   trace x in main
   ```

4. To trace the data address `0x2004000`, enter:

   ```
   set $A=0x2004000
   trace $A
   ```

   Note: The **tracei** subcommand is designed to trace addresses.

5. You can restrict the printing of source lines to when the specified *Procedure* is active. You can also specify an optional *Condition* to control when trace information must be produced. For example:

   ```
   (dbx) trace in sub2
   [1] trace in sub2
   (dbx) run
   trace in hellosub.c:  8  printf("%s",s);
   trace in hellosub.c:  9  i = '5';
   trace in hellosub.c:  10  }
   ```

6. You can display a message each time a procedure is called or returned. When a procedure is called, the information includes passed parameters and the name of the calling routine. On a return, the information includes the return value from *Procedure*. For example:

   ```
   (dbx) trace sub
   [1] trace sub
   (dbx) run
   calling sub(s = "hello", a = -1, k = delete) from function main
   returning "hello" from sub
   ```

7. You can print the value of *Expression* when the program reaches the specified source line. The lines number and file are printed, but the source line is not. For example:

```
(dbx) trace x*17 at "hellosub.c":8 if (x > 0)
[1] trace x*17 at "hellosub.c":8 if x > 0
(dbx) run
at line 8 in file "hellosub.c": x*17 = 51

(dbx) trace x
[1] trace x
initially (at line 4 in "hello.c"):  x = 0
after line 17 in "hello.c":  x = 3
```

8. To trace changes to the *x* variable that are made by thread `$t1`, enter the following command:

```
(dbx) trace x if ($running_thread == 1)
```

9. To trace the load or unload of all modules, enter the following command:

```
trace on load
```

10. To trace the load or unload of module `Module`, enter the following command:

```
trace on load "Module"
```

11. To trace the load or unload of member `Member` in module `Module`, enter the following command:

```
trace on load "Module(Member)"
```

Also, see the **tracei** subcommand.

tracehwp Subcommand

tracehwp *Address Size*

The **tracehwp** subcommand sets a hardware watchpoint stop for the specified memory region. The **dbx** debug program prints tracing information when the contents of the region change.

Notes:

1. The success of the **tracehwp** subcommand is hardware dependent. This feature is available only on POWER630 and POWER4 onwards.
2. As a result of the hardware limitation of being able to set only a single watchpoint, an active watchpoint event acts as a conflict when attempting to create another hardware watchpoint event with **stophwp** and **tracehwp**. As such, the previous event must be deleted before creating a new one. Also, since the existence of an active software watchpoint (created by some invocations of the **stop** and **trace** subcommands) negate the performance gains of hardware watchpoints, these types of events also act as conflicts which must be deleted before creating a hardware watchpoint.

Examples

1. To trace each time the contents of the 4 byte memory region starting at address `0x200004e8` change, enter the following command:

```
tracehwp 0x200004e8 4
```

See the **stophwp** subcommand.

tracei Subcommand

tracei [[*Address*] [**at** *Address* | **in** *Procedure*] | *Expression* **at** *Address*] [**if** *Condition*]

The **tracei** subcommand turns on tracing when:

- The contents of the address specified by the *Address* parameter change if the *Address* flag is included.
- The instruction at *Address* is run if the **at** *Address* parameter is specified.

- The procedure specified by *Procedure* is active if the **in** *Procedure* flag is included.
- The condition specified by the *Condition* parameter is true if the **if** *Condition* flag is included.

Flags

Item	Description
at *Address*	Specifies an address. Tracing is enabled when the instruction at this address is run.
if *Condition*	Specifies a condition. Tracing is enabled when this condition is met.
in *Procedure*	Specifies a procedure. Tracing is enabled when this procedure is active.

Examples

1. To trace each instruction executed, enter the following command:

```
tracei
```

2. To trace each time the instruction at address `0x100020f0` is executed, enter the following command:

```
tracei at 0x100020f0
```

3. To trace each time the contents of memory location `0x20004020` change while the `main` procedure is active, enter the following command:

```
tracei 0x20004020 in main
```

4. To trace each time the instruction at address `0x100020f0` is executed by thread `$t4`, enter the following command:

```
tracei at 0x100020f0 if ($running_thread == 4)
```

See the **trace** subcommand. Also, see Debugging at the Machine Level with **dbx** in *General Programming Concepts: Writing and Debugging Programs*.

tskip Subcommand

tskip [*Number*]

The **tskip** subcommand continues the execution of the running thread from the current stopping point. The number of thread-level breakpoints specified by the *Number* parameter is skipped for the running thread. This subcommand can be started for system-scope threads only.

All the other threads are run during this operation, and all breakpoints and watchpoints specified by the user are caught. The execution can cease when any thread hits a breakpoint or watchpoint. Even though the execution started by **tskip** subcommand can stop because of an event for another thread, the **tskip** count specified for the previous thread is still active and the number of thread-level breakpoints specified by the **tskip** count is ignored for that thread when the process continues. When the thread ends, the **tskip** count associated with it is deleted.

Use the **status** subcommand to view the remaining **tskip** count for the threads. Use the **delete** subcommand to delete the remaining **tskip** count for the threads.

Example

To continue execution until the second thread-level breakpoint is encountered starting from the current stopping point for the running thread, enter:

```
tskip 1
```

See the cont subcommand. Also, see Debugging Programs Involving Multiple Threads in *General Programming Concepts: Writing and Debugging Programs*.

tstep Subcommand

tstep [*Number*]

The **tstep** subcommand runs the specified number of source lines from the current source line for the running thread. The *Number* parameter specifies the number of times the `tstep` subcommand runs. If the *Number* parameter is not specified, `tstep` runs once only. This subcommand can be started only on system-scope threads.

All the threads are run during this operation. If `$hold_next` is set, all the threads except the running thread is held.

Note: Use the `$stepignore` variable of the `set` subcommand to control the behavior of the `tstep` subcommand. The `$stepignore` variable enables the `tstep` subcommand to step over large routines for which no debugging information is available.

Examples

1. To continue execution of the running thread up for one source line, enter:

   ```
   tstep
   ```

2. To continue execution of the running thread for five source lines, enter:

   ```
   tstep 5
   ```

3. To prevent the **dbx** program from single-stepping the `printf` function, as illustrated in the example code:

   ```
   60 printf ("hello world /n");
   ```

   enter:

   ```
   set $stepignore="function"; step
   ```

See the cont subcommand, the goto subcommand, **tnext** subcommand, the set subcommand, and the **tstepi** subcommand. Also, see Debugging Programs Involving Multiple Threads in *General Programming Concepts: Writing and Debugging Programs*.

tstepi Subcommand

tstepi [*Number*]

The **tstepi** subcommand runs the specified number of instructions from the current instruction for the running thread. The *Number* parameter specifies the number of times the `tstepi` subcommand runs. If the *Number* parameter is not specified, `tstepi` runs once only. This subcommand can be started only on system-scope threads.

All the threads are run during this operation. If `$hold_next` is set, all the threads except the running thread is held.

Examples

1. To continue execution of the running thread up for one machine instruction, enter:

   ```
   tstepi
   ```

2. To continue execution of the running thread for five machine instructions, enter:

   ```
   tstepi 5
   ```

See the gotoi subcommand, **tnexti** subcommand, and the **tstep** subcommand. Also, see Debugging Programs Involving Multiple Threads in *General Programming Concepts: Writing and Debugging Programs*.

tstop Subcommand

tstop { in *Procedure* | [*Variable*] at *SourceLine* [if *Condition*] } [**for $t***threadnumber*]

The **tstop** subcommand sets a source-level breakpoint stop for a thread and halts the application program when the specified thread reaches the breakpoint. The thread specified must exist at the same time as the creation of the event. The current thread is used if no thread is specified. The specified thread is stopped when any of the following occurs:

- The `if` *Condition* flag is used, and the *Condition* is true.
- The `in` *Procedure* flag is used, and the *Procedure* is called.
- The `at` *SourceLine* flag is used, and the *SourceLine* line number is reached. The *SourceLine* variable can be specified as an integer or as a file name string followed by a colon (:) and an integer.

Thread-level breakpoints can be set on system scope threads only. When a thread-level and a process-level breakpoint are hit at the same time, both the breakpoints are processed and the thread-level breakpoint is ported. When the thread terminates, the events associated with it are deleted.

Flags

Item	Description
at *SourceLine*	Specifies the line number.
for $t *threadnumber*	Specifies the thread number.
if *Condition*	Specifies the condition (for example, `true`).
in *Procedure*	Specifies the procedure to be called.

Examples

1. To stop execution at the first statement in the `func` procedure while running thread 2, enter:

```
tstop in func for $t2
```

2. To stop execution of the current thread when the value of the x variable is changed on line 12 of the execution, enter:

```
tstop x at 12
```

See the **ttrace** subcommand. Also, see Debugging Programs Involving Multiple Threads in *General Programming Concepts: Writing and Debugging Programs*.

tstophwp Subcommand

tstophwp *address size* [**for $t***threadnumber*]

The **tstophwp** subcommand sets a thread-level hardware watchpoint stop for the specified memory region. The program stops when the contents of the region changes while running the specified thread. The thread specified must exist at the same time as the creation of the event. The current thread is used if no thread is specified. The thread-level watchpoint events can be set only for system-scope threads. When the thread terminates, the events associated with it are deleted.

Notes:

1. The success of the `tstophwp` subcommand is hardware-dependent. This feature is available only on POWER630 and POWER4 onwards.
2. As a result of the hardware limitation allowing only a single watchpoint to be set, an active thread watchpoint event acts as a conflict when attempting to create another hardware watchpoint event for the same thread using `tstophwp` and `ttracehwp`. To avoid this, the previous event must be deleted

before creating a new one. Because the existence of an active software watchpoint (created by some invocations of the `stop` and `trace` subcommands) can negate the performance gains of hardware watchpoints, these types of events must also be deleted before creating a hardware watchpoint to avoid conflicts.

3. When a process-level watchpoint exists, a thread having no thread-level watchpoint watches the process watchpoint location. If a thread has a thread-level watchpoint, the thread watches the thread watchpoint location.
4. A thread-level hardware watchpoint and a process-level hardware watchpoint can coexist and do not conflict with each other.
5. If a process-level and a thread-level watchpoint exist for the same address, the process-level watchpoint event is reported.

Flags

Item	Description
for $t *threadnumber*	Specifies the thread number.

Example

To stop the program when thread 2 is running and the contents of the 4-byte memory region starting at address `0x200004e8` change, enter:

```
tstophwp 0x200004e8 4 for $t2
```

See the **ttracehwp** subcommand. Also, see Debugging Programs Involving Multiple Threads in *General Programming Concepts: Writing and Debugging Programs*.

tstopi Subcommand

tstopi { `in` *Procedure* | [*Address*] `at` *Address* [`if` *Condition*] } [**for $t***threadnumber*]

The **tstopi** subcommand sets an instruction-level breakpoint stop for a thread. The thread specified must exist at the same time as the creation of the event. The current thread is used if no thread is specified. The specified thread is stopped when any of the following occurs:

- The `if` *Condition* flag is used, and the *Condition* is true.
- The `in` *Procedure* flag is used, and the *Procedure* is called.
- The `at` *Address* flag is used, and the *Address* is reached.

Thread-level breakpoints can be set on system scope threads only. When a thread-level and a process-level breakpoint are hit at the same time, both the breakpoints are processed and the thread-level breakpoint is reported. When the thread terminates, the events associated with it are deleted.

Flags

Item	Description
at *Address*	Specifies the machine instruction address.
for $t *threadnumber*	Specifies the thread number.
if *Condition*	Specifies the condition.
in *Procedure*	Specifies the procedure to be called.

Example

1. To stop execution at address `0x100020f0` while running thread 2, enter:

```
tstopi at 0x100020f0 for $t2
```

2. To stop execution when the `func` procedure is entered while running the current thread, enter:

```
tstopi in func
```

See the **ttracei** subcommand. Also, see Debugging Programs Involving Multiple Threads in *General Programming Concepts: Writing and Debugging Programs*.

ttrace Subcommand

ttrace { [*Variable*] `at` *SourceLine* | *Procedure* } [`if` *Condition*] [**for $t***threadnumber*]

The **ttrace** subcommand prints tracing information when the specified thread runs for the specified procedure, function, source line, and variable. The *SourceLine* variable can be specified as an integer or as a file name string followed by a colon (:) and an integer. The **dbx** debug program associates a number with each **ttrace** subcommand. Use the `status` subcommand to view these numbers. Use the **delete** subcommand to turn tracing off. You can enable and disable traces using the **enable** and **disable** subcommands, respectively.

The current thread is used if no thread is specified. Thread-level trace can be set only for system-scope threads. The thread specified must exist at the same time as the creation of the event. When the thread ends, the events associated with it are deleted.

Flags

Item	Description
at *SourceLine*	Specifies the source line where the expression being traced is found.
for $t *threadnumber*	Specifies the thread number.
if *Condition*	Specifies a condition for the beginning of the trace. The trace begins only if *Condition* is true.
in *Procedure*	Specifies the procedure to find the procedure or variable being traced.

Examples

1. To trace each call to the `printf` procedure while running thread 2, enter:

```
ttrace printf for $t2
```

2. To trace each execution of line 22 in the `hello.c`/ file while the current thread is running, enter:

```
ttrace "hello.c":22
```

See the **ttracei** subcommand. Also, see Debugging Programs Involving Multiple Threads in *General Programming Concepts: Writing and Debugging Programs*.

ttracei Subcommand

ttracei [*Address*] `at` *Address* [`if` *Condition*] } [**for $t***threadnumber*]

The **ttracei** subcommand turns on tracing for the specified thread when any of the following occurs:

- The `if` *Condition* flag is included, and the *Condition* is true.
- The `at` *Address* flag is specified, and the instruction at *Address* is run.

The current thread is used if no thread is specified. Thread-level trace can be set only for system-scope threads. The thread specified must exist at the time as the creation of the event. When the thread ends, the events associated with it are deleted.

Flags

Item	Description
at *Address*	Specifies an address. Tracing is enabled when the instruction at this address is run.
for $t *threadnumber*	Specifies the thread number.
if *Condition*	Specifies a condition. Tracing is enabled when this condition is met.

Example

1. To trace each time the instruction at address `0x100020f0` is executed while thread 3 is running, enter:

```
tracei at 0x100020f0 for $t3
```

2. To trace each time the instruction at address `0x100020f0` is executed by the current thread, enter:

```
tracei at 0x100020f0
```

See the **ttrace** subcommand. Also, see Debugging Programs Involving Multiple Threads in *General Programming Concepts: Writing and Debugging Programs*.

ttracehwp Subcommand

ttracehwp *address size* [**for $t***threadnumber*]

The**ttracehwp** subcommand sets a thread-level hardware watchpoint trace for the specified memory region. The **dbx** debug program prints tracing information when the contents of the region change while running the specified thread. The thread specified must exist at the same time as the creation of the event. The current thread is used if no thread is specified. The thread-level watchpoint events can be set only for system-scope threads. When the thread terminates, the events associated with it are deleted.

Note:

1. The success of the `ttracehwp` subcommand is hardware dependent. This feature is available only on POWER630 and POWER4 onwards.
2. As a result of the hardware limitation allowing only a single watchpoint to be set, an active thread watchpoint event acts as a conflict when attempting to create another hardware watchpoint event for the same thread using `tstophwp` and `ttracehwp`. To avoid this, the previous event must be deleted before creating a new one. Because the existence of an active software watchpoint (created by some invocations of the `stop` and `trace` subcommands) can negate the performance gains of hardware watchpoints, these types of events must also be deleted before creating a hardware watchpoint to avoid conflicts.
3. When a process-level watchpoint exists, a thread having no thread-level watchpoint watches the process watchpoint location. If a thread has a thread-level watchpoint, the thread watches the thread watchpoint location.
4. A thread-level hardware watchpoint and a process-level hardware watchpoint can coexist and do not conflict with each other.
5. If a process-level and a thread-level watchpoint exist for the same address, the process-level watchpoint event is reported.

Flags

Item	Description
for $t *threadnumber*	Specifies the thread number.

Example

To trace each time the contents of the 4-byte memory region starting at address `0x200004e8` change while running thread 2, enter:

```
ttracehwp 0x200004e8 4 for $t2
```

See the **tstophwp** subcommand. Also, see Debugging Programs Involving Multiple Threads in *General Programming Concepts: Writing and Debugging Programs*.

unalias Subcommand

unalias *Name*

The **unalias** subcommand removes the alias specified by the *Name* parameter.

Example

To remove an alias named `printx`, enter:

```
unalias printx
```

See the **alias** subcommand. Also, see Creating Subcommand Aliases in *General Programming Concepts: Writing and Debugging Programs*.

unset Subcommand

unset *Name*

The **unset** subcommand deletes the **dbx** debug program variable associated with the name specified by the *Name* parameter.

Example

To delete the variable inhibiting the display of floating-point registers, enter:

```
unset $noflregs
```

See the **set** subcommand. Also, see Changing Print Output With Special Debugging Variables in *General Programming Concepts: Writing and Debugging Programs*.

up Subcommand

up [*Count*]

The **up** subcommand moves the current function up the stack *Count* number of levels. The current function is used for resolving names. The default for the *Count* parameter is one.

Examples

1. To move the current function up the stack 2 levels, enter:

   ```
   up 2
   ```

2. To display the current function on the stack, enter:

   ```
   up 0
   ```

See the **down** subcommand. Also, see Changing the Current File or Procedure, Displaying a Stack Trace in *General Programming Concepts: Writing and Debugging Programs*.

use Subcommand

use [{ + | *Directory* | '['*RegularExpression* = *NewPath*']' } ...]

The **use** subcommand sets the list of directories to be searched and path mappings to be applied when the **dbx** debug program looks for source files. If the **use** subcommand is specified without arguments, the current list of directories to be searched and path mappings to be applied are displayed.

The **@** (at-sign) is a special directory that directs the **dbx** program to look at the full-path name information in the object file, if it exists. If you have a relative directory called @ to search, you must use `./@` in the search path.

The **use** subcommand uses the + (plus-sign) to add more directories or mappings to the list of directories to be searched. The + represents the current list of directories and mappings when specified as input to the **use** subcommand. To append a directory or mapping to the end of the current list, the + must be specified before the new directory or mapping. To add a directory to the beginning of the current list, the + must be specified after the new directory or mapping. If you have a directory named +, specify the full-path name for the directory (for example, `./+` or `/tmp/+`).

The **use** subcommand interprets strings enclosed in `[` and `]` (square brackets) which contain an = (equal-sign) as path mappings. These path mappings are used with the special @ directory. They make it easier for the user to express source file locations in the case that entire directory structures of source files are relocated after compilation.

The following rules apply when attempting to locate a source file during debugging:

- Directories in the list are evaluated in the order specified.
- Upon evaluation of a directory in the list, the directory is searched for the specified file. If the file exists in the directory and is readable, this file is used.
- Upon evaluation of the special @ directory, when one or more path mappings are specified, if the *RegularExpression* portion of a path mapping matches the first *n* characters of the full-path name information in the object file and the substitution of the *NewPath* portion of the path mapping yields a readable file, this file is used.
- Upon evaluation of the special @ directory, when either no path mappings are specified or none match, the directory corresponding to the full-path name information is searched. If the file exists in the directory and is readable, this file is used.
- If more than one path mapping yields a readable file, the path mapping whose *RegularExpression* matches the most characters (1 ... n) of the full-path name information (that is, the most specific) is applied and the resulting file is used.
- If more than one path mapping yields a readable file and each path mapping has equal specificity, the path mapping nearest to the beginning of the list is applied and the resulting file is used.

Note: If the special @ directory is not a member of the list, any path mappings that might be specified are ignored.

Examples

1. To change the list of directories to be searched to the current directory (.), the parent directory (..), and the `/tmp` directory, enter:

   ```
   use . .. /tmp
   ```

2. To change the list of directories to be searched to the current directory (.), the directory the source file was located in at compilation time (@), and the `../source` directory, enter:

   ```
   use . @ ../source
   ```

3. To add the `/tmp2` directory to the list of directories to be searched, enter:

```
use + /tmp2
```

4. To add the /tmp3 directory to the beginning of the list of directories to be searched, enter:

```
use /tmp3 +
```

5. To express that source files whose full-path name information begins with /home/developer are now located under /mnt, enter:

```
use + [/home/developer=/mnt]
```

6. To direct the **dbx** program to first look under /latest and then, if the file does not exist there, to look under /stable for files with full-path name information beginning with /home/developer, enter:

```
use + [/home/developer=/latest] [/home/developer=/stable]
```

Also, see the **edit** subcommand and the **list** subcommand.

whatis Subcommand

whatis *Name*

The **whatis** subcommand displays the declaration of *Name*, where the *Name* parameter designates a variable, procedure, or function name, optionally qualified with a block name.

Note: Use the **whatis** subcommand only while running the **dbx** debug program.

Examples

1. To display the declaration of the x variable, enter:

```
whatis x
```

2. To display the declaration of the main procedure, enter:

```
whatis main
```

3. To display the declaration of the x variable within the main function, enter:

```
whatis main.x
```

4. To print the declaration of an enumeration, structure, or union tag, use $$TagName:

```
(dbx) whatis $$status
enum $$status { run, create, delete, suspend };
```

where Subcommand

where [all | *$tthreadumber* [(*startframe endframe*)] ...] [*startframe endframe*] [**>***File*]

The **where** subcommand displays a list of active procedures and functions associated with the frame numbers *startframe* to *endframe*. The numbering of the stack frame starts from the currently active function stack frame (which is always numbered 0). If there are *n* frames, the frame of the main function is numbered n-1. By using the >*File* flag, the output of this subcommand can be redirected to the specified file.

In the multithreaded environment option all displays the stack details for all available threads. The stack details of individual threads are displayed by specifying the thread number along with where subcommand. If start and end frames for individual threads are not specified, stack frames are displayed by the global start and end frame numbers. Command with no options displays the stack frames of current thread.

Flag

Item	Description
>*File*	Redirects output to the specified file.

See the `frame` subcommand, **up** subcommand, and **down** subcommand. Also, see Displaying a Stack Trace in *General Programming Concepts: Writing and Debugging Programs*.

Example

1. To display the stack details of all the threads, enter:

```
where all
```

2. To display the stack details of threads $t1, $t2 and $t3, enter:

```
where $t1 $t2 $t3
```

3. To display the stack details of threads $t2 with stack frames 2 -3, $t1 and $t3 both with stack frames 1-4, enter:

```
where $t1 $t2(2 3) $t3 1 4
```

See the `frame` subcommand, **up** subcommand, and **down** subcommand. Also, see Displaying a Stack Trace in *General Programming Concepts: Writing and Debugging Programs*.

whereis Subcommand

whereis *Identifier*

The **whereis** subcommand displays the full qualifications of all the symbols whose names match the specified identifier. The order in which the symbols print is not significant.

Examples

To display the qualified names of all symbols named x, enter:

```
whereis x
```

Also, see the **which** subcommand.

which Subcommand

which *Identifier*

The **which** subcommand displays the full qualification of the identifier. The full qualification consists of a list of the outer blocks with which the identifier is associated.

Examples

To display the full qualification of the x symbol, enter:

```
which x
```

See the **whereis** subcommand. Also. see Scoping of Names in in *General Programming Concepts: Writing and Debugging Programs*.

Files

Item	Description
`a.out`	Object file; contains object code.
`core`	Contains core dump.

Item	Description
`.dbxinit`	Contains initial commands.

Debugging applications that use transactional memory

For applications that use transactional memory (TM), the most reliable debugging aids are the transaction exception and summary register (*$texasr*), transaction failure handler address register (*$tfhar*), and transaction failure instruction address register (*$tfiar*) variables.

The *$texasr*, *$tfhar*, and *$tfiar* variables can be displayed by using the **print** subcommand, which is similar to displaying the *$iar* variable. However, you cannot manipulate the values in these registers by using the **assign** subcommand.

The *$tfhar* and *$tfiar* variables contain addresses of instructions from the debugged text segment in a transaction failure. Similar to the *$iar* variable, the two most significant bits of the *$tfhar* and *$tfiar* register variables are reserved and cannot be considered for reading the address from these register variables.

Note: When you use the `$tfiar` and `$tfhar` register variables with the **list** subcommand, the **dbx** command fetches the address after excluding the two most significant bits from the contents of these register variables.

```
(dbx) list at $tfiar
```

or

```
(dbx) list at $tfhar
```

The cause of a transaction failure can be determined by using the *$texasr* variable. The **tm_status** subcommand interprets the cause and nature of a transaction failure.

The **dbx** command monitors the occurrence of a transaction and displays the cause of the transaction failure through the following series of messages. These following messages are displayed after you run the **run** subcommand, the **rerun** subcommand, or the **continue** subcommand.

- `Process {PID} may have failed a transaction - $texasr, $tfiar, $tfhar are valid and may be inspected`
- `Process {PID} may have performed a transaction - $texasr, $tfiar, $tfhar, are valid and may be inspected`

  This message is displayed only when the *$trace_good_transaction* internal variable is set.
- `Process {PID} is in Transactional State – debugging efforts through dbx may result in repeated transaction failure or undefined behavior`
- `Process {PID} is in Suspended State – debugging efforts through dbx may result in repeated transaction failure or undefined behavior`

Where `PID` is the process ID of the process that is debugged.

dc Command

Purpose

Provides an interactive desk calculator for doing arbitrary-precision integer arithmetic.

Syntax

dc [*File*]

Description

The **dc** command is an arbitrary-precision arithmetic calculator. The **dc** command takes its input from the *File* parameter or standard input until it reads an end-of-file character. After the **dc** command receives the input, it evaluates the value and writes the evaluation to standard output. It operates on decimal integers, but you can specify an input base, an output base, and a number of fractional digits to be maintained. The **dc** command is structured as a stacking, reverse Polish notation calculation.

The **bc** command is a preprocessor for the **dc** command. It provides infix notation and a syntax similar to the C language, which implements functions and control structures for programs.

Subcommands

Item	Description
c	Cleans the stack: the **dc** command pops all values on the stack.
d	Duplicates the top value on the stack.
f	Displays all values on the stack.
i	Pops the top value on the stack and uses that value as the number radix for further input.
I	Pushes the input base on the top of the stack.
k	Pops the top of the stack and uses that value as a nonnegative scale factor. The appropriate number of places is displayed on output and is maintained during multiplication, division, and exponentiation. The interaction of scale factor, input base, and output base is reasonable if all are changed together.
l*x*	Pushes the value in the register that is represented by the *x* variable on the stack. The register that is represented by the *x* variable is not changed. All registers start with a value of 0.
L*x*	Treats the *x* variable as a stack and pops its top value onto the main stack.
o	Pops the top value on the stack and uses that value as the number radix for further output.
O	Pushes the output base on the top of the stack.
p	Displays the top value on the stack. The top value remains unchanged.
P	Interprets the top of the stack as a string, removes it, and displays it.
q	Exits the program. If the **dc** command is running a string, it pops the recursion level by two.
Q	Pops the top value on the stack and on the string execution level by that value.
s*x*	Pops the top of the stack and stores it in a register named *x*, where the *x* variable can be any character.
S*x*	Treats the *x* variable as a stack. It pops the top of the main stack and pushes that value onto the stack that is represented by the *x* variable.
v	Replaces the top element on the stack by its square root. Any existing fractional part of the option is taken into account, but otherwise, the scale factor is ignored.
x	Treats the top element of the stack as a character string and runs it as a string of **dc** commands.
X	Replaces the number on the top of the stack with its scale factor.

Item	Description
z	Pushes the number of elements in the stack onto the stack.
Z	Replaces the top number in the stack with the number of digits in that number.
Number	Pushes the specified value onto the stack. A *Number* is an unbroken string of the digits 0 through 9. To specify a negative number, precede it with _ (underscore). A number can contain a decimal point.
+ - / * % ^	Adds (+), subtracts (-), multiplies (*), divides (/), remainders (%), or exponentiates (^) the top two values on the stack. The **dc** command pops the top two entries off the stack and pushes the result on the stack in their place. The **dc** command ignores fractional parts of an exponent.
[*String*]	Puts the bracketed *String* parameter onto the top of the stack.
[**=** | **>** | **<**] *x*	Pops the top two elements of the stack and compares them. Evaluates the register that is represented by the *x* variable as if it obeys the stated relation.
!	Interprets the rest of the line as an operating system command.
?	Gets and runs a line of input.
;:	The **bc** command uses these characters for array operations.

Examples

1. To use the **dc** command as a calculator, enter the following command:

```
You:    1 4 / p
System: 0
You:    1 k   [ Keep 1 decimal place ]s.
        1 4 / p
System: 0.2
You:    3 k   [ Keep 3 decimal places ]s.
        1 4 / p
System: 0.250
You:    16 63 5 / + p
System: 28.600
You:    16 63 5 + / p
System: 0.235
```

   Comments can be used in the **dc** command as in the example. Comments are enclosed in brackets and can be followed the `s.` character. The comments in the format `[Comment]s.` are ignored by the **dc** command. Only those comments that are enclosed in brackets are stored on the top of the stack.

   When you enter the **dc** command expressions directly from the keyboard, press Ctrl-D to end the **bc** command session and return to the shell command line.

2. To load and run a **dc** program file, enter the following command:

```
You:    dc prog.dc
        5 lf x p [ 5 factorial ]s.
System: 120
You:    10 lf x p [ 10 factorial ]s.
System: 3628800
```

   This entry interprets the **dc** program saved in the `prog.c` program file, then reads from the workstation keyboard. The `lf x` evaluates the function that is stored in register `f`, which can be defined in the `prog.c` program file as:

```
[ f: compute the factorial of n ]s.
[ (n = the top of the stack) ]s.
[ If 1>n do b; If 1<n do r ]s.
 [d 1 >b d 1 <r] sf
[ Return f(n) = 1 ]s.
 [d - 1 +] sb
```

```
[ Return f(n) = n * f(n-1) ]s.
 [d 1 - lf x *] sr
```

You can create **dc** program files with any text editor or with the **-c** (compile) flag of the **bc** command. When you enter the **dc** command expressions directly from the keyboard, press Ctrl-D to end the **bc** command session and return to the shell command line.

Files

Item	Description
/usr/bin/dc	Contains the **dc** command.

dcp Command

Purpose

Runs commands concurrently on multiple nodes and hardware devices.

Syntax

dcp [**-h**] [**-V**] [**-q**] [**-a**] [**--all-nodes** *context_list*] [**-A**] [**--all-devices** *context_list*] [**-n** *node_list*] [**-N** *nodegroups*] [**-d** *device_list*] [**-D** *devicegroups*] [**-C** *context*] [**-f** *fanout*] [**-l** *user_ID*] [**-o** *node_options*] [**-O** *device_options*] [**-p**] [**-P**] [**-Q**] [**-r** *node_remote_copy*] [**--device-rcp** *device_remote_copy*] [**-R**] [**-t** *timeout*] [**-X** *env_list*] [**-T**] [**-v**] *source_file... target_path*

Description

The **dcp** command concurrently copies files to or from remote target nodes, hardware devices, or both. Targets can be selected from multiple contexts. A context is a target database that contains definitions of nodes and devices, such as NIM. The **dcp** command issues a remote copy command for each node or device specified. When files are pulled from a target, they are placed into the *target_path* with the name of the remote node or device that is appended to the copied *source_file* name. The **/usr/bin/rcp** command is the model for syntax and security. The **dcp** command is a DSM Distributed Shell Utility. The configuration and environmental settings for **dsh** impact the behavior of **dcp**. See the **dsh** command for more details.

Parameters

Item	Description
TARGET CONTEXT	Target context specification is identical for the **dcp** and **dsh** commands. See target context in the **dsh** man page for details on specifying contexts for the **dcp** command.
TARGET SPECIFICATION	Target specification is identical for the **dcp** and **dsh** commands. See the **dsh** man page for details on specifying targets for the **dcp** command.
TARGET LISTS	Target list syntax is identical for the **dcp** and **dsh** commands.
REMOTE USER	A *user_ID* can be specified for the remote copy command. Remote user specification is identical for the **dcp** and **dsh** commands.

Item	Description
REMOTE COPY COMMAND	The **dcp** command uses a configurable remote copy command to run remote commands on remote targets. Support is explicitly provided for AIX Remote Shell **rcp** command, the **OpenSSH scp** command and the **rsync** command. For node targets, the remote copy command is determined by using the parameters in the order of precedence: 1. The **-r** flag. 2. The **DCP_NODE_RCP** environment variable. 3. The **/usr/bin/rcp** command. For device targets, the remote shell is determined by the following order of precedence: 1. The **--device-rcp** flag. 2. The **DCP_DEVICE_RCP** environment variable. 3. The default device remote copy command as defined by the target context. 4. The **RemoteCopyCmd** attribute that is defined for the device target. The remote copy command is specified with a command-line flag or environment variable by using the following syntax: `[context:]path[,[context:]path]...` where *path* is the path to the remote copy command, and *context:* identifies the remote copy command context to use for copying files. A remote copy command path that is specified without a context applies to all other contexts where an explicit remote copy command path is not specified in the list. The remote copy command options can be configured by using command-line flags or environment variables. For node targets, the remote copy command options are determined by the following order of precedence: 1. The **-o** flag. 2. The **DCP_NODE_OPTS** environment variable. For device targets, the remote copy command options are determined by the following order of precedence: 1. The **-O** flag. 2. The **DCP_DEVICE_OPTS** environment variable. The remote copy command options are specified by using the following syntax: `[context:]"options"[, [context:]"options"]...` where *options* are the remote copy command options, and *context:* identifies the remote shell options context to use for copying files. Options that are specified without a context apply to all other contexts where explicit options have not been specified in the list. The options must be specified within double quotation marks ("") to distinguish them from the **dcp** options.
COMMAND EXECUTION	Specifies concurrent remote copy command processes (fanout) that can be specified with the **-f** flag or the **DSH_FANOUT** environment variable. The fanout is only restricted by the number of remote shell commands that can be run in parallel. You can experiment with the **DSH_FANOUT** value on your management server to see whether higher values are appropriate. A timeout value for remote copy command execution can be specified with the **-t** flag or **DSH_TIMEOUT** environment variable. If any remote target does not respond within the timeout value, the **dcp** command displays an error message and exits. The **-T** flag provides diagnostic trace information for **dcp** command execution. Default settings and the actual remote copy commands that are run to the remote targets are displayed. The **dcp** command can be run silently by using the **-Q** flag; no target standard output or standard error is displayed. The parameters for this variable follow: ***source_file...*** Specifies the complete path for the file to be copied to or from the target. Multiple files can be specified. When used with the **-R** flag, only a single directory can be specified. When used with the **-P** flag, only a single file can be specified. ***target_path*** Specifies the complete path to copy one or more *source_file* files to on the target. If the **-P** flag is specified, the *target_path* is the local host location for the copied files. The remote file directory structure is re-created under *target_path* and the remote target name is appended to the copied *source_file* name in the *target_path* directory.

Keywords

Item	Description
-a	Includes in the target list all nodes that are defined in the default context. The default context can be set by using the **-C** flag or the **DSH_CONTEXT** environment variable.
-A	Includes in the target list all devices that are defined in the default context. The default context can be set by using the **-C** flag or the **DSH_CONTEXT** environment variable. This flag is disabled on HMCs.
--all-devices *context_list*	Includes in the target list all devices that are defined in the contexts that are listed in *context_list*. The default context is not implicitly included in this list. This flag is disabled on HMC.
--all-nodes *context_list*	Includes in the target list all nodes that are defined in the contexts that are listed in *context_list*. The default context is not implicitly included in this list.
-C	Specifies the full path of the remote copy command that is used to copy files to or from device targets. A remote copy command for a specific context can be defined by including `context:` before the path.

Item	Description
--context*context*	Specifies the default context to use when the **dcp** command resolves target names. The context value must correspond to a valid context extension module in the `/opt/ibm/sysmgt/dsm/pm/Context` directory.
--device-rcp *device_remote_copy*	Starts the audit subsystem. The *device_remote_copy* syntax follows: `[context:]path[,[context:]path]...` This flag is disabled on HMCs. This keyword reads the instructions in the configuration files and determines the remote shell for device targets.
-d | **--devices** *device_list*	Specifies a list of device targets to include in the target list. The *device_list* syntax is: `[context:] [user_ID@] device_name[,\` `[context:][user_ID@]device_name]...` This flag is disabled on HMCs.
-D | **--devicegroups** *devicegroups*	Includes in the target list all devices that are defined in the device groups that are specified in the *devicegroups* list. The *devicegroups* syntax follow: `[context:] [user_ID@]devicegroup[,\` `[context:] [user_ID@]devicegroup]...` This flag is disabled on HMCs.
-f | **--fanout** *fanout*	Specifies a fanout value for the maximum number of concurrently running remote shell processes. Sequential execution can be specified by indicating a fanout value of 1. If this flag is omitted, the default fanout value of 64 is used.
-l | **--user** *user_ID*	Specifies a remote user name to use for remote copy execution.
-h | **--help**	Displays command usage information.
-n | **--nodes** *node_list*	Specifies a list of node targets to include in the target list. The *node_list* syntax follows: `[context:] [user_ID@]node_name[,\` `[context:] [user_ID@]node_name]...`
-o | **--node-options** *node_options*	Specifies options to pass to the remote copy command for node targets. The options must be specified within double quotation marks to distinguish them from the **dcp** command flags. Options for nodes in a specific context can be defined by including `context:` before the option list. The syntax of *node_options* follows: `[context:]"options"[, [context:]"options"]...`
-N | **--nodegroups** *nodegroups*	Includes all nodes in the target list that are defined in the node groups that are specified in the *nodegroups* list. The syntax of *nodegroups* follows: `[context:] [user_ID@]nodegroup[,\` `[context:] [user_ID@]nodegroup]...`
-O --device-options *device_options*	Specifies options to pass to the remote copy command for device targets. The options must be specified within double quotation marks to distinguish them from the **dcp** command flags. Options for devices in a specific context can be defined by including `context:` before the option list. The syntax of *device_options* follows: `[context:]"options"[,[context:]"options"]...` This flag is disabled on HMCs.
-p | --preserve	Preserves the source file characteristics as implemented by the configured remote copy command.
-P | --pull	Pulls (copies) the files from the targets and places them in the *target_path* directory on the local host. The *target_path* must be a directory. Files that are pulled from remote machines have `_target` appended to the file name to distinguish between them. When the **-P** flag is used with the **-R** flag, `_target` is appended to the directory. Only one file per invocation of the **dcp -P | --pull** command can be pulled from the specified targets.
-Q	Runs the **dcp** command silently such that no target standard output or standard error is displayed.
-q | --show-config	Displays the current environment settings relevant to all **dsh** utility commands. This flag includes the values of all environment variables and settings for all currently installed and valid contexts. Each setting is prefixed with `context:` to identify the source context of the setting.

Item	Description
-r | --node-rcp *node_remote_copy*	Specifies the full path of the remote copy command that is used to copy files to or from node targets. A remote copy command for a specific context can be defined by including `context:` before the path. The *node_remote_copy* syntax follows: `[context:]path[,[context:]path]...` If *path* contains **rsync**, it is assumed that the **rsync** command performs the remote copy.
-R | --recursive	Recursively copies files from a local directory to the remote targets, or when specified with the **-P** flag. It recursively pulls (copies) files from a remote directory to the local host. A single source directory can be specified by using the *source_file* parameter.
-t | --timeout *timeout*	Specifies the time, in seconds, to wait for the remote copy command to finish each remote target. If a target does not respond within the timeout value, the **dcp** command displays an error message and stops the remote copy process for the remote target. If not specified, the **dcp** command waits indefinitely for the remote copy process to finish each target.
-T | --trace	Activates the trace mode. Sends the **dcp** command diagnostic messages to standard output.
-v | --verify	Verifies each target before it runs any remote commands on the target. If a target is not responding, remote command execution for the target is canceled.
-X *env_list*	Ignores the **dcp** command environment variables. This option accepts an argument, which is a comma-separated list of environment variable names, that must not be ignored. If there is no argument to this option, or the argument is an empty string, all the **dcp** environment variables are not accepted.
-V | **--version**	Displays version information for the **dcp** command environment variables. **DSH_CONTEXT** Specifies the default context to use when it resolves targets. This variable is overridden by the **-C** flag. **DSH_DEVICE_LIST** Specifies a file that contains a list of device targets. This variable is overridden by the **-d** flag. This environment variable is ignored on HMCs. **DCP_DEVICE_OPTS** Specifies the options to use for the remote shell command with device targets only. This variable is overridden by the **-0** flag. This environment variable is ignored on HMCs. **DCP_DEVICE_RCP** Specifies the full path of the remote copy command that is used to copy files to or from device targets. This variable is overridden by the **--device-rcp** flag. This environment variable is ignored on HMCs. **DSH_FANOUT** Specifies the fanout value. This variable is overridden by the **-f** flag. **DCP_NODE_OPTS** Specifies the options to use for the remote copy command with node targets only. This variable is overridden by the **-o** flag. **DCP_NODE_RCP** Specifies the full path of the remote command that is used to copy files to and from node targets. This variable is overridden by the **-r** flag. **DSH_NODE_LIST** Specifies a file that contains a list of node targets. This variable is overridden by the **-n** flag. This variable has replaced the **WCOLL. DSH_NODEGROUP_PATH** variable. The **DSH_NODE_LIST** variable also specifies a colon-separated list of directories that contain node group files for the **dsh** context. When the **-a** flag is specified in the **dsh** context, a list of unique node names is collected from all node group files in the path. **DSH_TIMEOUT** Specifies the time, in seconds, to wait for output from each remote target. This variable is overridden by the **-t** flag. **RSYNC_RSH** This **rsync** environment variable specifies the remote shell to be used as the transport for the **rsync** command. Exit status and exit values for each remote copy command execution are displayed in messages from the **dcp** command, if the remote copy command exit value is non-zero. A non-zero return code from a remote copy command indicates that an error was encountered during the remote copy. If a remote copy command encounters an error, execution of the remote copy on that target is bypassed. The **dcp** command exit code is 0 if the **dcp** command ran without errors and all remote copy commands finished with exit codes of 0. If internal **dcp** command errors occur or the remote copy commands do not complete successfully, the **dcp** command exit value is greater than 0. The exit value is increased by 1 for each successive instance of an unsuccessful remote copy command execution. Security: The **dcp** command has no security configuration requirements. All remote command security requirements (configuration, authentication, and authorization) are imposed by the underlying remote command that is configured for the **dcp** command. Authentication and authorization are assumed to be configured between the local host and remote targets. Interactive password prompting is not supported; execution is bypassed and an error is displayed for a remote target if password prompting occurs. The security configurations as they pertain to the remote environment and remote shell command are user-defined. When `/usr/bin/rcp` is configured as your remote command by using Kerberos Version 5, you must first run the Kerberos **kinit** command to obtain a ticket-granting ticket. You must also ensure that your Kerberos principal is in the `.k5login` file in the remote `user&csqg;s` home directory on the targets.

Examples

1. To copy the /tmp/etc/hosts file from the local host to the /etc directory on node3, node4, node5, and to user gregb on device16 in the NIM context, enter the following command:

```
dcp -n node3-node5 -d NIM:gregb@device16 /tmp/etc/hosts /etc:
```

2. To copy the /etc/hosts file from all managed nodes in the cluster to the /tmp/hosts.dir directory on the local host, enter the following command:

```
dcp -aP /etc/hosts /tmp/hosts.dir
```

   A suffix that specifies the name of the target is appended to each file name. The contents of the /tmp/hosts.dir directory are like:

hosts._node1	hosts._node4	hosts._node7
hosts._node2	hosts._node5	hosts._node8
hosts._node3	hosts._node6	

3. To copy the /var/log/testlogdir directory from all targets in NodeGroup1 in the NIM context and DeviceGroup4 in the **dsh** context, with a fanout of 12, and save each directory on the local host as /var/log._target, enter the following command:

```
dcp -C DSH -N NIM:NodeGroup1 -D DeviceGroup 4 -f 12 \ -RP /var/log/testlogdir /var/log
```

4. To copy /localnode/smallfile and /tmp/bigfile to /tmp on node1 by using the **rsync** command, enter the following command:

```
RSYNC_RSH=/usr/bin/ssh; dcp -r /usr/bin/rsync -o "-z" \ -n node1 /localnode/smallfile /tmp/
bigfile /tmp
```

   This command uses **rsync** with the **RSYNC_RSH** environment variable and the **-z** flag on **rsync**.

5. To copy the /etc/hosts file from the local host to all the nodes in the cluster, and ignore all the **dcp** environment variable, enter the following command:

```
dcp -X -a /etc/hosts /etc/hosts
```

6. To copy the /etc/hosts file from node1 and node2 to the /tmp/hosts.dir directory on the local host and ignore all the **dcp** environment variables except the **DCP_NODE_OPTS**, enter the following command:

```
dcp -n node1,node2 -P -X 'DCP_NODE_OPTS' /etc/hosts /tmp/hosts.dir
```

dd Command

Purpose

Converts and copies a file.

Syntax

dd [**bs=***BlockSize*][**cbs=***BlockSize*]

[**conv=** [ascii |block|ebcdic | ibm | unblock]

[lcase | ucase] [iblock]

[noerror] [swab] [sync]

[oblock] [notrunc]][**count=**

InputBlocks] [**files=***InputFiles*] [**fskip=**

SkipEOFs] [**ibs=***InputBlockSize*] [**if=**

InFile] [**obs=***OutputBlockSize*] [**of=**

OutFile] [**seek=***RecordNumber*] [**skip=**

SkipInputBlocks][**span=***yes*|*no*]

dd [*Option*=*Value*]

Description

The **dd** command reads the *InFile* parameter or standard input, does the specified conversions, then copies the converted data to the *OutFile* parameter or standard output. The input and output block size can be specified to take advantage of raw physical I/O.

Note: The term *Block* refers to the quantity of data read or written by the **dd** command in one operation and is not necessarily the same size as a disk block.

Where sizes are specified, a number of bytes is expected. A number ending with **w**, **b**, or **k** specifies multiplication by 2, 512, or 1024 respectively; a pair of numbers separated by an **x** or an ***** (asterisk) indicates a product. The count parameter expects the number of blocks, *not* the number of bytes, to be copied.

The character-set mappings associated with the **conv=ascii** and **conv=ebcdic** flags are complementary operations. These flags map between ASCII characters and the subset of EBCDIC characters found on most workstations and keypunches.

Use the **cbs** parameter value if specifying any of the **block**, **unblock**, **ascii**, **ebcdic**, or **ibm** conversions. If **unblock** or **ascii** parameters are specified, then the **dd** command performs a fixed-length to variable-length conversion. Otherwise it performs a conversion from variable-length to fixed-length. The **cbs** parameter determines the fixed-length.

Attention: If the **cbs** parameter value is specified smaller than the smallest input block, the converted block is truncated.

After it finishes, the **dd** command reports the number of whole and partial input and output blocks.

Note:

1. Usually, you need only write access to the output file. However, when the output file is not on a direct-access device and you use the **seek** flag, you also need read access to the file.
2. The **dd** command inserts new-line characters only when converting with the **conv=ascii** or **conv=unblock** flags set; it pads only when converting with the **conv=ebcdic**, **conv=ibm**, or **conv=block** flags set.
3. Use the **backup**, **tar**, or **cpio** command instead of the **dd** command whenever possible to copy files to tape. These commands are designed for use with tape devices. For more information on using tape devices, see the **rmt** special file.
4. The block size values specified with the **bs**, **ibs** and **obs** flags must always be a multiple of the physical block size for the media being used.
5. When the **conv=sync** flag is specified, the **dd** command pads any partial input blocks with nulls. Thus, the **dd** command inserts nulls into the middle of the data stream if any of the reads do not receive a full block of data (as specified by the **ibs** flag). This is a common occurrence when reading from pipes.
6. If the **bs** flag is specified by itself and no conversions other than **sync**, **noerror** or **notrunc** are specified, then the data from each input block will be written as a separate output block; if the read returns less than a full block and **sync** is not specified, then the resulting output block will be the same size as the input block. If the **bs** flag is not specified, or a conversion other than **sync**, **noerror** or **notrunc** is specified, then the input will be processed and collected into fullsized output blocks until the end of input is reached.

Spanning across devices

The **dd** can be made to span across devices if the input file is larger than the output device physical size.

Note: Care has to be taken when specifying the block size *bs* as exact multiple of the physical size of the device because improper block size will result in data inconsistency, or overlap.

The spanning of **dd** across devices will not occur if either one of the InFile or the OutFile parameter is stdin or stdout.

Spanning will occur in such a way that **dd** will prompt for next device during write if the output device is full. During read from the input device, **dd** will prompt for next device if the data is completely read from the input device even when the device has not reached the end. In this case it would be required to press 'n' to quit.

Flags

Item	Description
bs=*BlockSize*	Specifies both the input and output block size, superseding the **ibs** and **obs** flags. The block size values specified with the **bs** flag must always be a multiple of the physical block size for the media being used.
cbs=*BlockSize*	Specifies the conversion block size for variable-length to fixed-length and fixed-length to variable-length conversions, such as **conv=block**.
count=*InputBlocks*	Copies only the number of input blocks specified by the *InputBlocks* variable.

Item	Description
conv= *Conversion*,....	Specifies one or more conversion options. Multiple conversions should be separated by commas. The following list describes the possible options: **ascii** Converts EBCDIC to ASCII. This option is incompatible with the **ebcdic**, **ibm**, **block**, and **unblock** options. **block** Converts variable-length records to fixed-length. The length is determined by the conversion block size (cbs). This option is incompatible with the **ascii**, **ebcdic**, **ibm**, and **unblock** options. **ebcdic** Converts ASCII to standard EBCDIC. This option is incompatible with the **ascii**, **ibm**, **block**, and **unblock** options. **ibm** Converts ASCII to an IBM version of EBCDIC. This option is incompatible with the **ascii**, **ebcdic**, **block**, and **unblock** options. **iblock, oblock** Minimize data loss resulting from a read or write error on direct access devices. If you specify the **iblock** variable and an error occurs during a block read (where the block size is 512 or the size specified by the **ibs=***InputBlockSize* variable), the **dd** command attempts to reread the data block in smaller size units. If the **dd** command can determine the sector size of the input device, it reads the damaged block one sector at a time. Otherwise, it reads it 512 bytes at a time. The input block size (**ibs**) must be a multiple of this retry size. This option contains data loss associated with a read error to a single sector. The **oblock** conversion works similarly on output. **lcase** Makes all alphabetic characters lowercase. **noerror** Does not stop processing on an error. **notrunc** Does not truncate the output file. Instead, blocks not explicitly written to output are preserved. **ucase** Makes all alphabetic characters uppercase. **swab** Swaps every pair of bytes. **sync** Pads every input block to the **ibs** value. **unblock** Converts fixed-length blocks to variable-length. The length is determined by the conversion block size (cbs). This option is incompatible with the **ascii**, **ebcdic**, **ibm**, and **block** options.
files=*InputFiles*	Copies the number of files specified by the *InputFiles* variable value of input files before ending (makes sense only where input is a magnetic tape or similar device).
fskip=*SkipEOFs*	Skips past the number of end-of-file characters specified by the *SkipEOFs* variable before starting to copy; this *SkipEOFs* variable is useful for positioning on multifile magnetic tapes.

Item	Description
ibs=*InputBlockSize*	Specifies the input-block size; the default is 512 bytes or one block. The block-size values specified with the **ibs** flag must always be a multiple of the physical block size for the media being used.
if=*InFile*	Specifies the input file name; standard input is the default.
obs=*OutputBlockSize*	Specifies the output-block size; the default is 512 bytes or one block. The block size values specified with the **obs** flag must always be a multiple of the physical block size for the media being used.
of=*OutFile*	Specifies the output file name; standard output is the default.
seek=*RecordNumber*	Seeks the record specified by the *RecordNumber* variable from the beginning of output file before copying.
skip=*SkipInputBlocks*	Skips the specified *SkipInputBlocks* value of input blocks before starting to copy.
span=*yes*/*no*	Allows spanning across devices if specified `yes` and works as default if specified as no. See Spanning Across Devices, for more information..

Exit Status

This command returns the following exit values:

Item	Description
0	The input file was copied successfully.
>0	An error occurred.

Examples

1. To convert an ASCII text file to EBCDIC, type:

   ```
   dd  if=text.ascii  of=text.ebcdic  conv=ebcdic
   ```

   This command converts the `text.ascii` file to EBCDIC representation, storing the EBCDIC version in the `text.ebcdic` file.

   Note: When you specify the **conv=ebcdic** parameter, the **dd** command converts the ASCII ^ (circumflex) character to an unused EBCDIC character (9A hexadecimal), and the ASCII ~ (tilde) to the EBCDIC ^ (NOT symbol).
2. To convert the variable-length record ASCII file **/etc/passwd** to a file of 132-byte fixed-length EBCDIC records, type:

   ```
   dd  if=/etc/passwd  cbs=132  conv=ebcdic  of=/tmp/passwd.ebcdic
   ```
3. To convert the 132-byte-per-record EBCDIC file to variable-length ASCII lines in lowercase, type:

   ```
   dd  if=/tmp/passwd.ebcdic  cbs=132  conv=ascii  of=/tmp/passwd.ascii
   ```
4. To convert the variable-length record ASCII file **/etc/passwd** to a file of 132-byte fixed-length records in the IBM version of EBCDIC, type:

   ```
   dd  if=/etc/passwd  cbs=132  conv=ibm  of=/tmp/passwd.ibm
   ```
5. To copy blocks from a tape with 1KB blocks to another tape using 2KB blocks, type:

   ```
   dd  if=/dev/rmt0  ibs=1024  obs=2048  of=/dev/rmt1
   ```
6. To use the **dd** command as a filter, type:

   ```
   ls -l | dd  conv=ucase
   ```

This command displays a long listing of the current directory in uppercase.

Note: The performance of the **dd** command and **cpio** command to the 9348 Magnetic Tape Unit Model 12 can be improved by changing the default block size. To change the block size, use the **chdev** command in the following way:

```
chdev -l Device_name -a block_size=32k
```

7. To perform efficient transfers to 3.5-inch 1.4MB diskette using 36 blocks of 512 bytes, type:

   dd if=Filename of=/dev/rfd0 bs=36b conv=sync

   This command writes the value of the *Filename* parameter to the diskette device a cylinder at a time. The conv=sync is required when reading from disk and when the file size is not a multiple of the diskette block size. Do not try this if the input to the **dd** command is a pipe instead of a file, it will pad most of the input with nulls instead of just the last block.
8. To copy blocks from a input file with block size set to 720b blocks into a 1.44MB size diskette type:

   ```
   dd if=testfile of=/dev/fd0 bs=720b conv=sync
   ```

   Note: If the input file is larger than the physical size of the output device then dd will prompt you for another device.
9. To copy blocks from a input file with block size set to 32k blocks to a tape type:

   ```
   dd if=inputfile of=/dev/rmt0 bs=32k conv=sync
   ```
10. To copy blocks of data from tape to a file in the current directory with block size set to 32k blocks type as follows:

    ```
    dd if=/dev/rmt0 of=outfile bs=32k conv=sync
    ```
11. To copy blocks from an input file with block size set to 720b, onto a 1.44MB size diskette, enter:

    ```
    dd if=testfile of=/dev/fd0 bs=720b conv=sync span=yes
    ```

    Note: If the input file is larger than the physical size of the output device, then **dd** will prompt you for another device.
12. To copy blocks from an input file with block size set to 32k, to a tape, enter:

    ```
    dd if=inputfile of=/dev/rmt0 bs=32k conv=sync span=yes
    ```
13. To copy blocks of data from tape with block size set to 32k, to a file in the current directory, enter:

    ```
    dd if=dev/rmt0 of=outfile bs=32k conv=sync span=yes
    ```

Files

Item	Description
/usr/bin/dd	Contains the **dd** command.

defif Method

Purpose

Defines a network interface in the configuration database.

Syntax

defif [**-c** *Class* **-s** *Subclass*] **-t** *Type*

Description

The **defif** method defines the specified instance of a network interface. It only defines interfaces for currently configured adapters. To define the specified instance, the **defif** method does the following:

1. Creates a customized interface instance in the configuration database.
2. Derives the logical name of the interface instance.
3. Retrieves the predefined attributes.
4. Updates the Customized Dependency object class to reflect dependencies of the defined interface instance.
5. Sets the status flag of the interface instance to **defined**.

Flags

Item	Description
-c *Class*	Specifies the interface class to be defined. The valid value is **if**.
-s *Subclass*	Specifies the subclass of interface to be defined. Valid values are: **TR** Token-ring **EN** Ethernet **SL** Slip **XT** X.25 **LO** Loopback
-t *Type*	Specifies the type of interface to be defined. Valid values are: **tr** Token-ring **en** Ethernet **sl** Slip **ie3** IEEE 802.3 Ethernet **lo** Loopback **xt** X.25

Examples

To define a token-ring network interface instance, enter the method in the following format:

```
defif -t tr
```

definet Method

Purpose

Defines an inet instance in the system configuration database.

Syntax

definet [**-c** *Class*]

Description

The **definet** method creates an object in the ODM configuration database specifying the customized attributes of the inet instance. It performs the following operations:

1. Creates a customized inet instance.
2. Sets the status flag of the inet instance to defined.

This method is called by the **mkdev** high-level command and is not meant to be issued on the command line.

Note: The **definet** method is a programming tool and should not be executed from the command line.

Flags

Item	Description
-c *Class*	Specifies the inet instance to be defined. The only valid value for the *Class* variable is **tcpip**.

Examples

To define the inet0 instance, issue the following method:

```
definet
```

defragfs Command

Purpose

Increases a file system's contiguous free space.

Syntax

defragfs [**-q** | **-r** | **-s**] [**-f** [**-v**] [**-y**]] { *Device* | *FileSystem* }

Description

The **defragfs** command increases a file system's contiguous free space by reorganizing allocations to be contiguous rather than scattered across the disk. The file system to be defragmented can be specified with the *Device* variable, which is the path name of the logical volume (for example, **/dev/hd4**). It can also be specified with the *FileSystem* variable, which is the mount point in the **/etc/filesystems** file.

The **defragfs** command is intended for fragmented and compressed file systems. However, you can use the **defragfs** command to increase contiguous free space in nonfragmented file systems.

You must mount the file system read-write for this command to run successfully. Using the **-q** flag, the **-r** flag or the **-s** flag generates a fragmentation report. These flags do not alter the file system.

The **defragfs** command is slow against a Enhanced Journaled File System (JFS2) file system with a snapshot due to the amount of data that must be copied into snapshot storage object. The **defragfs** command issues a warning message if there are snapshots. The **snapshot** command can be used to delete the snapshots and then used again to create a new snapshot after the **defragfs** command completes.

On a JFS2 file system, you can specify the **-f** flag with the **defragfs** command to defragment the file system by relocating data extents to be adjacent, and then combining them. Additionally, if you specify the **-v** flag, the **defragfs** command also displays the fragmentation of the file system before and after running the **defragfs** command. The **-f** , **-v**, and **-y** flags can only be used on a JFS2 file system. The **-v** flag is compatible only with the **-f** flag.

The **defragfs** command takes more time to run if you use with the **-f** flag. It is recommended the **defragfs** command be run during a maintenance window.

Any file system activity might decrease the performance of the defragmentation process.

The **defragfs** command might not significantly improve performance of a file system in which the logical volume is located in part or completely on Solid-State Drives (SSDs).

The **defragfs** command cannot run if internal snapshots exist in the system. The **defragfs** command issues a warning message if external snapshots exist in the system unless the **defragfs** command is run with the **-f** flag. If the **defragfs** command is run with the **-f** flag, the **defragfs** command cannot be run with external snapshots. The **defragfs** command takes time to run on a JFS2 file system with a snapshot because of the amount of data that must be copied into the snapshot storage object. The **snapshot** command can be used to delete the snapshots and then the **snapshot** command can be used again to create a new snapshot after the **defragfs** command completes.

The **defragfs** command shows better performance if run on a file system that does not share a log volume with other file systems. If the **defragfs** command is run on a file system that shares a log volume with other file systems, the **defragfs** command displays a warning and asks for confirmation. If you run the **defragfs** command with the **-y** flag, it suppresses the warning. The **-y** flag is only compatible with the **-f** flag.

Flags

Item	Description
-f	Relocates and combines data extents for each file in the file system. This process prioritizes file organization over file system free space contiguity.
-q	Reports the current state of the file system.
-r	Reports the current state of the file system and the state that would result if the **defragfs** command is run without either the **-q**, **-r** or **-s** flag.
-s	Reports the fragmentation in the file system. This option causes **defragfs** to pass through meta data in the file system which may result in degraded performance.
-v	Displays the fragmentation percentage of file system at the start and at the end of the defragmentation operation.
-y	Suppresses warning message that is displayed by the **defragfs** command when multiple file systems are currently mounted by using the same log volume. When the warning messages are suppressed, the **defragfs** command operation continues without any interruption.

Note: The **-v** and the **-y** flag can only be used with the **-f** flag.

Output

On a JFS file system, the definitions for the messages reported by the **defragfs** command are as follows:

Number of free fragments
The number of free fragments in the file system.

Number of allocated fragments
The number of allocated fragments in the file system.

Number of free spaces shorter than a block
The number of free spaces within the file system that are shorter than a block. A free space is a set of contiguous fragments that are not allocated.

Number of free fragments in short free spaces
The total number of fragments in all the short free spaces. A short free space is one that is shorter than a block.

Number of fragments moved
The total number of fragments moved.

Number of logical blocks moved
The total number of logical blocks moved.

Number of allocation attempts
The number of times free fragments were reallocated.

Number of exact matches
The number of times the fragments that are moved would fit exactly in some free space.

Total number of fragments
The total number of fragments in the file system.

Number of fragments that may be migrated
The number of fragments that may be moved during defragmentation.

File system is in percent fragmented
Shows to what extent the file system is fragmented in percentage.

On a JFS2 file system the definitions for the messages reported by the **defragfs** command are as follows:

Total allocation groups
The number of allocation groups in the file system. Allocation groups divide the space on a file system into chunks. Allocation groups allow JFS2 resource allocation policies to use well known methods for achieving good I/O performance.

Allocation groups defragmented
The number of allocation groups that were defragmented.

Allocation groups skipped - entirely free
The number of allocation groups that were skipped because they were entirely free.

Allocation groups skipped - too few free blocks
The number of allocation groups that were skipped because there were too few free blocks in them for reallocation.

Allocation groups skipped - contains a large contiguous free space
The number of allocation groups that were skipped because they contained a large contiguous free space which is not worth defragmenting.

Allocation groups are candidates for defragmenting
The number of allocation groups that are fit for defragmenting.

Average number of free runs in candidate allocation groups
The average number of free runs per allocation group, for allocation groups that are found fit for defragmentation. A free run is a contiguous set of blocks which are not allocated.

Total number of blocks
The total number of blocks in the file system.

Number of blocks that may be migrated
The number of blocks that may be moved during defragmentation.

File system is in percent fragmented
Shows to what extent the file system is fragmented in percentage.

Percentage of fragmentation in the file system: *percentage*
The percentage of fragmentation in the file system before and after running the **defragfs** command. The following example shows the percentage of fragmentation in the file system:

```
# defragfs -fv /exampleFS
File fragmentation before defrag: 100.00%
File fragmentation after defrag: 0.00%
```

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To defragment the **/data1** file system located on the **/dev/lv00** logical volume, enter:

   ```
   defragfs /dev/lv00
   ```

2. To defragment the **/data1** file system by specifying its mount point, enter:

   ```
   defragfs /data1
   ```

3. To generate a report on the **/data1** file system that indicates its current status as well as its status after being defragmented, enter:

   ```
   defragfs  -r /data1
   ```

4. To generate a report on the fragmentation in the **/data1** file system, enter:

   ```
   defragfs -s /data1
   ```

Files

Item	Description
/etc/filesystems	Lists the known file systems and defines their characteristics.

defvsd Command

Purpose

Designates a node as either having or using a virtual shared disk.

Syntax

defvsd *logical_volume_name global_group_name vsd_name*

Description

This command is run to specify logical volumes residing on globally accessible volume groups to be used as virtual shared disks.

You can use the System Management Interface Tool (SMIT) to run the **defvsd** command. To use SMIT, enter:

```
smit vsd_data
```

and select the **Define a Virtual Shared Disk** option.

Flags

-r
Resets the outgoing and expected sequence numbers for the nodes specified on the node on which the command is run. Use this flag when another node has either been rebooted, cast out, or all virtual shared disks have been reconfigured on that node. The specified nodes are also cast in.

Note: This option should be used only under direct guidance from IBM Service. It should never be used under normal circumstances.

-R
Resets the outgoing and expected sequence number for all nodes on the node on which the command is run. Use this flag after rebooting the node. All nodes in the virtual shared disk network will be cast in.

Note: This option should be used only under direct guidance from IBM Service. It should never be used under normal circumstances.

-p
Sets the level of virtual shared disk parallelism to the number specified. The valid range is 1 to 9. The default is 9. A larger value can potentially give better response time to large requests. (See *RSCT for AIX 5L: Managing Shared Disks* for more information regarding tuning virtual shared disk performance.)

This value is the *buf_cnt* parameter on the `uphysio` call that the virtual shared disk IP device driver makes in the kernel. Use `statvsd` to display the current value on the node on which the command is run.

-k
Casts out the node numbers specified on the local node. The local node ignores requests from cast out nodes. Use `-r` to cast nodes back in.

Note:

1. Before using this flag, refer to the “Restrictions” section that follows.
2. This option should be used only under direct guidance from IBM Service. It should never be used under normal circumstances.

-t
Lists the current routing table and mbuf headers cached by the virtual shared disk driver.

-T
Clears or releases all cached routes.

-v *vsd_name ...*
Resets the statistics in the number of read and write requests on the specified virtual shared disks.

-V
Resets all the configured virtual shared disk's statistics in the number of read and write requests.

-C
Resets the virtual shared disk device driver counters displayed by the `statvsd` command. Exceptions are the outgoing and expected request sequence numbers among the client and server nodes.

-K
Casts out all nodes on the local node. Local requests are still honored.

Note:

1. Before using this flag, refer to the “Restrictions” section that follows.
2. This option should be used only under direct guidance from IBM Service. It should never be used under normal circumstances.

-M
Sets the virtual shared disk maximum IP message size. This is the largest sized block of data the virtual shared disk sends over the network for an I/O request. This limit also affects local virtual shared disk I/O block size. The value is in bytes and must not be greater than the maximum transmission unit (MTU) size of the network. All nodes should use the same value. The recommended values are:

- 61440 (60KB) for a switch
- 8192 (8KB) for jumbo frame Ethernet
- 1024 (1KB) for 1500-byte MTU Ethernet

Parameters

logical_volume_name
Is the name of the logical volume you want to specify as a virtual shared disk. This logical volume must reside on the global volume group indicated. The length of the name must be less than or equal to 15 characters.

global_group_name
Is the name of the globally-accessible volume group previously defined by the **vsdvg** command where you want to specify a virtual shared disk. The length of the name must be less than or equal to 31 characters.

vsd_name
Specifies a unique name for the new virtual shared disk. This name must be unique within the RSCT peer domain, and, in order to avoid possible future naming conflicts, should also be unique across the overall cluster. The suggested naming convention is **vsdnn***gvg_name*. The length of the name must be less than or equal to 31 characters.

Note: If you specify a *vsd_name* that is already the name of another device, the **cfgvsd** command will be unsuccessful for that virtual shared disk. This error ensures that the special device files created for the name do not overlay and destroy files of the same name representing some other device type (such as a logical volume).

Security

You must have `root` authority to run this command.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to *RSCT Administration Guide* .

Examples

1. The following example specifies that, on the globally accessible volume group **vg1n1**, the logical volume known as **lv1vg1n1** is used as a virtual shared disk named **vsd1vg1n1**.

```
defvsd lv1vg1n1 vg1n1 vsd1vg1n1
```

Location

/opt/rsct/vsd/bin/defvsd

deleteX11input Command

Purpose

Deletes an X11 input extension record from the ODM (Object Data Manager) database.

Syntax

deleteX11input *DeviceName ...*

Description

The **deleteX11input** command is used to delete an X11 input extension record from the ODM database. For each *DeviceName* specified, the ODM database finds as many instances of the object as possible. This command queries the user to verify whether to delete each specific device found. A partial name may be specified.

The command is a root or system user command. Its action fails with a permissions error if an unauthorized user attempts to delete a record.

Parameter

Item	Description
DeviceName	Specifies the name of the X11 input extension device.

Error Codes

Item	Description
No DeviceName is found in ODM Database	No objects that match the specified pattern were found in the ODM database.
Usage: deleteX11input DeviceName	The user has not specified a device name.

delta Command

Purpose

Creates a delta in a SCCS file.

Syntax

delta [**-r** *SID*] [**-s**] [**-n**] [**-g** *List*] [**-p**] [**-m** *ModificationRequestList*] [**-y** [*Comment*]] *File ...*

Description

The **delta** command introduces into the named Source Code Control System (SCCS) file any changes that were made to the file version retrieved by a **get -e** command.

The **delta** command reads the g-files that correspond to the specified files (see the **get** command for a description of files created and used by SCCS) and creates a new delta. No line of a g-file can contain more than 512 characters.

If you specify a directory for the *File* value, the **delta** command performs the requested actions on all SCCS files within that directory that have been checked out previously for editing (that is, on all files with an **s.** prefix). If you specify a **-** (minus sign) in place of the *File* value, the **delta** command reads standard

input and interprets each line as the name of an SCCS file. When the **delta** command reads standard input, you must supply the **-y** flag. You must also supply the **-m** flag if the **v** header flag is set. The **delta** command reads standard input until it reaches an end-of-file character.

Note: Lines beginning with an SOH ASCII character (binary 001) cannot be placed in the SCCS file unless the SOH is quoted using a **** (backslash). SOH has special meaning to SCCS and causes an error.

Use of a **get** command on SCCS files, followed by the **delta** command on those same files, should be avoided when the **get** command generates a large amount of data. Instead, you should alternate the use of the **get** and **delta** commands.

The **delta** command saves the changes made to a particular version of an SCCS file. To use the **delta** command:

1. Use the **get -e** command to get an editable version of the file.
2. Edit that file.
3. Use the **delta** command to create a new version of the SCCS file.

The **delta** command prompts you for comments if the **-y** option is not specified. The comments apply to that particular delta and appear in the SCCS file header. The comments are not retrieved when you use the **get** command to get the delta and do not appear in the text of a retrieved file. Use comments to keep track of why a delta was created.

To see the comments, use an editor to look at the SCCS file, write the SCCS file to the display screen with the **cat** command, or print selected parts of the file to standard output using the **prs** command. Remember not to change the contents of the SCCS file directly. To change the delta comments, use the **cdc** command.

Note: Do not use the **delta** command on a file if it contains expanded identification keywords. Read-only file versions replace keywords with text values. Using the **delta** command on a read-only file causes the keywords to be lost. To recover from this situation, remove the delta or edit the file again and replace the identification keywords.

The SCCS does not allow use of the **delta** command unless an editable copy of the file exists.

To prevent the loss of keywords, use the **admin** command with the **-f** flag to specify the **i** header flag. Afterwards, the absence of keywords in a file version will cause an error.

Flags

Item	Description
-g *List*	Specifies a list of SIDs (deltas) to be ignored when the **get** command creates the g-file. After you use this flag, the **get** command ignores the specified delta when it builds the g-file.

Item	Description
-m *ModificationRequestList*	If the SCCS file has the **v** header flag set, then a Modification Request (MR) number must be supplied as the reason for creating the new delta. If you do not specify the **-m** flag, and the **v** header flag is set, the **delta** command reads MRs from standard input. If standard input is a workstation, the **delta** command prompts you for the MRs. The **delta** command continues to take input until it reads an end-of-file character. It always reads MRs before the comments (see the **-y** flag). You can use blanks, tab characters, or both to separate MRs in a list. If the **v** header flag has a value, it is interpreted as the name of a program that validates the MR numbers. If the **delta** command returns a nonzero exit value from the MR validation program, the **delta** command assumes some of the MR numbers were invalid and stops running.
-n	Retains the g-file, which is normally removed at completion of the **delta** command processing.
-p	Writes to standard output (in the format of the **diff** command) the SCCS file differences before and after the delta is applied. See the **diff** command for an explanation of the format.
-r *SID*	Specifies which delta is to be created in the SCCS file. You must use this flag only if two or more outstanding **get -e** commands were done on the same SCCS file by the same person. The *SID* value can be either the SID specified on the **get** command line or the SID to be created (as reported by the **get** command.) An error results if the specified SID cannot be uniquely identified, or if an SID must be specified but it is not.
-s	Suppresses the information normally written to standard output on normal completion of the **delta** command.
-y[*Comment*]	Specifies text that describes the reason for making a delta. A null string is considered a valid *Comment* value. If your comment line includes special characters or blanks, the line must be enclosed in single or double quotation marks. If you do not specify the **-y** flag, the **delta** command reads comments from standard input until it encounters a blank line or an end-of-file character. For keyboard input, the **delta** command prompts for the comments. If the last character of a line is a \ (backslash), it is ignored. Comments must be no longer than 512 characters.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.

Item	Description
>0	An error occurred.

Examples

1. To record changes you have made to an SCCS file, enter:

   ```
   delta s.prog.c
   ```

   This adds a delta to the SCCS file `s.prog.c`, recording the changes made by editing `prog.c`. The `delta` program then asks you for a comment that summarizes the changes you made. Enter the comment, and then enter an end-of-file character or press the return key twice to indicate that you have finished the comment.
2. To record the changes you have made to an SCCS file with a brief descriptive comment, enter:

   ```
   delta -y "This delta contains the payroll function" s.prog.c
   ```

Files

Item	Description
/usr/bin/delta	Contains the **delta** command.

deroff Command

Purpose

Removes **nroff**, **troff**, **tbl**, and **eqn** command constructs from files.

Syntax

deroff { **-ma** **-me** **-ms** [**-mm** [**-ml**]] } [**-i** | **-l**] [**-k**] [**-p**] [**-u**] [**-w**] [*File* ...]

Description

The **deroff** command reads the specified files (standard input by default) containing English-language text, removes all **troff** requests, macro calls, backslash constructs, **eqn** command constructs (between **.EQ** and **.EN** lines and between delimiters), and **tbl** command descriptions, then writes the remainder of the file to standard output.

The **deroff** command normally follows chains of included files (**.so** and **.nx troff** command requests). If a file has already been included, a **.so** request naming it is ignored and an **.nx** request naming that file ends execution.

Note: The **deroff** command is not a complete **troff** command interpreter, so it can be confused by subtle constructs. Most errors result in too much rather than too little output.

Parameters

Item	Description
File	Specifies English-language text files for the **deroff** command to remove the effects of **troff**, **eqn**, and **tbl** command processing. The default file is standard input.

Flags

Item	Description
-ma	Ignores **MA** (**man**) macros in text so that only running text is output.
-me	Ignores **ME** macros in text so that only running text is output. This is the default.
-ml	Ignores **MM** macros in text (**-mm** flag) and also deletes **MM** list structures. The **-mm** flag must be specified with this flag. **Note:** Do not use the **-ml** flag with nested lists.
-mm	Ignores **MM** macros.
-ms	Ignores **MS** macros in text so that only running text is output.
-i	Suppresses the processing of included files.
-l	Suppresses the processing of included files whose names begin with **/usr/lib**, such as macro files in **/usr/lib/tmac**.
-k	Retains blocks specified to be kept together. The default is to remove kept blocks of text; for example, the **.ne** construct is removed.
-p	Processes special paragraphs.
-u	Removes the ASCII underline and boldface control sequences. This flag automatically sets the **-w** flag.
-w	Makes the output a word list, with one word per line and all other characters deleted. Otherwise, the output follows the original. In text, a word is any string that begins with a letter, contains at least two letters, and is composed of letters, digits, ampersands (&), and apostrophes ('). In a macro call, however, a word is a string that begins with at least two letters and contains a total of at least three letters. Delimiters are any characters other than letters, digits, punctuation, apostrophes, and ampersands. Trailing apostrophes and ampersands are removed from words.

detachrset Command

Purpose

Detaches an rset from a process.

Syntax

detachrset [**-P**] *pid*

Description

The **detachrset** command detaches an rset from a process. Detaching an rset from a process will allow the process to use any of the processors and/or memory regions in the system.

Flags

Item	Description
-P	Detaches the partition rset from the specified process (*pid*).

Parameters

Item	Description
pid	Process ID.

Security

The user must have `root` authority or have CAP_NUMA_ATTACH capability and the target process must have the same effective **userid** as the command issuer. The user must have `root` authority to remove the partition rset from a process (the **-P** option).

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

To detach the rset from process 21414, type:

```
detachrset 21414
```

Files

Item	Description
/usr/bin/detachrset	Contains the **detachrset** command.

devinstall Command

Purpose

Installs software support for devices.

Syntax

devinstall **-f** *File* **-d** *Device* [**-s**] [**-v**]

Description

The **devinstall** command installs software support for devices. It installs the software packages listed in the file specified by the `-f` flag.

For most new devices that are added after the initial software installation, the software for the new device can be installed using the **-i** flag of the **cfgmgr** command.

In some instances, the new device replaces a device that is needed to start the machine. For example, you might be replacing the SCSI adapter card that supports the root volume group or the graphics adapter card that supports the console. In this case, the machine will not start in normal mode until you have installed software support for this new device. To do this, turn your system off and install the new hardware according to the directions included with your hardware. Next, start up your machine in maintenance mode. During the startup process, the new adapter is detected and the **/tmp/device.pkgs**

file is created containing the name of the software package needed to support the new hardware. Once the machine is in maintenance mode, you can install the software for this new device by running the **devinstall** command.

Flags

Item	Description
-f *File*	Specifies the file containing the list of packages to be installed. Typically, this will be the **/tmp/device.pkgs** file generated by the **cfgmgr** command.
-d *Device*	Specifies where the installation medium can be found. This can be a hardware device, such as tape or diskette; it can be a directory that contains installation images; or it can be the installation image file itself. When the installation media is an IBM Installation tape or IBM Corrective Service tape, the tape device should be specified as no-rewind-on-close and no-retension-on-open. Examples of this would be **/dev/rmt0.1** for a high-density tape or **/dev/rmt0.5** for a low-density tape. For non-IBM-supplied tapes, use the options specified by the tape supplier.
-s	Overwrites the **/var/adm/dev_pkg.fail** file. This file contains a list of all packages that did not install successfully and can be used to facilitate recovery or installation from a different source.
-v	Specifies the verbose option, causing the **devinstall** command to display additional information while processing.

The **devinstall** command installs the device packages listed in the file specified on the command line. It runs the **geninstall** command with the **-I** "acXge **/var/adm/ras/devinst.log**", where a: apply, c: commit, X: extend fs, e: log and **/var/adm/ras/devinst.log** is the log file full path name, g: auto_include. (See the **geninstall** command for more information on these flags.) The **devinstall** command checks the summary file generated by the **geninstall** command for the results of each package install attempt and, based on this information, creates two files. The **/var/adm/dev_pkg.fail** file lists the packages that fail to install (if any). The `/var/adm/dev_pkg.success` file lists all packages that are installed successfully.

Return Values

A return value of 0 indicates that no packages were installed.

A return value of 1 indicates that at least one package was successfully installed, and the **bosboot** command should be executed.

A return value of 2 indicates that the `devinstall` command failed.

The `/var/adm/dev_pkg.success` file lists those packages that successfully installed. The **/var/adm/dev_pkg.fail** file lists those packages that failed installation.

Security

Privilege Control: Only the root user can run this command.

Examples

To install software to support a new device after you have started the machine from the device installation tape and entered maintenance mode, enter:

```
devinstall -f /../tmp/device.pkgs -d /dev/rmt0.1
```

Then, run the **bosboot** command.

```
bosboot -ad /dev/ipldevice
```

File

Item	Description
/dev/rmtn	Specifies the raw streaming tape interface.

devnm Command

Purpose

Names a device.

Syntax

devnm *Path* ...

Description

The **devnm** command reads the *Path* parameter, identifies the special file associated with the mounted file system where the *Path* parameter resides, and writes the special file name to standard output. Each *Path* parameter must be a full path name.

The most common use of the **devnm** command is by the **/etc/rc** command file to construct a mount table entry for the root device.

Note: This command is for local file systems only.

Examples

1. To identify the device on which a file resides, enter:

   ```
   devnm /diskette0/bob/textfile
   ```

   This displays the name of the special device file on which the `/diskette0/bob/textfile` file resides. If a diskette is mounted as the `/diskette0` device, the **devnm** command displays:

   ```
   fd0 /diskette0/bob/textfile
   rfd0 /diskette0/bob/textfile
   ```

   This means the `/diskette0/bob/textfile` file resides on the **/dev/fd0** diskette drive.
2. To identify the device on which a file system resides, enter:

   ```
   devnm /
   ```

   This displays the name of the device on which the root file system(/) resides. The following list is displayed on the screen:

   ```
   hd0 /
   ```

   This means that the root file system (/) resides on the **/dev/hd0** device.

Files

Item	Description
/dev	Specifies the directory.
/usr/sbin/devnm	Contains the **devnm** command.

devrsrv Command

Purpose

Queries and breaks the single-path and persistent reservations on a device.

Syntax

devrsrv **-c** *query* [**-e**] | *release* | **prin** **-s** *sa* | (**prout** **-s** *sa* **-r** *rkey* **-k** *sa_key* **-t** *prtype*) **-l** *devicename*

devrsrv **-f** **-l** *devicename*

devrsrv **-d**

Description

The **devrsrv** command queries and breaks the single-path and persistent reservations on the device. The command runs the persistent reserve in (**prin**) and persistent reserve out (**prout**) service actions.

The **query** subcommand queries and displays the current reservation status of the device. The **release** subcommand releases the reservation on the device by using the single-path reservation.

The **prin** subcommand displays all the registered reservation keys, reservation key holder, and capabilities information. The **prout** subcommand requests service action that reserves a device for the exclusive or shared use of a particular I/O path to the device. The **prout** subcommand supports the following service actions:

Item	Description
RELEASE	Releases the specified persistent reservation for the device.
CLEAR	Clears all the reservation keys and all the persistent reservations.
PREEMPT	Preempts the persistent reservations or removes registrations, or both.
PREEMPT AND ABORT	Preempts persistent reservations or removes registrations, or both and stops all tasks for all preempted I/O paths to the device.
REGISTER AND IGNORE KEY	Registers the new key value in place of the old key value.

Flags

Item	Description
-c	Specifies the following subcommands: **query** Queries and displays the status of reservations on a device. **release** Releases the device with the single-path reservation by using SCSI-2. **prin** Specifies the persistent reservation in service action. **prout** Specifies the persistent reservation out service action.
-d	Lists the disk name and other identifying information for all disks that are queried or manipulated by using the **devrsrv** command.

Item	Description
-e	Avoids opening the disk in exclusive mode, which includes both single and diagnostic modes. This flag is applicable only for PR_exclusive and PR_shared reservation types. If Object Data Manager (ODM) reservation policy is single_path, this flag is ignored. **Note:** In some cases, if you use this flag, the **devrsrv** command might not determine the reservation status of the disk or whether the disk is already open on the local host.
-f	Breaks the reservation that is held by other I/O path or host. For single-path reservations, the **devrsrv** command issues a SC_FORCED_OPEN action to break the reservation. For persistent reservations, the **devrsrv** command issues a **prout** subcommand along with the CLEAR service action to clear the persistent reservation and the registrations.
-k	Specifies the service action reservation key. The **-k** flag is required for the REGISTER, PREEMPT, and PREEMPT_ABORT service actions.
-l	Specifies the name of the device.
-r	Specifies the reservation key. The **-r** flag is required for the REGISTER, PREEMPT, PREEMPT AND ABORT, and RELEASE service actions.
-s	Specifies the service action for persistent reservations. The valid service actions for the **prin** subcommand follow: **0** READ KEYS **1** READ RESERVATION **2** REPORT CAPABILITIES The valid service actions for the **prout** subcommand follow: **2** RELEASE **3** CLEAR **4** PREEMPT **5** PREEMPT AND ABORT **6** REGISTER AND IGNORE EXISTING KEY

Item	Description
-t	Specifies the persistent reservation type. The types of persistent reservations follow: **1** Write exclusive **2** Exclusive access **3** Write exclusive registrants only **4** Exclusive access registrants only **5** Write exclusive all registrants **6** Exclusive access all registrants

Examples

The following are the examples that are related to different scenarios.

`Query operation`

1. To query the reservation status of the `hdisk0` device when it is not reserved by any host, enter the following command:

```
# devrsrv -c query -l hdisk0
```

```
Device Reservation State Information
================================================
Device Name                     :  hdisk0
Device Open On Current Host?    :  NO
ODM Reservation Policy          :  SINGLE PATH RESERVE
Device Reservation State        :  NO RESERVE
```

   The output shows that the device is not opened on the current host and the Object Data Manager (ODM) reservation policy is SINGLE PATH RESERVE. It indicates that the reservation policy is set in the ODM for this device. The device reservation state indicates the reservation that is present on the device. You can find the device reservation state by running a sequence of SCSI commands.

2. To query the reservation status of the `hdisk1` device when it is reserved by a host, enter the following command:

```
# devrsrv -c query -l hdisk1
```

   The device is reserved by using the single path reservation by a host.

```
Device Reservation State Information
================================================
Device Name                    :  hdisk1
Device Open On Current Host?   :  NO
ODM Reservation Policy         :  SINGLE PATH RESERVE
Device Reservation State       :  SINGLE PATH RESERVE
```

3. To query the reservation status of the `hdisk2` device when it is reserved on the same host, enter the following command:

```
# devrsrv -c query -l hdisk2
```

```
Device Reservation State Information
================================================
Device Name                    :  hdisk2
```

```
Device Open On Current Host?    :  YES
ODM Reservation Policy          :  SINGLE PATH RESERVE
Device Reservation State        :  SINGLE PATH RESERVE
Path Id of Reserved Path        :  0
```

4. To query the reservation status of the hdisk2 device when the ODM reservation policy is PR SHARED and the device is not reserved by any host, enter the following command:

```
# devrsrv -c query -l hdisk0
```

```
Device Reservation State Information
==================================================
Device Name              :  hdisk0
Device Open              :  NO
ODM Reservation Policy   :  PR SHARED
ODM PR Key Value         :  7777
Device Reservation State :  NO RESERVE
Registered PR Keys       :
555
777
PR Capabilities Byte[2]  :  0xd  SIP_C   ATP_C   PTPL_C
PR Capabilities Byte[3]  :  0x0
PR Types Supported       :  NOT VALID
```

Descriptions of several fields from the query output follow:

Registered PR Keys:
Displays keys that are registered by running the **prout** subcommand along with the REGISTER service action from all I/O paths that are sharing this device.

PR Capabilities Bytes:
Indicates the content of bytes 2 and 3 returned by the REPORT CAPABILITIES service action of the **prin** subcommand. See the SPC standard to interpret the output of the Example 4.

PR Types Supported:
Displays the persistent reservation types that are supported by the device that are reported by the persistent reservation type mask field in the report capabilities output.

If the persistent reservation is held on a device, the query output displays additional information about the device reservation as follows:

PR Reservation Type:
Displays one of the values of the PR Types that are described in the Flags section.

PR Holder key Value:
Displays the PR key value of the current reservation holder. The persistent reservation key value is 0 if the PR Type is 5 or 6.

Persistent reserve in (prin) operation

1. To read all of the registered reservation keys, enter the following command:

```
# devrsrv -c prin -s 0 -l hdisk0
```

```
Registered PR Keys             :
555
777
```

2. To read the current reservation key holder and type, enter the following command:

```
# devrsrv -c prin -s 1 -l hdisk0
```

```
PR Generation Value     :  2
PR Type                 :  PR_EA_RO (EXCLUSIVE ACCESS, REGISTRANTS ONLY)
PR Holder Key Value     :  777
```

3. To return the PR capabilities information that is supported by sending the report capabilities service action, enter the following command:

```
# devrsrv -c prin -s 2 -l hdisk0
```

```
PR Capabilities Byte[2]         :  0xd  SIP_C   ATP_C   PTPL_C
PR Capabilities Byte[3]         :  0x0
PR Types Supported              :  NOT VALID
```

Persistent reserve out (prout) operation

RELEASE service action

To release the persistent reservation from IT-nexus that is registered and reserved with key 1777 and PR reservation type 4, enter the following command:

```
# devrsrv -c prout -s 2 -r 1777 -t 4 -l hdisk0
```

```
Device Reservation State Information
================================================
Device Name                     :  hdisk0
Device Open On Current Host?    :  YES
ODM Reservation Policy          :  PR SHARED
ODM PR Key Value                :  7777
Device Reservation State        :  PR SHARED
Reservation will be cleared on the device. Do you want to continue y/n:y
```

If you run the query now, the result displays the `Device Reservation State` as NO RESERVE.

```
# devrsrv -c query -l hdisk0
```

```
Device Reservation State Information
================================================
Device Name              :  hdisk0
Device Open              :  NO
ODM Reservation Policy   :  PR SHARED
ODM PR Key Value         :  7777
Device Reservation State :  NO RESERVE
Registered PR Keys       :
555
1777
PR Capabilities Byte[2]  :  0xd  SIP_C   ATP_C   PTPL_C
PR Capabilities Byte[3]  :  0x0
PR Types Supported       :  NOT VALID
```

CLEAR service action

To release the persistent reservation and to remove all the registrations from a device server that uses the CLEAR service action by using a registered I/O path with key 555, enter the following command:

```
# devrsrv -c prout -s 3 -r 555 -l hdisk0
```

```
Device Reservation State Information
================================================
Device Name                     :  hdisk0
Device Open On Current Host?    :  YES
ODM Reservation Policy          :  PR SHARED
ODM PR Key Value                :  5555
Device Reservation State        :  PR SHARED
Reservation will be cleared on the device. Do you want to continue y/n:y
```

If you run the query now, the persistent reservation is released and the registrations are removed from the device.

```
# devrsrv -c query -l hdisk0
```

```
Device Reservation State Information
================================================
Device Name              :  hdisk0
Device Open              :  NO
ODM Reservation Policy   :  PR SHARED
ODM PR Key Value         :  5555
Device Reservation State  :  NO RESERVE
```

```
Registered PR Keys        :  No Keys Registered
PR Capabilities Byte[2]   :  0xd  SIP_C   ATP_C   PTPL_C
PR Capabilities Byte[3]   :  0x0
PR Types Supported        :  NOT VALID
```

PREEMPT and PREEMPT_ABORT service actions

To preempt the persistent reservation that is held with reservation holder 444 by another IT-nexus with the registered key 777, enter the following command:

```
# devrsrv -c prout -s 4 -r 777 -k 444 -t 2 -l hdisk0
```

Before you run the `# devrsrv -c prout -s 4 -r 777 -k 444 -t 2 -l hdisk0` command, the query output is displayed as follows.

```
# devrsrv -c query -l hdisk0
```

```
Device Reservation State Information
==================================================
Device Name               :  hdisk0
Device Open               :  NO
ODM Reservation Policy    :  PR SHARED
ODM PR Key Value          :  7777
Device Reservation State  :  PR EXCLUSIVE
PR Generation Value       :  5
PR Type                   :  PR_WE (WRITE EXCLUSIVE)
PR Holder Key Value       :  444
Registered PR Keys        :
777
444
PR Capabilities Byte[2]   :  0xd  SIP_C   ATP_C   PTPL_C
PR Capabilities Byte[3]   :  0x0
PR Types Supported        :  NOT VALID
```

After you run the `# devrsrv -c prout -s 4 -r 777 -k 444 -t 2 -l hdisk0` command, the query output shows that the reservation is preempted by IT-nexus with key 777 and key 444 is unregistered.

```
# devrsrv -c query -l hdisk0
```

```
Device Reservation State Information
==================================================
Device Name              :  hdisk0
Device Open              :  NO
ODM Reservation Policy   :  PR SHARED
ODM PR Key Value         :  7777
Device Reservation State :  PR EXCLUSIVE
PR Generation Value      :  6
PR Type                  :  PR_EA (EXCLUSIVE ACCESS)
PR Holder Key Value      :  777
Registered PR Keys       :
777
PR Capabilities Byte[2]  :  0xd  SIP_C   ATP_C   PTPL_C
PR Capabilities Byte[3]  :  0x0
PR Types Supported       :  NOT VALID
```

RELEASE operation for SINGLE PATH RESERVE policy

To release the reservation on the `hdisk0` device, enter the following commands:

- Scenario 1: The current host is the owner of the reservation.

```
# devrsrv -c query -l hdisk0

Device Reservation State Information
================================================
Device Name                    :  hdisk0
Device Open On Current Host?   :  YES
ODM Reservation Policy         :  SINGLE PATH RESERVE
Device Reservation State       :  SINGLE PATH RESERVE
Path Id of Reserved Path       :  0
```

```
# devrsrv -c release -l hdisk0

Device Reservation State Information
================================================
Device Name                     :  hdisk0
Device Open On Current Host?    :  YES
ODM Reservation Policy          :  SINGLE PATH RESERVE
Device Reservation State        :  SINGLE PATH RESERVE
Device is currently Open on this host by a process.Do you want to continue y/n:y
Command Successful
Reservation cleared on the device. Query operation may not work properly.
Close the application that holds the reservation and retry.
```

- Scenario 2: The current host is not the owner of the reservation.

```
# devrsrv -c query -l hdisk0

Device Reservation State Information
================================================
Device Name                     :  hdisk0
Device Open On Current Host?    :  NO
ODM Reservation Policy          :  SINGLE PATH RESERVE
Device Reservation State        :  SINGLE PATH RESERVE

Because the current host does not own the reservation on the device,
try the force option if you want to break the reservation.
```

```
# devrsrv -f -l hdisk0
```

The device is already reserved by using the single-path reservation by another host.

```
Device Reservation State Information
================================================
Device Name                     :  hdisk0
Device Open On Current Host?    :  NO
ODM Reservation Policy          :  SINGLE PATH RESERVE
Device Reservation State        :  SINGLE PATH RESERVE
Reservation will be cleared on the device. Do you want to continue y/n:y
```

After you run the release command successfully, the query option must display NO RESERVE as the device reservation state.

```
# devrsrv -c query -l hdisk0

Device Reservation State Information
================================================
Device Name                     :  hdisk0
Device Open On Current Host?    :  NO
ODM Reservation Policy          :  SINGLE PATH RESERVE
Device Reservation State        :  NO RESERVE
```

Forced mode

The hdisk0 device is reserved with key 777 from another I/O path. To release this reservation from the other client, enter the following command:

```
# devrsrv -f -l hdisk0

Device Reservation State Information
================================================
Device Name                     :  hdisk16
Device Open On Current Host?    :  NO
ODM Reservation Policy          :  PR SHARED
ODM PR Key Value                :  5555
Device Reservation State        :  PR SHARED
Reservation will be cleared on the device. Do you want to continue y/n:y
Command Successful
```

Before you run the # devrsrv -f -l hdisk0 command, the query displays the following output:

```
# devrsrv -c query -l hdisk0

Device Reservation State Information
================================================
```

```
Device Name                :  hdisk0
Device Open                :  NO
ODM Reservation Policy     :  PR SHARED
ODM PR Key Value           :  5555
Device Reservation State   :  PR EXCLUSIVE
PR Generation Value        :  1
PR Type                    :  PR_WE (WRITE EXCLUSIVE)
PR Holder Key Value        :  777
Registered PR Keys         :
777
PR Capabilities Byte[2]    :  0xd  SIP_C   ATP_C   PTPL_C
PR Capabilities Byte[3]    :  0x0
PR Types Supported         :  NOT VALID
```

After you execute the `# devrsrv -f -l hdisk0` command, the output indicates that the device is not reserved.

```
# devrsrv -c query -l hdisk0

Device Reservation State Information
================================================
Device Name                    :  hdisk16
Device Open On Current Host?   :  NO
ODM Reservation Policy         :  PR SHARED
ODM PR Key Value               :  5555
Device Reservation State       :  NO RESERVE
Registered PR Keys             :  No Keys Registered
PR Capabilities Byte[2]        :  0x0
PR Capabilities Byte[3]        :  0x0
PR Types Supported             :  NOT VALID
```

df Command

Purpose

Reports information about space on file systems. This document describes the AIX **df** command as well as the System V version of **df**.

Syntax

df [[**-P**] | [**-I** | **-M** | **-i** | **-t** | **-v**]] [**-c**] [**-T** { `local` | `remote` | `vfstype`}] [**-F** {*output1 output2 output3 ...*}] [**-k**] [**-m**] [**-g**] [**-s**] [*FileSystem ...* | *File...*]

Description

The **df** command displays information about total space and available space on a file system. The *FileSystem* parameter specifies the name of the device on which the file system resides, the directory on which the file system is mounted, or the relative path name of a file system. The *File* parameter specifies a file or a directory that is not a mount point. If the *File* parameter is specified, the **df** command displays information for the file system on which the file or directory resides. If you do not specify the *FileSystem* or *File* parameter, the **df** command displays information for all currently mounted file systems. File system statistics are displayed in units of 512-byte blocks by default.

The **df** command gets file system space statistics from the **statfs** system call. However, specifying the **-s** flag gets the statistics from the virtual file system (VFS) specific file system helper. If you do not specify arguments with the **-s** flag and the helper fails to get the statistics, the **statfs** system call statistics are used. Under certain exceptional conditions, such as when a file system is being modified while the **df** command is running, the statistics displayed by the **df** command might not be accurate.

Note: Some remote file systems, such as the Network File System (NFS), do not provide all the information that the **df** command needs. The **df** command prints blanks for statistics that the server does not provide.

The **df** command does not fully support NFSv4 filesystems. Use the **nfs4cl** command to extract block and space information.

Flags

Item	Description
-c	Displays the output in colon separated format.
-F { *output1 output2 output3* ... }	Displays only those values that are specified by the headings in the output parameters. By default, the file system and blocks-allocated headings are always turned on. The following values are acceptable for headings: **%m** Mounted on **%u** Used **%z** Percentage used **%f** Free **%l** Inodes used **%n** Inodes free **%p** Percentage of inodes used
-g	Displays statistics in units of GB blocks. The output values for the file system statistics would be in floating point numbers as value of each unit in bytes is significantly high.
-i	Displays the number of used inodes and the percentage of inodes in use for the file system. This output is the default when the specified file system is mounted.
-I	Displays information on the total number of blocks, the used space, the free space, the percentage of used space, and the mount point for the file system.
-k	Displays statistics in units of 1024-byte blocks.
-m	Displays statistics in units of MB blocks. The output values for the file system statistics would be in floating point numbers as value of each unit in bytes is significantly high.
-M	Displays the mount point information for the file system in the second column.
-P	Displays information on the file system in POSIX portable format. When the **-P** flag is specified, the header line appears similar to: `Filesystem 512-blocks Used Available Capacity Mounted on\n` If the **-k**, **-m** or **-g** flag is specified in addition to the **-P** flag, the column heading 512-blocks is replaced by the respective units, depending on which of these flags is used with the **-P** flag. File system statistics are displayed on one line in the following order: *FileSystem, TotalSpace, UsedSpace, FreeSpace, UsedPercentage, MountPoint*
-s	Displays statistics on unmounted JFS or Enhanced JFS file systems by the command line arguments. If there are no arguments specified, the `-s` flag has no effect. If the file systems specified by the argument are currently mounted or an argument is a file, the `-s` flag has no effect for that particular argument. To collect statistics on unmounted file systems, an argument must be a JFS or Enhanced JFS file system mount point or device, the file system must be listed in **/etc/filesystems**, and the user must have read access to the device.
-t	Includes figures for total allocated space in the output.

Item	Description
-T {local|remote|vfstype}	Filters the output by the type of file system. This flag can have one of the following parameters: **local** Displays only the Journaled File System (JFS) and Enhanced Journaled File System (JFS2) file systems. **remote** Displays all non-local file systems. **vfstype** Displays file systems only of a specific virtual file system (VFS), for example, JFS, JFS2, Network File System version 4 (NFSv4), and so on.
-v	Displays all information for the specified file system.

The values of the output parameters with the flags **-m** and **-g** would be rounded off to nearest second decimal digit. If all or any two of the **-k**, **-m** and **-g** flags are specified, the last one specified takes effect.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To display information about all mounted file systems, enter:

   ```
   df
   ```

   If your system has the **/**, **/usr**, **/site**, and **/usr/venus** file systems mounted, the output from the **df** command resembles the following:

   ```
   Filesystem 512-blocks Free   %Used   Iused  %Iused  Mounted on
   /dev/hd0    19368     9976    48%     4714    5%     /
   /dev/hd1    24212     4808    80%     5031   19%     /usr
   /dev/hd2     9744     9352     4%     1900    4%     /site
   /dev/hd3     3868     3856     0%      986    0%     /usr/venus
   ```

2. To display information about /test file system in 1024-byte blocks, enter:

   ```
   df -k /test
   ```

   ```
   Filesystem    1024 blocks    Free    %Used   Iused  %Iused  Mounted on
   /dev/lv11         16384     15824       4%      18      1%  /tmp/ravi1
   ```

   This displays the file system statistics in 1024-byte disk blocks.

3. To display information about /test file system in MB blocks, enter:

   ```
   df -m /test
   ```

   ```
   Filesystem    MB blocks    Free    %Used    Iused  %Iused  Mounted on
   /dev/lv11       16.00     15.46       4%       18      1%  /tmp/ravi1
   ```

   This displays file system statistics in MB disk blocks rounded off to nearest 2nd decimal digit.

4. To display information about the **/test** file system in GB blocks, enter:

```
df -g /test
```

```
Filesystem    GB blocks   Free     %Used    Iused  %Iused  Mounted on
/dev/lv11          0.02   0.02        0%       18      1%  /tmp/ravi1
```

   This displays file system statistics in GB disk blocks rounded off to nearest 2nd decimal digit.

5. To display available space on the file system in which your current directory resides, enter:

```
cd/
df .
```

   The output from this command resembles the following:

```
Device   512-blocks  free   %used   iused   %iused  Mounted on
/dev/hd4    19368    9976    48%     4714    5%     /
```

6. To display the output in a colon separated format, enter:

```
df -c
```

   The output resembles the following example:

```
Filesystem:512-blocks:Free:%Used:Iused:%Iused:Mounted on
/dev/hd4:491520:113168:77%:9930:42%:/
/dev/hd2:5046272:27696:100%:43014:86%:/usr
```

7. To display information about all the file systems that are mounted locally, enter:

```
df -T local
```

   The output resembles the following example:

```
Filesystem    512-blocks      Free  %Used    Iused %Iused Mounted on
/dev/hd4          5898240   2104184   65%    16390     7%  /
/dev/hd2          7602176   1698696   78%    56001    23%  /usr
/dev/hd9var       3014656   2190976   28%    10987     5%  /var
/dev/hd3          2883584   2137928   26%     1213     1%  /tmp
/dev/hd1          655360    645240    2%     1727     3%   /home
/dev/hd11admin    262144    261384    1%        5     1%   /admin
/proc                  -         -    -         -     -    /proc
/dev/hd10opt      786432    362672   54%     8926    18%   /opt
/dev/livedump     524288    523552    1%        4     1%   /var/adm/ras/livedump
/aha                   -         -    -       328     2%   /aha
```

8. To display information about all the JFS2 file systems, enter:

```
df -T jfs2
```

   The output resembles the following example:

```
Filesystem     512-blocks      Free %Used    Iused %Iused Mounted on
/dev/hd4          5898240   2104184   65%    16390     7%  /
/dev/hd2          7602176   1698696   78%    56001    23%  /usr
/dev/hd9var       3014656   2190976   28%    10987     5%  /var
/dev/hd3          2883584   2137928   26%     1213     1%  /tmp
/dev/hd1          655360    645240    2%     1727     3%   /home
/dev/hd11admin    262144    261384    1%        5     1%   /admin
/dev/hd10opt      786432    362672   54%     8926    18%   /opt
/dev/livedump     524288    523552    1%        4     1%   /var/adm/ras/livedump
```

9. To display the `free`, `used`, and `mounted` on information about all the JFS2 file systems, enter:

```
df -T jfs2 -F %f %u %m
```

   The output resembles the following example:

```
Filesystem    512-blocks      Free %Used    Mounted on
/dev/hd4         5898240   2104184   65%    /
```

```
/dev/hd2         7602176   1698696   78%    /usr
/dev/hd9var      3014656   2190976   28%    /var
/dev/hd3         2883584   2137928   26%    /tmp
/dev/hd1          655360    645240    2%    /home
/dev/hd11admin     262144    261384         /admin
/dev/hd10opt       786432    362672         /opt
/dev/livedump      524288    523552        /var/adm/ras/livedump
```

Files

Item	Description
/etc/filesystems	Lists the known file systems and defines their characteristics.
/etc/vfs	Contains descriptions of virtual file system types.

System V df Command

Purpose

Reports number of free disk blocks and files.

Syntax

/usr/sysv/bin/df [**-a**] [**-l**] [[[**-e**] [**-g**] [**-n**]] | [[**-i**] [**-v**]] | **-t**]] [*FileSystem* ...] [*File* ...]

Description

The **df** command displays information about total space and available space on a file system. File system statistics are displayed in units of 512-byte blocks

Flags

Item	Description
-a	Performs the default operation and prints the mount point, the device name, number of free blocks and number of used inodes (files).
-e	Print only the number of free files.
-g	Print the entire **statvfs** structure. This option overrides the **-a** , **-e**, **-i**, **-n**, **-t** and **-v** options. The numbers for available, total, and free blocks are reported in 512 byte blocks.
-i	Displays the total number of inodes, the number of free inodes, the number of used inodes, and the percentage of inodes in use.
-l	Reports on local file systems only.
-n	Prints the type of filesystem.
-t	Causes total allocated block figures to be reported.
-v	Reports percent of blocks used as well as the number of blocks used and free.

Parameters

Item	Description
File	The *File* parameter specifies a file or a directory that is not a mount point. If the *File* parameter is specified, the **df** command displays information for the file system on which the file or directory resides.
FileSystem	The *FileSystem* parameter specifies the name of the device on which the file system resides, the directory on which the file system is mounted, or the relative path name of a file system.

Note: If the *FileSystem* or *File* parameter is not specified, the **df** command displays information for all currently mounted file systems.

Exit Status

0
The command completed successfully

>0
An error occurred.

Examples

1. To display information about all mounted file systems, enter:

```
/usr/sysv/bin/df
```

   The output looks similar to the following:

```
/              (/dev/hd4       ):     19656 blocks    1504 files
/usr           (/dev/hd2       ):   1139904 blocks   20254 files
/var           (/dev/hd9var    ):     23096 blocks     512 files
/tmp           (/dev/hd3       ):      2464 blocks     204 files
/home          (/dev/hd1       ):     44208 blocks     146 files
/proc          (/proc          ):         0 blocks       0 files
/opt           (/dev/hd10opt   ):     13880 blocks     310 files
```

2. To display information about the file system in which your current directory resides, enter:

```
/usr/sysv/bin/df .
```

3. To display the total number of inode, the number of free inodes and the number of available inodes in all mounted file systems, enter:

```
/usr/sysv/bin/df -i
```

   The output looks similar to the following:

```
Mount Dir  Filesystem      iused    avail    itotal  %iused
/          /dev/hd4         1504     6688      8192     19%
/usr       /dev/hd2        20254   127202    147456     14%
/var       /dev/hd9var       512     3584      4096     13%
/tmp       /dev/hd3          204     5940      6144      4%
/home      /dev/hd1          146    14190     14336      2%
/proc      /proc               0        0         0       0
/opt       /dev/hd10opt      310     5834      6144      6%
```

4. To display the total number of blocks , the number of used blocks and the number of free blocks on a the **/tmp** file system, enter:

```
/usr/sysv/bin/df -v /tmp
```

5. To display the type of filesystem, enter:

```
/usr/sysv/bin/df -n
```

6. To display inode information on all local filesystems, enter:

```
/usr/sysv/bin/df -i -l
```

7. To display the statvfs structure information on all the filesystems, enter:

```
/usr/sysv/bin/df -g
```

8. To display the number of free files on filesystems, enter:

```
/usr/sysv/bin/df -e
```

Files

Item	Description
/usr/sysv/bin/df	Contains the System V **df** command.
/etc/filesystems	Contains filesystem information.

dfmounts Command

Purpose

Displays mounted resource information.

Syntax

dfmounts [**-F** *fstype*] [**-h**] [*server* ...]

Description

The **dfmounts** command prints local systems that are remotely mounted by clients through Network File System (NFS). It also prints the list of clients that have mounted the resource. The **dfmounts** command prints a header that is followed by a list of resource information separated with whitespace characters within fields.

For each resource, the following fields are displayed:

RESOURCE
For NFS, a hyphen "-" is marked.

SERVER
Indicates the machine from which the resource was mounted.

PATHNAME
Indicates the path of the shared resource.

CLIENTS
A comma separated list of systems that currently have the resource mounted.

Flags

Item	Description
-F *fstype*	Specifies the File System Type (fstype). Only **nfs** type of file system is supported.
-h	Suppress the header line in the output of **dfmounts**.

Parameters

Item	Description
Server	Represents a system on the network that had made its resources available to the local system. *Server* prints the resources that is made available from the machine together with the current clients using each resource. If this parameter is not specified, then the **dfmounts** command prints information by assuming that server is the local system. Multiple server names can be provided with the **dfmounts** command.

Exit Status

0
The command completed successfully

>0
An error occurred.

Security

Examples

1. To print the mounted resource information on the system "mercury" for file system type "**nfs**", enter:

```
dfmounts -F nfs mercury
```

2. To print mounted resource information without header on the system for file system type "**nfs**", enter:

```
dfmounts -hF nfs
```

Files

Item	Description
/usr/bin/dfmounts	Contains the generic System V **dfmounts** command.
/usr/lib/fs/nfs/dfmounts	Contains the System V **dfmounts** command for nfs.
/etc/vfs	Contains the description for known virtual file system implementations.

dfpd Command

Purpose

Provides load statistics about servers being load balanced to the Load Manager.

Syntax

/usr/sbin/dfpd [**-d**] [**-f** *ConfigurationFile*]

Description

The DFP daemon (**dfpd**) runs on the server being load balanced and provides load statistics about the server to the Load Manager. This enables the Load Manager to send future connections to the servers that are more available which helps in balancing the load.

When the **dfpd** daemon starts, it reads its configuration information from the file specified in the *ConfigurationFile* parameter. If the parameter is not specified, the **dfpd** daemon reads its configuration information from the **/etc/dfpd.conf** file.

Once started, the **dfpd** daemon listens for connections from the Load Manager on the port specified in the configuration file.

DFP daemon Configuration File

The **/etc/dfpd.conf** file can be updated by editing it. The entries in the **/etc/dfpd.conf** file include the following information:

The MD5 key entry specifies the secret key (up to 64 characters) that should be the same between the DFP clients, server and the Load Manager. An example of the MD5 key entry is:

```
md5key 1234567890abcdefabcdef12345678901234567890abcdefabcdef1234567890
```

The Load Manager listener entry specifies the port on which the DFP server listens for Load Manager connection. An example of the Load Manager entry is:

```
ldlistener 9503
```

The poll idle time entry specifies the period between successive computations of the CPU idle time. An example of the poll idle time entry is:

```
pollidletime 30
```

The computed idle time is multiplied by the *mfactor* value before reporting the time to the Load Manager. This is useful in rationalizing the weights among machines of different capacities. The default value is the number of CPUs on the host. An example of the mfactor entry is:

```
mfactor 1
```

Flags

Item	Description
-d	Runs in debug mode and does not become a daemon process.
-f *ConfigurationFile*	Causes the daemon to use the specified *ConfigurationFile*.

dfsck Command

Purpose

Checks and repairs two file systems simultaneously on different drives.

Syntax

dfsck [*FlagList1*] *FileSystem1* [*FlagList2*] *FileSystem2*

Description

The **dfsck** command lets you simultaneously check two file systems on two different drives. Use the *FlagList1* and *FlagList2* parameters to pass flags and parameters for the two sets of file systems. For a list of valid flags for *FlagList1* and *FlagList2*, see the flags section. Use a **-** (minus sign) to separate the file system groups if you specify flags as part of the arguments.

The **dfsck** command permits you to interact with two **fsck** commands at once. To aid in this, the **dfsck** command displays the file system name with each message. When responding to a question from the **dfsck** command, prefix your response with a 1 or a 2 to indicate whether the answer refers to the first or second file system group.

Attention: Do not use the **dfsck** command to check the root file system.

Flags

Item	Description
-d*BlockNumber*	Searches for references to a specified disk block. Whenever the **fsck** command encounters a file that contains a specified block, it displays the i-node number and all path names that refer to it.

Item	Description
-f	Performs a fast check. Under normal circumstances, the only file systems likely to be affected by halting the system without shutting down properly are those that are mounted when the system stops. The **-f** flag prompts the **fsck** command not to check file systems that were unmounted successfully. The **fsck** command determines this by inspecting the **s_fmod** flag in the file system superblock. This flag is set whenever a file system is mounted and cleared when it is unmounted successfully. If a file system is unmounted successfully, it is unlikely to have any problems. Because most file systems are unmounted successfully, not checking those file systems can reduce the checking time.
-i*i-NodeNumber*	Searches for references to a specified i-node. Whenever the **fsck** command encounters a directory reference to a specified i-node, it displays the full path name of the reference.
-n	Assumes a no response to all questions asked by the **fsck** command; does not open the specified file system for writing.
-o *Options*	Passes comma-separated options to the **fsck** command. These options are assumed to be file system implementation-specific, except that the following are currently supported for all file systems: **mountable** Causes the **fsck** command to exit with success, returning a value of 0, if the file system in question is mountable (clean). If the file system is not mountable, the **fsck** command exits returning with a value of 8. **mytype** Causes the **fsck** command to exit with success (0) if the file system in question is of the same type as either specified in the **/etc/filesystems** file or by the **-V** flag on the command line. Otherwise, 8 is returned. For example, `fsck -o mytype -V jfs /` exits with a value of 0 if **/** (the root file system) is a journaled file system.
-p	Does not display messages about minor problems but fixes them automatically. This flag does not grant the wholesale license that the **-y** flag does and is useful for performing automatic checks when the system is started normally. You should use this flag as part of the system startup procedures, whenever the system is being run automatically. Also allows parallel checks by group.
-t*File*	Specifies a *File* parameter as a scratch file on a file system other than the one being checked, if the **fsck** command cannot obtain enough memory to keep its tables. If you do not specify the **-t** flag and the **fsck** command needs a scratch file, it prompts you for the name of the scratch file. However, if you have specified the **-p** flag, the **fsck** command is unsuccessful. If the scratch file is not a special file, it is removed when the **fsck** command ends.
-V *VfsName*	Uses the description of the virtual file system specified by the *VFSName* variable for the file system instead of using the **/etc/filesystems** file to determine the description. If the **-V** *VfsName* flag is not specified on the command line, the **/etc/filesystems** file is checked and the **vfs=***Attribute* of the matching stanza is assumed to be the correct file system type.
-y	Assumes a yes response to all questions asked by the **fsck** command. This flag lets the **fsck** command take any action it considers necessary. Use this flag only on severely damaged file systems.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To simultaneously check two file systems on two different drives, enter:

   ```
   dfsck  -p /dev/hd1 -  -p /dev/hd7
   ```

   This command checks both file systems simultaneously, if the file systems on the **/dev/hd1** and **/dev/hd7** devices are located on two different drives. You can also specify the file system names found in the **/etc/filesystems** file.

Files

Item	Description
/usr/sbin/dfsck	Contains the **dfsck** command.
/etc/filesystems	Lists the known file systems and defines their characteristics.
/etc/vfs	Contains descriptions of virtual file system types.
/etc/rc	Contains commands (including the **fsck** command) that are run when the system is started.

dfshares Command

Purpose

Lists available resources from remote systems.

Syntax

dfshares [**-F** *FileSystemType*] [**-h**] [*Server* ...]

Description

The **dfshares** command provides information about resources that are available to the host through the Network File System. The **dfshares** command prints a header line, followed by a list of lines that contain white spaces as field separators.

For each resource, the following fields are displayed:

RESOURCE
Displays the resource name that is exported in the form of server:path.

SERVER
Displays the machine that is providing the resource.

ACCESS
Displays the access permissions granted to the client systems. However, **dfshares** cannot determine this information for a NFS resource and therefore populates the field with a hyphen ("-").

TRANSPORT
Displays the transport provider over which the resource is shared. However, **dfshares** cannot determine this information for a NFS resource and therefore populates the field with a hyphen ("-").

Flags

Item	Description
-F *FileSystemType*	Specifies the filesystem type. Only **nfs** type of filesystem is supported.
-h	Suppress the header line in the output of **dfshares**.

Parameters

Item	Description
Server	Represents a system on the network that has provided resources to the local machine. If this parameter is not specified, then the **dfshares** command prints the information for the local system, itself. More than one server name can be specified with **dfshares**.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Examples

1. To print the resource information on the system "mercury" for an **nfs** type filesystem, enter:

```
dfshares -F nfs mercury
```

2. To print resource information without header on the system, enter:

```
dfshares -hF nfs
```

Files

Item	Description
/usr/bin/dfshares	Contains the generic System V **dfshares** command.
/usr/lib/fs/nfs/dfshares	Contains the System V **dfshares** command for filesystems of type **nfs**.
/etc/vfs	Contains the descriptions for known virtual filesystem implementations.

dhcpaction Command

Purpose

Provides a script that runs every time that a client updates its lease.

Syntax

/usr/sbin/dhcpaction *HostName DomainName IPAddress LeaseTime ClientID* { **A** | **PTR** | **BOTH** | **NONE** } { **NONIM** | **NIM** }

Description

The **dhcpaction** command provides methods to update the DNS server by calling the **nsupdate** command with the appropriate sequence of events to update the A record, PTR record, or both.

The **dhcpaction** command is called by the DHCP client and server daemons. It is called from the updateDNS string. This setting is configurable because in some environments, mainly heterogeneous ones, some clients might not be able to update the A record or the PTR record. The default action is for the client to update the A record and the server to update the PTR record. The options might be set in the daemon configuration files to allow for any policy that the network administrator wants to implement.

The **dhcpaction** command also runs NIM and DHCP concurrently. The **dhcpaction** command, when provided with the NIM parameter, tries to issue updates to NIM objects when their IP addresses change. This action keeps the objects in sync. To do so, some pending operations might have to be canceled. The objects are commented and a message is sent to the console of the master machine. The objects must not be reset often. Addresses must not commonly change in the DHCP environment. Only the clients must set the NONIM option.

Parameters

Item	Description
ClientID	Specifies the client ID to use when you update the DNS server.
DomainName	Specifies the domain name to use when you update the DNS server.
HostName	Specifies the host name to update in the DNS server.
IPAddress	Specifies the IP address to associate with the host name in the DNS server.
LeaseTime	Specifies the duration of the association between the host name and IP address in the DNS server in seconds.

Options

Item	Description
A | PTR | BOTH | NONE	Specifies which record (if any) is to be updated in the DNS server.
NONIM | NIM	Specifies whether the script must run to help NIM and DHCP interact correctly. It must be set only to NIM on DHCP servers.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: Any user, but might need to be root for some NIM actions.

Files

Item	Description
/usr/sbin/dhcpaction	Contains the **dhcpaction** command.
/etc/dhcpcd.ini	Contains the DHCP client configuration file.

dhcpcd Daemon

Purpose

Implements a Dynamic Host Configuration Protocol (DHCP) client. Serves addresses and configuration information to the DHCP server.

Syntax

To implement a DHCP client by using the System Resource Controller (SRC):

startsrc -s dhcpcd [**-a** *Argument*] ...

To implement a DHCP client without using SRC:

dhcpcd [**-f** *ConfigurationFile*] [**-i** *IPAddress*] [**-l** *LeaseFile*] [**-n**] [**-o** *OptionsFile*] [**-r**] [**-t** *Seconds*] [**-T** *Minutes*]

Description

The **dhcpcd** daemon implements a DHCP client by setting up IP (Internet Protocol) addresses and other parameters by using the DHCP protocol.

The **dhcpcd** daemon is normally started by the `/etc/rc.tcpip` file that normally runs at start time. By default, it is commented out and not run on system startup. There are System Management Interface Tool (SMIT) options to enable the DHCP client.

The **dhcpcd** daemon reads its configuration file and attempts to start and get an IP address and other configuration options for the interfaces specified within the configuration file. The **dhcpcd** daemon runs in the background while the system is up. It renews an already received address as required.

The **dhcpcd** daemon also runs in DHCP Inform mode when the **-i** flag is used. By using this mode, a client can retrieve configuration information from a DHCP server without getting an IP address. It is useful for static addresses, but not for dynamic items like print servers and other options. When you use the **-i** flag with an IP address parameter, the **dhcpcd** daemon runs once for the specified address.

The **refresh** command can be used to cause the **dhcpcd** daemon to reread the configuration file. A SIGHUP might also be used to get the same response.

The default configuration file for the **dhcpcd** daemon is `/etc/dhcpcd.ini`. It contains logging and network interface information.

You can use the SMIT **smit usedhcp** fast path to run this command.

Flags

Item	Description
-f *ConfigurationFile*	Specifies the configuration file to be used. The default is the `/etc/dhcpcd.ini` file.
-i *IPAddress*	Specifies that the **dhcpcd** daemon must use DHCP Inform mode. The IP address tells DHCP on which interface to get configuration information.
-l *LeaseFile*	Specifies a different lease file. The lease file gets generated by the client when it obtains a lease. By default, the lease file is `/etc/dhcpc.db`.
-n	Prevents the interface from being reconfigured when it receives a new address.
-o *OptionsFile*	Specifies the options file. By default, the options file is `/etc/dhcpc.opt`.

Item	Description
-r	Brings up the client daemon and then down when run once.
-t *Seconds*	Specifies the number of seconds that DHCP waits before it places itself in the background. It allows a system to continue booting if a DHCP server cannot be found.
-T *Minutes*	Specifies the time in minutes. If the DHCP client fails to configure an address for an interface (for example, because of non-availability of DHCP server) within this timeout value, it stops further attempts.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Files

Item	Description
`/usr/sbin/dhcpcd`	Contains the **dhcpcd** daemon.
`/etc/dhcpcd.ini`	Contains the default client configuration file
`/etc/services`	Defines sockets and protocols that are used for internet services.
`/etc/inetd.conf`	Defines the services that are controlled by the **inetd** daemon.

dhcpcd6 Daemon

Purpose

Implements a Dynamic Host Configuration Protocol for IPv6 (DHCPv6) client. Obtains IPv6 addresses and configuration information for an IPv6 node from the DHCPv6 server.

Syntax

To start a DHCPv6 client by using the System Resource Controller (SRC):

startsrc -s dhcpcd6 [**-a** *Argument*] ...

To start a DHCPv6 client without using SRC:

dhcpcd6 [**-f** *ConfigurationFileName*] [**-u** *Client_duid_File*] [**-p** *ClientPort*] [**-t** *SolicitTimeout*]

Description

The **dhcpcd6** daemon implements a DHCPv6 client by setting up IPv6 (Internet Protocol version 6) addresses and other parameters by using the DHCPv6 protocol.

The **dhcpcd6** daemon is normally started by the `/etc/rc.net` file that normally runs at boot time. By default, it is commented out and not run on machine startup. The **dhcpcd6** daemon runs in the background while the system is up.

The **dhcpcd6** daemon reads its configuration file and attempts to bring up and get one or more IPv6 addresses and other configuration options for the interfaces specified within the configuration file. The addresses that are obtained from the server are renewed as mandated by the server.

When a DHCPv6 client does not need to have a DHCPv6 server assign it IPv6 addresses, the client can obtain only configuration information such as a list of available DNS servers or NTP servers. It is useful when the node is configured with static addresses.

The **refresh** command can be used to cause the **dhcpcd6** daemon to reread the configuration file. A SIGHUP might also be used to get the same response.

The default **dhcpcd6** configuration file is `/etc/dhcpv6/dhcpc6.cnf`. It contains logging and network interface information.

Flags

Item	Description
-f *ConfigurationFileName*	Specifies the configuration file to be used. Default is `/etc/dhcpv6/dhcpc6.cnf`.
-p *ClientPort*	Specifies the client port to be used. Default is 546.
-t *SolicitTimeout*	Specifies the time until the client solicits configuration information from the server before exiting.
-u *Client_duid_File*	Specifies the client identifier file to be used. Default is `/etc/dhcpv6/dhcpc6.duid`.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Examples

1. To start the DHCPv6 client with the configuration file `dhcpcd6.cnf` located in `/usr/local`, type the following command:

```
startsrc -s dhcpcd6 -a "-f /usr/local/dhcpcd6.cnf"
```

Location

`/usr/sbin/dhcpcd6`

Files

Item	Description
`/usr/sbin/dhcpcd6`	Contains the **dhcpcd6** client daemon.
`/etc/dhcpv6/dhcpc6.cnf`	Contains the default configuration file.
`/etc/dhcpv6/dhcpc6.db`	Contains the client lease file. This file is created by the client daemon and is not configurable.
`/etc/dhcpv6/dhcpc6.duid`	Contains the client identifier file. This file is created by the client daemon and is not configurable.

dhcprd Daemon

Purpose

Forwards BOOTP and Dynamic Host Configuration Protocol (DHCP) packets off the local network.

Syntax

To forward information to the DHCP server by using the System Resource Controller (SRC):

startsrc -s dhcprd [**-a** *Argument*] [**-a** *Argument*] ...

To forward information to the DHCP server without using SRC:

dhcprd [**-f** *ConfigurationFile*]

Description

The **dhcprd** daemon listens for broadcast packets, receives them, and forwards them to the appropriate server. It keeps broadcasts from having to be propagated to other networks. The DHCP relay agent handles the forwarding of the DHCP and BOOTP client broadcast packets off the local network and on to a set of servers. The initial packets that are sent by a BOOTP or DHCP client are broadcasts on the local interface of the client system. These packets are not allowed to be passed through network gateways and routers. A BOOTP or DHCP relay agent, the **dhcprd** daemon, sends these packets to the appropriate servers.

The DHCP Server reads `/etc/services` file to determine which port it must use for receiving requests. The default service is **dhcps**. Because it is the same port that the **bootpd** daemon uses, you can have only one (either **dhcprd** or **bootpd**) daemon running. If you choose the **dhcprd** daemon, you must uncomment **bootp** from the `/etc/inetd.conf` file, then type `refresh -s inetd` on the command line.

Note: If the **bootpd** daemon is running, this program must be stopped before you start the daemons.

Flags

Item	Description
-f *ConfigurationFile*	Specifies the configuration file to be used. The default is the `/etc/dhcprd.cnf` file.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Files

Item	Description
/usr/sbin/dhcprd	Contains the **dhcprd** daemon.
/etc/dhcprd.cnf	Contains the default configuration file.
/etc/services	Defines sockets and protocols that are used for internet services.
/etc/inetd.conf	Defines the services that are controlled by the **inetd** daemon.

dhcpsconf Command

Purpose

Simplifies DHCP (Dynamic Host Configuration Protocol) server configuration through a graphical user interface (GUI).

Syntax

dhcpsconf

Description

The **dhcpsconf** command opens an X Window System GUI that lets the network administrator read, save, and modify configuration files. It also lets you start, stop, and retrieve statistics from a running server.

The **dhcpsconf** command displays a set of lists. The lists on the left show the available options and keys. The **dhcpsconf** command reads the /etc/options file to determine its basic options and keys and starts with these as generic resource types. The GUI lets the network administrator define a set of named resources by selecting the resource menu button.

The resource definition dialog box lets the network administrator generate all the options and specifics that are on the networks. The network administrator can define and name the network, printers, name servers, DHCP servers, and other valid resource objects. Once it is done, these new resources are added to the key and option display on the main panel. These resources can be used to generate a server configuration file or set of server configuration files.

The GUI starts with an empty master file. A master file might contain either a single server or the definition of many servers and one actual server readable file. The master file is readable by one DHCP server, but multiple server information can be stored in it. It lets the network administrator configure a single server image of the network, create a set of servers to handle the same set of data, and view and maintain it all in one file.

Options and keys are added to the server window by selecting the key or option, selecting where in the edit window the option or key must go, and selecting the add button corresponding to the key or option section. The option is added to the edit window at the position specified. If the item is a named resource,

then it is added as is. If the item is one of the standard defaults, then a window that is requesting a value for the item appears.

DHCP servers are added just like other keys, except that they specify systems in the network that are responsible for the items within their scope. The keys have scoping and syntactic ordering. Comments are not really keys, but they are allowed anywhere.

A server might have a network, class, client, or options that are specified within it. A network might have a subnet, class, client, or option. A subnet might have a class, client, or options. A class and client might have only options.

The servers have a set of configuration parameters that apply only to them. These are specified by the DHCP server key in the key list, or by using the default server options under the Server menu bar. The default server options apply to the master file. A DHCP Server specified within the master file receives the default options, but may be modified.

Any item that is placed in the **Edit** window might be edited, renamed, viewed, or deleted. It lets you place an item, see whether it looks appropriate and change as necessary.

Upon completion of the configuration file, a single master file might be saved and a set of server files might be generated. The `File menu` button and `Server menu` button both have save options. The `File save` button is for saving the master file. The `Server save` button is for saving a particular server to a file.

The `File menu` button also contains a quit option, an open option to retrieve a file, and a new option to erase everything that is created so far.

The `Operations menu` button contains a status button, a start button, a stop button, a refresh, and a send configuration file button. From these buttons, a remote server can report status, refresh itself with a new configuration file, might be stopped, and a configuration file can be sent and restarted.

The `Help` button contains a set of help statements that describe each of the windows items.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: Any user

Files

Item	Description
`/usr/sbin/dhcpsconf`	Contains the **dhcpsconf** command.
`/etc/dhcpcd.cnf`	Contains the default client configuration file.

dhcpsd Daemon

Purpose

Implements a Dynamic Host Configuration Protocol (DHCP) server. Serves addresses and configuration information to DHCP clients.

Syntax

To serve information to the DHCP clients by using the System Resource Controller (SRC):

startsrc -s dhcpsd [**-a** *Argument*] [**-a** *Argument*] ...

To serve information to the DHCP clients without using SRC:

dhcpsd [**-f** *ConfigurationFile*]

Description

The DHCP server handles the assignment and maintenance of dynamic address assignment. It also handles the distribution of additional configuration information. The **dhcpsd** daemon runs in the background and maintains a database of server information that contains logging parameters, IP (Internet Protocol) address ranges, other network configuration information, and accessibility information. The initial database is specified by the configuration file. The configuration file contains all the data to start configuring DHCP clients.

The DHCP server maintains a database of addresses it provided and who has them. These databases are kept in the files `/etc/dhcpsd.ar` and `/etc/dhcpsd.cr`. A server on startup reads the configuration file and sets up its initial database of available addresses. The server accepts the **refresh** command or a SIGHUP signal to reread the configuration file.

The DHCP server reads `/etc/services` file to determine which port it must use for receiving requests. The default service is **dhcps**. Because it is the same port that the **bootpd** daemon uses, you can have only one (either **dhcpsd** or **bootpd**) daemon running. If you choose the **dhcpsd** daemon, you must comment **bootp** from the `/etc/inetd.conf` file, then enter `refresh -s inetd` on the command line.

Note: If the **bootpd** daemon is running, this program must be stopped before you start the daemons.

Flags

Item	Description
-f *ConfigurationFile*	Specifies the configuration file to be used.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Files

Item	Description
`/usr/sbin/dhcpsd`	Contains the **dhcpsd** daemon.
`/etc/services`	Defines sockets and protocols that are used for internet services.
`/etc/inetd.conf`	Defines the services that are controlled by the **inetd** daemon.

dhcpsdv6 Daemon

Purpose

Implements a Dynamic Host Configuration Protocol (DHCPv6) server. Serves addresses and configuration information to DHCPv6 clients.

Syntax

To serve information to the DHCPv6 clients by using the System Resource Controller (SRC):

startsrc -s dhcpsdv6 [**-a** *Argument*]

To serve information to the DHCP clients without using SRC:

dhcpsdv6 [**-d**] [**-f** *ConfigurationFile*] [**-a** *DadminPort*] [**-p** *ServerPort*]

Description

The DHCPv6 server handles the assignment and maintenance of dynamic address assignment. It also handles the distribution of additional configuration information. The **dhcpsd** daemon runs in the background and maintains a database of server information that contains logging parameters, IP (Internet Protocol) address ranges, other network configuration information, and accessibility information. The initial database is specified by the configuration file. The configuration file contains all the data to start configuring DHCP clients.

The DHCPv6 server maintains a database of addresses it provided and who has them. These databases are kept in the files `/etc/dhcpv6/db_file.crbk` and `/etc/dhcpv6/db_file.cr`. A server on startup reads the configuration file and setup its initial database of available addresses. The server accepts the refresh command or a `SIGHUP` signal to reread the configuration file.

Flags

Item	Description
-a	Specifies the Dadmin port; by default it is 942.
-d	Displays debugging information.
-f *ConfigurationFile*	Specifies the configuration file to be used. By default, the configuration file is `/etc/dhcpv6/dhcpsdv6.cnf`.
-p	Specifies the port that is used by the server to listen for incoming request; by default it is 547.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Examples

1. To start the DHCPv6 server with the configuration file `dhcpsdv6.cnf` located in `/usr/local`, type:

```
startsrc -s dhcpsdv6 -a "-f /usr/local/dhcpsdv6.cnf"
```

Location

`/usr/sbin/dhcpsdv6`

Files

Item	Description
`/usr/sbin/dhcpsdv6`	Contains the **dhcpsdv6** daemon.
`/etc/dhcpv6/db_file.cr`	Contains the client records. This file is created by the server daemon and is not configurable.
`/etc/dhcpv6/db_file.crbk`	Contains the client records. This file is created by the server daemon and is not configurable.
`/etc/dhcpv6/dhcpsdv6.duid`	Contains the server identifier file. This file is created by the server daemon and is not configurable.

diag Command

Purpose

Performs hardware problem determination.

Syntax

diag [[**-a**] | [**-s** [**-c**]] [**-E** *days*] [**-e**] | [**-d** *Device* [**-c**] [**-v**] [**-e**] [**-A**]] | [**-B** [**-c**]] | [**-T** *taskname*] | [**-S** *testsuite*] | [`-c` `-d` *Device* `-L pending | complete`]

Description

The **diag** command is the starting point to run a wide choice of tasks and service aids. Most of the tasks and service aids are platform-specific. The following tasks and service aids are available:

- Run diagnostics
- Display or change diagnostic run time options
- Display Service Hints
- Display previous diagnostic results
- Display hardware error report
- Display software product data
- Display configuration and resource list
- Display hardware vital product data
- Display resource attributes
- Change hardware vital product data
- Format media
- Certify media
- Display test patterns
- Local area network analyzer

- Add resource to resource list
- Delete resource from resource list
- SCSI bus analyzer
- Download microcode
- Display or change bootlist
- Periodic diagnostics
- Backup and restore media
- Disk maintenance
- Configure dials and LPFkeys
- Add or delete drawer Config
- Create customized configuration diskette
- Update disk based diagnostics
- Configure ISA adapter
- Operating system shell prompt (online service mode only)
- Display or change multiprocessor configuration
  - Enable and disable individual processors
- Display or change BUMP configuration
  - Update the flash EPROM with a new binary image
  - Display or change diagnostic modes
  - Display or change remote phone numbers and modem configurations
- Display or change electronic mode switch
- Process supplemental media (stand-alone mode only)
- Generic microcode download
- Run error log analysis
- Service aids for use with Ethernet
- 7135 RAIDiant array service aids
- SCSI device identification and removal
- SCSD tape drive service aid
- Escon bit error rate service aid
- PCI RAID physical disk identify
- Configure ring indicate Power On Policy (CHRP)
- Configure surveillance policy (CHRP)
- Configure reboot policy (CHRP)
- Configure remote maintenance policy (CHRP)
- Save or restore hardware management policies (CHRP)
- Display firmware device node information (CHRP)
- Spare sector availability
- Update system or service processor flash (CHRP)
- Display system environmental sensors (CHRP)
- Display checkstop analysis results
- Analyze adapter internal log
- Log repair action
- Flash SK-NET FDDI firmware

- Display microcode level
- Run the Non-Volatile Memory Express (NVMe) sanitize command on the NVMe-compliant devices that support the sanitize operation. The NVMe sanitize operation changes the Non-Volatile Memory (NVM) subsystem such that you cannot recover the previous user data from any cache or non-volatile media.

You can use the System Management Interface Tool (SMIT) **smit diag** fast path to run this command.

Flags

Note: Most users do not need to use any flags since the **diag** command is a menu driven program.

Item	Description
-A	Specifies advanced mode. You must also specify a device by using the **-d** flag.
-a	Processes any changes in the hardware configuration by asking if missing resources are removed, turned off, and so on. Missing resources (indicated by an 'M') and missing resource paths (indicated by a 'P') are integrated into the diagnostic resource selection list.
-B	Instructs diagnostic tests to run the base system test. Error log analysis is also done on areas in the base system that supports error log analysis.
-c	Indicates that the machine is not attended. No questions are asked. Results are written to standard output. You must also use an optional flag that specifies a device to be tested (**d**, **B**, or **s**).
-d *Device*	Specifies the device to run the diagnostic tests on.
-E *Days*	Specifies the number of days to be used for searching the error log during run error log analysis. This flag works with any other flag.
-e	Performs error log analysis if supported on the selected device. No tests are performed. This flag must be used with the **-d** flag, otherwise the resource selection menu appears. If used with the **-v** flag, the **-v** flag takes precedence and the **-e** flag is ignored.
-S *testsuite*	Indicates a particular test suite of devices to test: 1. Base system 2. I/O devices 3. Async devices 4. Graphic devices 5. SCSI devices 6. Storage devices 7. Common devices 8. Multimedia devices
-L pending | complete	Logs repair action for a resource that is specified with the `-d` and `-c` options. Use the `pending` parameter if the part is replaced, but it is not yet known whether this part remains in the system. Use the `complete` parameter if the part is replaced and it is known that this part remains in the system.
-s	Runs the diagnostic tests on all resources.

Item	Description
-T *taskname*	Specifies the specific Fastpath task to be run. The following list displays the current fastpath tasks : **format** Format media task **certify** Certify media task **download** Download microcode task **disp_mcode** Display microcode level task **chkspares** Spare sector availability task **identifyRemove** Hot plug task **nvme_sanitize** Sanitize NVMe-compliant devices that support the sanitize operation. When you specify the **nvme_sanitize** task, you must also specify the *force* option by using the **-f** flag. The **-f** flag is only required when the **-c** flag is specified with the **diag** command, otherwise the **diag** command fails with an error. **Note:** Tasks are platform and device dependent. Some tasks might not be available on the system.
-v	Runs the diagnostic tests in the system verification mode, no error log analysis performed. The default is Problem Determination mode that tests the device and runs error log analysis. If used with the **-e** flag, the **-v** flag takes precedence and the **-e** flag is ignored. Must be used with the **-d** flag to specify a device to run the diagnostic tests on.

Security

Access Control: Only the root user can run this command.

Privilege Control: System group.

Examples

1. To run the diagnostic tests on the `scdisk0` device, without questions, enter:

```
diag -d scdisk0 -c
```

2. To run the NVMe sanitize operation on the NVMe device (nvmeX), enter:

```
diag -c -d nvmeX -T "nvme_sanitize -f"
```

File

Item	Description
/usr/sbin/diag	Contains the **diag** command.

diaggetrto Command

Purpose

Displays diagnostic run-time options.

Syntax

diaggetrto [[**-a**] [**-d**] [**-l**] [**-m**] [**-n**] [**-p**] [**-s**]]

Description

The **diaggetrto** command displays the value of one or more diagnostic run time options. The following run-time options can be displayed with the **diaggetrto** command:

Display Diagnostic Mode Selection Menus
When this option is off, diagnostics run in Problem Determination mode only. The default is on.

Include Advanced Diagnostics
When this option is on, diagnostics run in advanced mode when run from the Task Selection Menu or command line. The default is off.

Number of days used to search error log
This option controls how old error log entries must be before they are no longer analyzed by diagnostics. The default is 7.

Display Progress Indicators
When this option is on, diagnostic applications that support progress indicators will display them. The default is on.

Diagnostic Event Logging
When this option is on, diagnostics log events. The default is on.

Diagnostic Event Log file size
This option controls the maximum size of the diagnostic event log. Allowable sizes are in increments of hundreds of kilobytes. The default is 100K.

Flags

Item	Description
-a	Displays the value of **Include Advanced Diagnostics**.
-d	Displays the value of **Diagnostic Event Logging**.
-l	Displays the value of **Diagnostic Event Log file size**.
-m	Displays the value of **Display Diagnostic Mode Selection Menus**.
-n	Displays the value of **Number of days used to search error log**.
-p	Displays the value of **Display Progress Indicators**.
-s	Displays all of the diagnostic run-time options.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Examples

1. To display the diagnostic event log size, type:

```
/usr/lpp/diagnostics/bin/diaggetrto -l
```

2. To check if progress indicators are turned on and to check if diagnostic event logging is turned on, type:

```
/usr/lpp/diagnostics/bin/diaggetrto -p -d
```

3. To display the number of days to search the error log, type:

```
/usr/lpp/diagnostics/bin/diaggetrto -n
```

Files

Item	Description
/usr/lpp/diagnostics/bin/diaggetrto	Contains the **diagsetrto** command.

diagrpt Command

Purpose

Displays previous diagnostic results.

Syntax

diagrpt [[**-o**] | [**-s** *mmddyy*] | [**-a**] | [**-r**]]

Description

The **diagrpt** command displays the results of previous diagnostic sessions. There are three types of results that can be viewed:

- Diagnostic result files stored in **/etc/lpp/diagnostic/data** directory.
- Diagnostic Event Log Information.
- Diagnostic results stored in NVRAM on CHRP systems.

Flags

Item	Description
-o	Displays the last diagnostic results file stored in the **/etc/lpp/diagnostics/data** directory.
-s *mmddyy*	Displays all diagnostic result files logged since the date specified.
-a	Displays the long version of the Diagnostic Event Log.
-r	Displays the short version of the Diagnostic Event Log.

Examples

1. To list all previous diagnostic result files since Jan 31, 1999, enter:

```
/usr/lpp/diagnostics/bin/diagrpt -s 013199
```

2. To view the short version of the diagnostic event log, enter:

```
/usr/lpp/diagnostics/bin/diagrpt -r
```

File

Item	Description
/usr/lpp/diagnostics/bin/diagrpt	Contains the **diagrpt** command.

diagsetrto Command

Purpose

Sets diagnostic run-time options.

Syntax

diagsetrto [[**-a on** | **off**] [**-d on** | **off**] [**-l** *Size*] [**-m on** | **off**] [**-n** *Days*] [**-p on** | **off**]]

Description

The **diagsetrto** command sets the value of any number of diagnostic run-time options. The following run-time options can be altered with the **diagsetrto** command:

Display Diagnostic Mode Selection Menus
When this option is off, diagnostics run in Problem Determination mode only. The default is on.

Include Advanced Diagnostics
When this option is on, diagnostics run in advanced mode when run from the Task Selection Menu or command line. The default is off.

Number of Days Used to Search Error Log
This option controls how old error log entries must be before they are no longer analyzed by diagnostics. The default is 7.

Display Progress Indicators
When this option is on, diagnostic applications that support progress indicators will display them. The default is on.

Diagnostic Event Logging
When this option is on, diagnostics log events. The default is on.

Diagnostic Event Log File Size
This option controls the maximum size of the diagnostic event log. Allowable sizes are in increments of hundreds of kilobytes. The default is 100K.

Flags

Item	Description
-a on | **off**	Sets the value of **Include Advanced Diagnostics**.
-d on | **off**	Sets the value of **Diagnostic Event Logging**.
-l *Size*	Sets the value of **Diagnostic Event Log file size**.
-m on | **off**	Sets the value of **Display Diagnostic Mode Selection Menus**.
-n *Days*	Sets the value of **Number of days used to search the error log**.
-p on | **off**	Sets the value of **Display Progress Indicators**.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Examples

1. To set the diagnostic event log size to 500K, type:

```
/usr/lpp/diagnostics/bin/diagsetrto -l 500
```

2. To turn off progress indicators and turn off diagnostic event logging, type:

```
/usr/lpp/diagnostics/bin/diagsetrto -p off -d off
```

3. To set the number of days to search the error log to 50, type:

```
/usr/lpp/diagnostics/bin/diagsetrto -n 50
```

Files

Item	Description
/usr/lpp/diagnostics/bin/diagsetrto	Contains the **diagsetrto** command.

diction Command

Purpose

Highlights unclear or wordy sentences.

Syntax

diction [**-ml**] [**-mm**] [**-f** *PatternFile*] [**-n**] *File* ...

Description

The **diction** command finds all sentences in an English-language document that contain phrases from a database of unclear or wordy diction. Each phrase is bracketed with [ ] (brackets). Because the **diction** command runs the **deroff** command before looking at the text, include header files that contain appropriate formatting information as part of the input. The **explain** command provides an interactive thesaurus for the phrases found by the **diction** command.

Use of nonstandard formatting macros may cause incorrect sentence breaks. In particular, the **diction** command does not understand the **-me** flag.

Flags

Item	Description
-f *PatternFile*	Specifies a file containing examples of unclear diction; this file is used in addition to the default file.
-ml	Causes the **deroff** command to skip **mm** macro lists; can be used if a document contains many lists of sentence fragments.
-mm	Overrides the default **ms** macro package.

Item	Description
-n	Suppresses the use of the default file when used with the **-f** flag; only the file specified by the *PatternFile* parameter is used.

Files

Item	Description
/usr/lib/dict.d	Contains default pattern.

diff Command

Purpose

Compares text files.

Syntax

To Compare the Contents of Two Files

diff [**-c**| **-C** *Lines* | **-D** [*String*] | **-e** | **-f** | **-n** | **-u** | **-U** *Lines*] [**-b**] [**-i**] [**-t**] [**-w**] *File1 File2*

diff [**-h**] [**-b**] *File1 File2*

To Sort the Contents of Directories and Compare Files That Are Different

diff [**-c** | **-C** *Lines* | **-e** | **-f** | **-n** | **-u** | **-U** *Lines*] [**-b**] [**-i**] [**-l**] [**-r**] [**-s**] [**-S** *File*] [**-t**] [**-w**] *Directory1 Directory2*

diff [**-h**] [**-b**] *Directory1 Directory2*

Description

The **diff** command compares text files. It can compare single files or the contents of directories.

Note: The **diff** command only works with input files that are text files.

If the *Directory1* and *Directory2* parameters are specified, the **diff** command compares the text files that have the same name in both directories. Binary files that differ, common subdirectories, and files that appear in only one directory are listed.

When the **diff** command is run on regular files, and when comparing text files that differ during directory comparison, the **diff** command tells what lines must be changed in the files to make them agree. If neither the *File1* nor *File2* parameter is a directory, then either may be given as - (minus sign), in which case the standard input is used. If the *File1* parameter is a directory, then a file in that directory whose file name is the same as the *File2* parameter is used.

The typical output contains lines of these forms:

```
Lines Affected in File1   Action        Lines Affected in File2
Number1                   a             Number2[,Number3]
Number1[,Number2]         d             Number3
Number1[,Number2]         c             Number3[,Number4]
```

These lines resemble **ed** subcommands to convert *File1* into *File2*. The numbers before the action letters pertain to *File1*; those after pertain to *File2*. Thus, by exchanging **a** for **d** and reading from right to left, you can also tell how to convert *File2* into *File1*. As in the **ed** command, identical pairs (where *Number1* = *Number2*) are abbreviated as a single number.

Following each of these lines, the **diff** command displays all lines affected in the first file preceded by a **<:** (less than sign, colon), then displays all lines affected in the second file are preceded by a **>** (greater than sign).

An exit value of 0 indicates no differences, 1 indicates differences found, and 2 indicates an error.

Note: If more than one of the **-c**, **-C**, **-D**, **-e**, **-f**, or **-n**, **-u**, or **-U** flags are specified, the last one on the command line takes precedence. The system does not issue an error message.

Flags

Item	Description
-b	Causes any amount of white space at the end of a line to be treated as a single newline character (the white-space characters preceding the newline character are ignored) and other strings of white-space characters, not including newline characters, to compare equally.
-C *Lines*	Produces a **diff** command comparison with a number of lines of copied context equal to the value specified by the *Lines* variable. The **-C** flag modifies the output slightly. The output begins with identification of the files involved and their creation dates. Each change is separated by a line with a dozen * (asterisks). The lines removed from *File1* are marked with a - (minus sign) and those added to *File2* are marked with a + (plus sign). Lines changed from one file to the other are marked in both files with an ! (exclamation point). Changes that lie within the specified copied context lines of each other are grouped together as output.
-c	Produces a **diff** command comparison with three lines of copied context. The **-c** flag modifies the output slightly. The output begins with identification of the files involved and their creation dates. Each change is separated by a line with a dozen * (asterisks). The lines removed from *File1* are marked with a - (minus sign) and those added to *File2* are marked with a + (plus sign). Lines changed from one file to the other are marked in both files with an ! (exclamation point). Changes within the specified copied context lines of each other are grouped together as output.
-D [*String*]	Causes the **diff** command to create a merged version of *File1* and *File2* on the standard output. The C preprocessor controls are included so that a compilation of the result without defining *String* is equivalent to compiling *File1*, while defining *String* yields *File2.*
-e	Produces output in a form suitable for use with the **ed** editor to convert *File1* to *File2*. When using this flag, the following shell program may help maintain multiple versions of a file. Only an ancestral file (**$1**) and a chain of version-to-version **ed** scripts (**$2**, **$3**, ...) made by the **diff** command need to be on hand. The latest version appears on the standard output as follows: `(shift; cat $*; echo '1,$p') | ed - $1` Extra commands are added to the output when the **-e** flag is used to compare directories, so the result is a shell script for converting text files that are common to the two directories from their state in *Directory1* to their state in *Directory2.* **Note:** Editing scripts produced by the **-e** or **-f** flags cannot create lines consisting of a single . (period).
-f	Produces output in a form not suitable for use with the **ed** editor, showing the modifications necessary to convert *File1* to *File2* in the reverse order of that produced under the **-e** flag.
-h	Performs an alternate comparison that may be faster if the changed sections are short and well separated. The **-h** flag works on files of any length. The **-c**, **-C**, **-D**, **-e**, **-f**, and **-n** flags cannot be used with the **-h** flag. All other flags except the **-b** flag are ignored when used with the **-h** flag.

Item	Description
-i	Ignores the case of letters. For example, a lowercase **a** is treated the same as an uppercase **A**.
-l	Long output format. Each result from the **diff** command text file comparison is piped through the **pr** command for pagination. Other differences are remembered and summarized after all text file differences are reported.
-n	Produces output similar to that of the **-e** flag, but in the opposite order and with a count of changed lines on each insert or delete command. This is the form used by the revision control system (RCS).
-r	Causes application of the **diff** command recursively to common subdirectories encountered.
-s	Reports files that are the same and otherwise not mentioned.
-S [*File*]	Ignores files whose names collate before the file specified by the *File* variable when comparing directories. The **-S** flag only applies to the directories specified in the *Directory1* and *Directory2* parameters. If you use the **-r** flag with the **-S** flag, the **-S** flag does not work recursively in the *Directory1* and *Directory2* subdirectories.
-t	Expands tabs in output lines. Typical output or the **-c** flag output adds characters to the front of each line, which may affect indentation of the original source lines and makes the output listing difficult to interpret. This flag preserves the original source's indentation.
-u	Produces a **diff** command comparison with three lines of unified context. The output is similar to that of the **-c** flag, except that the context lines are not repeated; instead, the context, deleted, and added lines are shown together, interleaved.
-U *Lines*	Produces a **diff** command comparison with a number of lines of unified context equal to the value specified by the *Lines* variable. The output is similar to that of the **-C** flag, except that the context lines are not repeated; instead, the context, deleted, and added lines are shown together, interleaved.
-w	Ignores all spaces and tab characters and treats all other strings of blanks as equivalent. For example, `if ( a == b )` compares equally to `if(a==b)`.

Exit Status

This command returns the following exit values:

Item	Description
0	No differences were found.
1	Differences were found.
>1	An error occurred.

Examples

1. To compare two files, enter:

   ```
   diff chap1.back chap1
   ```

   This displays the differences between the files `chap1.bak` and `chap1`.
2. To compare two files while ignoring differences in the amount of white space, enter:

```
diff  -w prog.c.bak prog.c
```

If two lines differ only in the number of spaces and tabs between words, the **diff -w** command considers them to be the same.

3. To create a file containing commands that the **ed** command can use to reconstruct one file from another, enter:

```
diff  -e chap2 chap2.old >new.to.old.ed
```

This creates a file named `new.to.old.ed` that contains the **ed** subcommands to change `chap2` back into the version of the text found in `chap2.old`. In most cases, `new.to.old.ed` is a much smaller file than `chap2.old`. You can save disk space by deleting `chap2.old`, and you can reconstruct it at any time by entering:

```
(cat new.to.old.ed ; echo '1,$p') | ed - chap2 >chap2.old
```

The commands in parentheses add `1,$p` to the end of the editing commands sent to the **ed** editor. The `1,$p` causes the **ed** command to write the file to standard output after editing it. This modified command sequence is then piped to the **ed** command (`|  ed`), and the editor reads it as standard input. The **-** flag causes the **ed** command not to display the file size and other extra information because it would be mixed with the text of `chap2.old`.

Files

Item	Description
/usr/bin/diff	Contains the **diff** command.

diff3 Command

Purpose

Compares three files.

Syntax

diff3 [**-e** | **-x** | **-E** | **-X** | **-3**] *File1 File2 File3*

Description

The **diff3** command compares three files and writes to standard output the ranges of text that differ, flagged with the following codes:

Item	Description
====	All three files differ.
====1	*File1* differs.
====2	*File2* differs.
====3	*File3* differs.

The type of change needed to convert a given range of a given file to match another file is indicated in one of these two ways in the output:

Item	Description
File:*Number1* **a**	Text is to be added after line number *Number1* in *File*, where *File* is **1**, **2**, or **3**.

Item	Description
File:*Number1*[,*Number2*]**c**	Text in the range line *Number1* to line *Number2* is to be changed. If *Number1* is the same as *Number2*, the range may be abbreviated to *Number1*.

The contents of the range follows a **c** indication. When the contents of two files are identical, the **diff3** command does not show the contents of the lower-numbered file, although it shows the location of the identical lines for each.

Note: Edit scripts produced by the **-e** flag cannot create lines consisting of a . (period).

Flags

Item	Description
-3	Produces an edit script to incorporate only changes flagged ====3.
-E, **-X**	These are similar to **-e** and **-x** respectively, but treat overlapping changes (that is, changes that would be flagged ==== in the normal listing) differently. The overlapping lines from both files are inserted by the edit script, bracketed by <<<<<< and >>>>>> lines. The **-E** option is used by Revision Control System (RCS) Merge to ensure that overlapping changes in the merged files are preserved and brought to someone's attention.
-e	Creates an edit script for use with the **ed** command to incorporate into *File1* all changes between *File2* and *File3* (that is, the changes that normally would be flagged ==== and ====3).
-x	Produces an edit script to incorporate only changes flagged ====.

Examples

To list the differences among three files:

```
diff3 fruit.a fruit.b fruit.c
```

If `fruit.a`, `fruit.b`, and `fruit.c` contain the following data:

```
fruit.a                 fruit.b                 fruit.c
banana                  apple                   grape
grape                   banana                  grapefruit
kiwi                    grapefruit              kiwi
lemon                   kiwi                    lemon
mango                   orange                  mango
orange                  peach                   orange
peach                   pear                    peach
pare
```

then the output from the **diff3** command shows the differences between these files as follows. (The comments on the right do not appear in the output.)

```
====            All three files are different.
1:1,2c          Lines 1 and 2 of the first file, fruit.a
  banana
  grape
2:1,3c          Lines 1 through 3 of fruit.b
  apple
  banana
  grapefruit
3:1,2c          Lines 1 and 2 of fruit.c
  grape
  grapefruit
====2           The second file, fruit.b, is different.
1:4,5c          Lines 4 and 5 the same in fruit.a and fruit.c.
2:4a            To make fruit.b look same, add after line 4.
3:4,5c
  lemon
  mango
```

```
====            The first file, fruit.a, is different.
1:8c
  pare
2:7c            fruit.b line 7 and fruit.c line 8 are the same
  pear
3:7a
```

Files

Item	Description
/usr/bin/diff3	Indicates the **diff3** command.
/usr/lbin/diff3prog	Called by the **diff3** shell script.

diffmk Command

Purpose

Marks differences between files.

Syntax

diffmk [{ **-ab***X* | **-ae***X*] [**-b**] [**-cb***X* | **-ce***X*] [**-db***X* | **-de***X*] *File1 File2* [*File3*]

Description

The **diffmk** command compares the English-language file specified by the *File1* parameter with the file by the *File2* parameter. It then creates a third file that includes **.mc** requests (for creating change marks) for the **nroff** and **troff** commands. The *File1* and *File2* parameters specify the old and new versions, respectively, of the files. The **diffmk** command writes the newly created file to the *File3* parameter, if specified, or else to standard output. The *File3* file contains the lines of the *File2* file plus inserted formatter **.mc** requests. When the *File3* file is formatted, the changed or inserted text is marked by a | (vertical bar) at the right margin of each line. An * (asterisk) in the margin indicates that a line was deleted.

If the **DIFFMARK** environment variable is defined, it names a command string that the **diffmk** command uses to compare the files. (Normally, the **diffmk** command uses the **diff** command.) For example, to handle extremely large files better, you can set the **DIFFMARK** variable to `diff -h`.

Parameters

Item	Description
File1	Specifies an English-language file that is compared to the file specified by the *File2* parameter. The results of the comparison comprise the file specified by the *File3* parameter. *File1* is considered the "old" file.
File2	Specifies an English-language file that is compared to the file specified by the *File1* parameter. The results of the comparison comprise the file specified by the *File3* parameter. *File2* is considered the "new" file.
File3	Specifies a file that contains lines of the *File2* file and includes inserted formatter **.mc** requests for the **nroff** and **troff** commands. The contents of this file are the results of a comparison between the files specified by the *File1* and *File2* parameters. When formatted, the changed text is marked by a (|) vertical bar at the right margin of each line. An * (asterisk) indicates the line was deleted. If *File3* is not specified, the results of the comparison are written to standard input.

Flags

Item	Description
-ab*X*	Uses *X* to mark where added lines begin.
-ae*X*	Uses *X* to mark where added lines end.
-b	Ignores differences that are only changes in tabs or spaces on a line.
-cb*X*	Uses *X* to mark where changed lines begin.
-ce*X*	Uses *X* to mark where changed lines end.
-db*X*	Uses *X* to mark where deleted lines begin.
-de*X*	Uses *X* to mark where deleted lines end.

Examples

1. To mark the differences between two versions of a text file, enter:

   ```
   diffmk chap1.old chap1 chap1.nroff
   ```

   This produces a copy of `chap1` containing **nroff** and **troff** change mark requests to identify text that has been added to, changed in, or deleted from `chap1.old`. This copy is saved in the `chap1.nroff` file.
2. To mark differences with non-**nroff** and **troff** messages, enter:

   ```
   diffmk -ab'>>New:' -ae'<<End New' \
   chap1.old chap1 chap1.nroff
   ```

   This causes the **diffmk** command to write `>>New:` on the line before a section of newly added lines to `chap1`, and to write `<<End New` on the line following the added lines. Changes and deletions still generate **nroff** and **troff** commands to put a | (vertical bar) or * (asterisk) in the margin.
3. To use different **nroff** and **troff** command-marking requests and ignore changes in white space, enter:

   ```
   diffmk -b -cb'.mc %' chap1.old chap1 chap1.nroff
   ```

   This imbeds commands that mark changes with `%` (percent sign) additions with a | (vertical bar), and deletions with an * (asterisk). It does not mark changes that only involve a different number of spaces or tabs between words (`-b`).

dig Command

Purpose

DNS lookup utility.

Syntax

dig [*@server*] [**-b** *address*] [**-c** *class*] [**-f** *filename*] [**-k** *filename*] [**-p** *port#*] [**-q** *name*] [**-t** *type*] [**-x** *addr*] [**-y** [hmac:] *name:key*] [**-4**] [**-6**] [*name*] [*type*] [*class*] [*queryopt...*]

dig [**-h**]

dig [*global-queryopt...*] [*query...*]

Description

The **dig** (domain information groper) command is a flexible tool for interrogating DNS name servers. It performs DNS lookups and displays the answers that are returned from the queried name server(s). Most

DNS administrators use the **dig** command to troubleshoot DNS problems because of its flexibility, ease of use, and clarity of output. Although **dig** is normally used with command-line arguments, it also has a batch mode for reading lookup requests from a file. Unlike earlier versions, the BIND9 implementation of **dig** allows multiple lookups to be issued from the command line. Unless it is told to query a specific name server, the **dig** command tries each of the servers listed in the **/etc/resolv.conf** file. If you specify no command line arguments or options, the **dig** command performs an NS query for "**.**" (the root).

It is possible to set per-user defaults for the **dig** command through the **${HOME}/.digrc** file. The **dig** command reads this file and applies any options in it before the command line arguments.

The **IN** and **CH** class names overlap with the **IN** and **CH** top level domains names. When you look up these top level domains, you can either use the **-t** and **-c** options to specify the type and class or use the **-q** option to specify the domain name or use the **IN** and **CH** names.

Flags

Item	Description
-b *address*	Sets the source IP address of the query-to address. This must be a valid address on one of the host's network interfaces or "0.0.0.0" or "::". You can specify an optional port by appending "#*port*".
-c *class*	Overrides the default query class (IN for internet). The ***class*** parameter value is any valid class, such as HS for Hesiod records or CH for CHAOSNET records.
-f *filename*	Makes the **dig** command operate in batch mode by reading a list of lookup requests to process from the specified file name. The file contains a number of queries; one per line. Each entry in the file must be organized in the same way they are presented as queries to the **dig** command using the command-line interface.
-h	Prints a brief summary of command-line arguments and options.
-k *filename*	Specifies a TSIG key file using the **-k** option to sign the DNS queries sent by the **dig** command.
-p *port#*	Queries a non-standard port number. The *port#* parameter value is the port number that the **dig** command sends its queries to instead of the standard DNS port number 53. You can use this option to test a name server that has been configured to listen for queries on a non-standard port number.
-q *name*	Distinguishes the name from other arguments. Sets the query name to the *name* parameter value specified.
-t *type*	Sets the query type to the ***type*** parameter value. It can be any valid query type that is supported in BIND9. The default query type is **A**, unless the **-x** option is supplied to indicate a reverse lookup. A zone transfer can be requested by specifying a ***type*** of AXFR. When an incremental zone transfer (IXFR) is required, the ***type*** parameter value is set to **ixfr=N**. The incremental zone transfer contains the changes made to the zone because the serial number in the zone's SOA record was **N**.
-x *addr*	Simplifies reverse lookups (mapping addresses to names). The ***addr*** parameter value is an IPv4 address in dotted-decimal notation, or a colon-delimited IPv6 address. When you use this option, there is no need to provide the *name*, *class*, and *type* arguments. The **dig** command automatically performs a lookup for a name like `11.12.13.10.in-addr.arpa` and sets the query type and class to PTR and IN respectively.

Item	Description
-y [hmac:] *name:key*	Specifies the TSIG key itself on the command line; *hmac* is the type of the TSIG. The default value is HMAC-MD5. The ***name*** parameter value is the name of the TSIG key and the ***key*** parameter value is the actual key. The key is a base-64 encoded string, typically generated by **dnssec-keygen**(8). Caution must be taken when using the **-y** option on multi-user systems as the key can be visible in the output from ps(1) or in the shell's history file. When using TSIG authentication with the **dig** command, the name server that is queried needs to know the key and algorithm that is being used. In BIND, this is done by providing appropriate key and server statements in the **named.conf** file.
-4	Forces the **dig** command to only use the IPv4 query transport.
-6	Forces the **dig** command to only use the IPv6 query transport.

Parameters

Item	Description
global-queryopt...	Global query option (see Multiple Queries).
query	Query option (see Query Options).

Query Options

The **dig** command provides a number of query options that affect the way in which lookups are made and the results displayed. Some of these set or reset flag bits in the query header, some determine which sections of the answer get printed, and others determine the timeout and retry strategies. Each query option is identified by a keyword preceded by a plus sign (+). Some keywords set or reset an option. These can be preceded by the string no to negate the meaning of that keyword. Other keywords assign values to options like the timeout interval. They have the form **+keyword=value**. The query options are:

+[no]tcp
Use or do not use TCP when querying name servers. The default behavior is to use UDP unless an AXFR or IXFR query is requested, in which case a TCP connection is used.

+[no]vc
Use or do not use TCP when querying name servers. This alternate syntax to **+[no]tcp** is provided for backwards compatibility. The **vc** stands for virtual circuit.

+[no]ignore
Ignore truncation in UDP responses instead of retrying with TCP. By default, TCP retries are performed.

+domain=somename
Set the search list to contain the single domain ***somename***, as if specified in a domain directive in the **/etc/resolv.conf** file, and enable search list processing as if the **+search** option was given.

+[no]search
Use or do not use the search list defined by the search list or domain directive in the **/etc/resolv.conf** file (if any). The search list is not used by default.

+[no]defname
Deprecated, treated as a synonym for **+[no]search**.

+[no]aaonly
Sets the "aa" flag in the query.

+[no]adflag
Set or do not set the AD (authentic data) bit in the query. The AD bit currently has a standard meaning only in responses, not in queries, but the ability to set the bit in the query is provided for completeness.

+[no]cdflag
Set or do not set the CD (checking disabled) bit in the query. This requests the server to not perform DNSSEC validation of responses.

+[no]cl
Display or do not display the CLASS when printing the record.

+[no]ttlid
Display or do not display the TTL when printing the record.

+[no]recursive
Toggle the setting of the RD (recursion desired) bit in the query. This bit is set by default, which means **dig** normally sends recursive queries. Recursion is automatically disabled when the **+nssearch** or **+trace** query options are used.

+[no]nssearch
When this option is set, the **dig** command attempts to find the authoritative name servers for the zone containing the name being looked up and display the SOA record that each name server has for the zone.

+[no]trace
Toggle tracing of the delegation path from the root name servers for the name being looked up. Tracing is disabled by default. When tracing is enabled, the **dig** command makes iterative queries to resolve the name being looked up. It follows referrals from the root servers, showing the answer from each server that was used to resolve the lookup.

+[no]cmd
Toggle the printing of the initial comment in the output identifying the version of **dig** and the query options that have been applied. This comment is printed by default.

+[no]short
Provide a terse answer. The default is to print the answer in a verbose form.

+[no]identify
Show or do not show the IP address and port number that supplied the answer when the **+short** option is enabled. If short form answers are requested, the default is not to show the source address and port number of the server that provided the answer.

+[no]comments
Toggle the display of comment lines in the output. The default is to print comments.

+[no]stats
Toggle the printing of statistics: when the query was made, the size of the reply, and so on. The default behavior is to print the query statistics.

+[no]qr
Print or do not print the query as it is sent. By default, the query is not printed.

+[no]question
Print or do not print the question section of a query when an answer is returned. The default is to print the question section as a comment.

+[no]answer
Display or do not display the answer section of a reply. The default is to display it.

+[no]authority
Display or do not display the authority section of a reply. The default is to display it.

+[no]additional
Display or do not display the additional section of a reply. The default is to display it.

+[no]all
Set or clear all display flags.

+time=T
Set the timeout for a query to ***T*** seconds. The default timeout is 5 seconds. An attempt to set the ***T*** parameter value to less than 1 results in a query timeout of 1 second being applied.

+tries=A
Set the number of times to try UDP queries to server to the ***A*** parameter value instead of the default, 3. If the ***A*** parameter value is less than or equal to zero, the number of retries is silently rounded up to 1.

+retry=T
Set the number of times to retry UDP queries to server to the ***T*** parameter value instead of the default, 2. Unlike **+tries**, this does not include the initial query.

+ndots=D
Set the number of dots that have to appear in name to the ***D*** parameter value as it is considered absolute. The default value is one that is defined using the **ndots** statement in the **/etc/resolv.conf** file, or 1 if no **ndots** statement is present. Names with fewer dots are interpreted as relative names and is searched for in the domains listed in the search or domain directive in the **/etc/resolv.conf** file.

+bufsize=B
Set the UDP message buffer size advertised using EDNS0 to ***B*** bytes. The maximum and minimum sizes of this buffer are 65535 and 0, respectively. Values outside of this range are rounded up or down appropriately. Values other than zero cause an EDNS query to be sent.

+edns=#
Specify the EDNS version to query with. Valid values are 0 to 255. Setting the EDNS version causes a EDNS query to be sent. **+noedns** clears the remembered EDNS version.

+[no]multiline
Print records like the SOA records in a verbose multi-line format with human-readable comments. The default is to print each record on a single line to facilitate machine parsing of the **dig** output.

+[no]fail
Do not try the next server if you receive a SERVFAIL. The default is not to try the next server which is the reverse of normal stub resolver behavior.

+[no]besteffort
Attempt to display the contents of messages that are malformed. The default is not to display malformed answers.

+[no]dnssec
Request DNSSEC records to be sent by setting the DNSSEC OK bit (DO) in the OPT record in the additional section of the query.

+[no]sigchase
Chase DNSSEC signature chains. Require the **dig** command to be compiled with -DDIG SIGCHASE.

+trusted-key=####
Specify a file containing trusted keys to be used with **+sigchase**. Each DNSKEY record must be on its own line. If not specified, the **dig** command looks for the **/etc/trusted-key.key** file then the **trusted-key.key** file in the current directory. Require the **dig** command to be compiled with -DDIG SIGCHASE.

+[no]topdown
When chasing DNSSEC signature chains, perform a top down validation. Require the **dig** command to be compiled with -DDIG SIGCHASE.

Multiple Queries

The BIND 9 implementation of **dig** supports specifying multiple queries on the command line (in addition to supporting the **-f** batch file option). Each of those queries can be supplied with its own set of flags, options and query options.

In this case, each query argument represents an individual query in the command-line syntax. Each consists of any of the standard options and flags, the name to be looked up, an optional query type, class, and any query options that must be applied to that query.

A global set of query options, which must be applied to all queries, can also be supplied. These global query options must precede the first tuple of name, class, type, options, flags, and query options supplied

on the command line. Any global query options (except the **+[no]cmd** option) can be overridden by a query-specific set of query options. For example:

```
dig +qr www.isc.org any -x 127.0.0.1 isc.org ns +noqr
```

This **dig** command string shows how the **dig** command could be used from the command line to make three lookups: an ANY query for **www.isc.org**, a reverse lookup of 127.0.0.1, and a query for the NS records of **isc.org**. A global query option of **+qr** is applied, so that the **dig** command shows the initial query it made for each lookup. The final query has a local query option of **+noqr**, which means that the **dig** command does not print the initial query when it looks up the NS records for **isc.org**.

IDN SUPPORT

If the **dig** command has been built with internationalized domain name (IDN) support, it can accept and display non-ASCII domain names. The **dig** command appropriately converts character encoding of domain name before sending a request to the DNS server or displaying a reply from the server. If you would like to turn off the IDN support for some reason, define the IDN DISABLE environment variable; the following IDN support is disabled if the variable is set when the **dig** command runs.

Examples

A typical invocation of **dig** looks like:

```
dig @server name type
```

where:

server
The name or IP address of the name server to query. This can be an IPv4 address in dotted-decimal notation or an IPv6 address in colon-delimited notation. When the supplied server argument is a hostname, the **dig** command resolves that name before querying that name server. If no server argument is provided, the **dig** command consults the **/etc/resolv.conf** file and queries the name servers listed there. The reply from the name server that responds is displayed.

name
The name of the resource record that is to be looked up.

type
Indicates what type of query is required — **ANY**, **A**, **MX**, **SIG**, and so on. The *type* argument value can be any valid query type. If no *type* argument is supplied, the **dig** command performs a lookup for an ***A*** record.

Files

Item	Description
/etc/resolv.conf	
${HOME}/.digrc	

digest Command

Purpose

Converts the ASCII form of the **/etc/qconfig** file into the **/etc/qconfig.bin** file, a binary version of the queue configuration used by the **qdaemon** command. This command should not be entered on the command line; it is called by the **qdaemon** command.

Syntax

/usr/lib/lpd/digest *ASCIIFile BinaryFile*

Description

The **digest** command accepts an input file of ASCII characters and converts it into a binary file. This command is only used by the **qdaemon** command to translate the **/etc/qconfig** file into the binary version of the file, the **/etc/qconfig.bin** file.

Files

Item	Description
/etc/qconfig	Contains the queue configuration file.
/usr/sbin/qdaemon	Contains the queuing daemon.
/etc/qconfig.bin	Contains the digested, binary version of the **/etc/qconfig** file.

dircmp Command

Purpose

Compares two directories and the contents of their common files.

Syntax

dircmp [**-d**] [**-s**] [**-w** *num*] *Directory1 Directory2*

Description

The **dircmp** command compares the two directories specified by the *Directory1* and *Directory2* parameters and writes information about their contents to standard output. First, the **dircmp** command compares the file names in each directory. If the same file name appears in both, the **dircmp** command compares the contents of both files.

In the output, the **dircmp** command lists the files unique to each directory. It then lists the files with identical names in both directories, but with different contents. If no flag is specified, it also lists files that have identical contents as well as identical names in both directories.

The **diff -r** command offers a function similar to the **dircmp** command.

Flags

Item	Description
-d	Displays for each common file name both versions of the differing file contents. The display format is the same as that for the **diff** command.
-s	Does not list the names of identical files.
-w *num*	Change the width of the output to *num* number of characters.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Note: Differences in directory contents are not considered errors.

Examples

1. To summarize the differences between the files in two directories, type the following:

   ```
   dircmp proj.ver1 proj.ver2
   ```

   This displays a summary of the differences between the directories `proj.ver1` and `proj.ver2`. The summary lists separately the files found only in one directory or the other, and those found in both. If a file is found in both directories, the **dircmp** command notes whether the two copies are identical.
2. To show the details of the differences between files, type the following:

   ```
   dircmp  -d  -s proj.ver1 proj.ver2
   ```

   The **-s** flag suppresses information about identical files. The **-d** flag displays a **diff** listing for each of the differing files found in both directories.
3. To show the details of the differences between files with the width of the output line set to 90 characters, type the following:

   ```
   $dircmp -w 90 dir1 dir2
   ```

Files

Item	Description
/usr/bin/dircmp	Contains the **dircmp** command.

Related Information

The **cmp** command, **diff** command.

Directories in *Operating system and device management* describes the structure and characteristics of directories in the file system.

Input and output redirection in *Operating system and device management* describes how the operating system processes input and output.

dirname Command

Purpose

Writes to standard output all but the last part of a specified path.

Syntax

dirname *Path*

Description

The **dirname** command reads the specified path name, deletes all but the last / (slash) and the characters following it, and writes the result to standard output. If no characters follow the last /, the **dirname**

command uses the next to last / and ignores all characters following it. The **dirname** command applies the following rules in creating the path name:

1. If the *Path* parameter is a // (double slash), or if the *Path* parameter consists entirely of slash characters, change the string to a single / (slash). Skip steps 2 through 7.
2. Remove any trailing / characters from the specified path.
3. If there are no / characters remaining in the *Path* parameter, change the path to a single **.** (period). Skip steps 4 through 7.
4. Remove any trailing, non-slash characters from the path.
5. If the remaining path is // (double slash), go to step 6.
6. Remove any trailing slash characters from the path.
7. If the remaining path is empty, change the path to a single /.

For example, entering:

```
dirname //
```

results in a single / (slash). Entering:

```
dirname /a/b/
```

results in /a. Entering:

```
dirname a
```

results in a single . (period). Entering:

```
dirname a/b
```

results in the path name a.

The **dirname** and **basename** commands are generally used inside command substitutions within a shell procedure to specify an output file name that is some variation of a specified input file name.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion
>0	An error occurred.

Examples

To construct the name of a file located in the same directory as another, enter:

```
AOUTFILE=`dirname $TEXTFILE`/a.out
```

This sets the shell variable AOUTFILE to the name of an **a.out** file that is in the same directory as TEXTFILE. If TEXTFILE is **/home/fran/prog.c**, the value of `dirname $TEXTFILE` is **/home/fran** and AOUTFILE becomes **/home/fran/a.out**.

Files

Item	Description
/usr/bin/dirname	Contains the **dirname** command.

disable Command

The **disable** command includes information for the AIX Print Subsystem **disable** and the **System V Print Subsystem disable**.

Purpose

Disables printer queue devices.

Syntax

disable [**-c**] [**-r***Reason*] *PrinterName ...*

Description

The **disable** command disables or brings offline the printer queue devices specified by the *PrinterName* parameter.

Note: You must have root user authority or belong to the printq group to run this command.

Flags

Item	Description
-c	Cancels all job requests. Using this flag is the same as entering the **enq -K** command.
-r*Reason*	Specifies the reason for disabling the printer queue device with the *Reason* variable. This flag is a "no operation" flag, which means that the system ignores this flag.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To bring printer queue `lp0` offline without waiting for the current print jobs to finish, type:

   ```
   disable  -c lp0
   ```

2. To bring printer queue `lp0` offline after all print jobs are finished, type:

   ```
   disable lp0
   ```

Files

Item	Description
/usr/sbin/qdaemon	Queuing daemon
/etc/qconfig	Queue configuration file
/etc/qconfig.bin	Digested, binary version of the **/etc/qconfig** file
/var/spool/lpd/qdir/*	Queue requests
/var/spool/lpd/stat/*	Information on the status of the devices
/var/spool/qdaemon/*	Temporary copies of enqueued files

System V Print Subsystem disable Command

Purpose

Disable LP printers

Syntax

disable [*flags*] *printers*

Description

The **disable** command deactivates the named *printers*, disabling them from printing requests submitted by **lp**. By default, any requests that are currently printing on the designated printers will be reprinted in their entirety either on the same printer or on another member of the same class of printers. If the printer is remote, this command will only stop the transmission of jobs to the remote system. The **disable** command must be run on the remote system to disable the printer. (Run **lpstat -p** to get the status of printers.)

Printer names are *system-defined words* and as such should be restricted to uppercase and lowercase ASCII characters.

If you enter `disable -?`, the system displays the command usage message and returns 0.

Flags

-c
Cancel any requests that are currently printing on any of the designated printers. This flag cannot be used with the **-W** flag. If the printer is remote, the **-c** flag is ignored.

-r *reason*
Assign a *reason* for the disabling of the printers. This *reason* applies to all *printers* specified. This *reason* is reported by **lpstat -p**. *reason* must be enclosed in quotes if it contains blanks. The default reason is *unknown reason* for existing printers, and *new printer* for printers just added to the system but not yet enabled.

-W
Wait until the request currently being printed is finished before disabling the specified printer.

This flag cannot be used with the **-c** flag. If the printer is remote, the **-W** flag will be silently ignored.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Files

/var/spool/lp/*

diskusg Command

Purpose

Generates disk accounting data by user ID.

Syntax

diskusg [**-X**] [**-U** *MaxUsers*] [**-i** *FileListName*] [**-p** *File*] [**-u** *File*] [**-v**] { **-s** [*File ...*] | *FileSystem ...* }

Description

The **diskusg** command generates intermediate disk-accounting information from data in the files specified with the *File* or *FileSystem* parameters or from standard input. The **diskusg** command writes

one record per user to standard output. This command is called by the **dodisk** command, which can be run under the **cron** daemon. The output is in the following format:

Item	Description
UID	Contains the numerical user ID of the user.
Login	Contains the login name of the user.
Blocks	Contains the total number of 512-byte disk blocks allocated to the user.

The output of this command becomes the input of the **acctdisk** command, which converts the information to a total accounting record. The total accounting record is merged with other total accounting records to produce the daily report.

If you specify the *FileSystem* parameter, the **diskusg** command reads the i-nodes of the specified file systems to generate the usage data. The *FileSystem* parameters must be the special file names of the file system devices. For example, use the **/dev/hd4** device instead of / (root) directory to generate usage data for the root file system.

If you specify the *File* parameter, the input must be in a **diskusg** output format.

For more information on disk usage, see the **acctdusg** command.

Note: This command is for local devices only.

Flags

Item	Description
-i *FileListName*	Ignores the data in the *FileListName* file system. The *FileListName* variable specifies a list of file system names separated by commas or enclosed within quotation marks.
-p *File*	Uses the password file specified by the *File* variable to generate login names. The default is the **/etc/passwd** file.
-s [*File*]	Combines all records from the input file(s) or from standard input into a single record. The input data is already in a **diskusg** output format.
-U *MaxUsers*	Sets the maximum number of users that can be processed by the **diskusg** command. You need to use this flag only if the number of users is greater than the default of 5000.
-u *File*	Writes a record to the specified *File* variable for each file that is charged to a user ID of no one. Each record consists of the special file name, the i-node number, and the user ID.
-v	Writes a list of all files that are charged to no one to the standard error output.
-X	Prints and processes all available characters for each user name instead of truncating to the first 8 characters.

Security

Access Control: This command should grant execute (x) access only to members of the **adm** group.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To generate daily disk-accounting information, add a line similar to the following to the **/var/spool/cron/crontab/root** file:

```
0 2 * * 4 /usr/sbin/acct/dodisk
```

This command tells the **cron** daemon to run the **dodisk** command at 2 a.m. (02) each Thursday (4). The **dodisk** command calls both the **diskusg** and **acctdisk** commands.

Note: To perform this example, you must have root authority.

Files

Item	Description
/usr/sbin/acct/diskusg	Contains the **diskusg** command.
/etc/passwd	Contains the basic attributes of users.

dispgid Command

Purpose

Displays a list of all valid group names.

Syntax

dispgid

Description

The **dispgid** command can be used to display a list of all group names on the system (one name per line). There are no parameters for this command. The following files are accessed in read-only mode to retrieve the data:

- **/etc/passwd**
- **/etc/group**
- **/etc/security/user**
- **/etc/security/limits**
- **/etc/security/group**
- **/etc/security/environ**

Exit Status

0
The command completed successfully.

>0
An error occurred.

Examples

1. To list all the valid groups in the machine enter the **dispgid** command as follows:

   ```
   dispgid
   ```

   The output looks similar to the following:

```
system
staff
bin
sys
adm
uucp
mail
security
cron
printq
audit
ecs
nobody
usr
perf
```

Files

Item	Description
/usr/sbin/dispgid	Contains the **dispgid** command
/etc/group	Contains group information

dispuid Command

Purpose

Displays a list of all valid user names.

Syntax

dispuid

Description

This command can be used to display a list of all user names on the system (one line per name). There are no parameters for this command. The following files are accessed in read-only mode to retrieve the user data:

- **/etc/passwd**
- **/etc/security/user**
- **/etc/security/user.roles**
- **/etc/security/limits**
- **/etc/security/environ**
- **/etc/group**
- **/etc/group**

Exit Status

0
The command completed successfully.

>0
An error occurred.

Examples

1. To list all the valid users in your machine enter the dispuid command as follows:

```
dispuid
```

The output looks similar to the following:

```
root
daemon
bin
sys
adm
uucp
guest
nobody
lpd
invscout
imnadm
user1
```

Files

Item	Description
/usr/sbin/dispuid	Contains the dispuid command.
/etc/passwd	Contains password information.

dist Command

Purpose

Redistributes a message to additional addresses.

Syntax

dist [**+** Folder] [**-nodraftfolder** | **-draftfolder +***Folder*] [*Message* | **-draftmessage** *Message*] [**-annotate** [**-inplace** | **-noinplace**] | **-noannotate**] [**-form** *FormFile*] [**-editor** *Editor* | **-noedit**] [**-nowhatnowproc** | **-whatnowproc** *Program*]

Description

The **dist** command provides an interface for redistributing existing messages to a new list of addresses. By default, the **dist** command copies the current message in the current folder to the *UserMHDirectory***/draft** file and starts an editor. To specify a message in the current folder other than the default, use the *Message* parameter.

Once started, the editor prompts you to enter values for each header field. The **dist** command uses the header format defined in the *UserMHDirectory***/distcomps** file. (If this file does not exist, the system uses the **/etc/mh/distcomps** file.) Since the body of the message is the message you are redistributing, do not fill in the body. To define a format file other than *UserMHDirectory***/distcomps** file, use the **-form** flag.

To change the default editor, use the **-editor** flag or define the `Editor:` entry in your **$HOME/.mh_profile** file.

Press the Ctrl-D key sequence to exit the editor. Upon exiting the editor, the **dist** command starts the Message Handler (MH) `What Now?` prompt. Press the Enter key to see a list of the available **whatnow** subcommands. These subcommands enable you to continue editing the message header, list the message header, direct the disposition of the message, or end the processing of the **dist** command.

Note: A line of dashes or a blank line must be left between the header and the body of the message for the message to be identified when it is sent.

Redistributed messages consist of the original header and body appended to a new header. The **draft** file you edit using the **dist** command consists of header fields only. A copy of the original message with the new draft message is not automatically stored.

To annotate the original message with redistribution information, use the **-annotate** flag. This flag appends the original message with the `Resent:` field, and the current date and time.

Flags

Item	Description
-annotate	Annotates the message being redistributed with the lines: `Resent: date` `Resent: address` Since the **-annotate** flag is not preserved over multiple executions of the command, annotation is completed only if the message is sent directly from the **dist** command. The **-inplace** flag forces annotation to be done in place in order to preserve links to the annotated message.
-draftfolder *+Folder*	Places the draft message in the specified folder. If **-draftfolder** **+***Folder* flag is followed by a *Message* variable, it is the same as using the **-draftmessage** flag. If *+Folder* is not specified, the draft message is placed in *Current-Folder*.
-draftmessage *Message*	Specifies a draft message. By default, the system creates a new draft message in the current folder. The draft message becomes the current message.
-editor *Editor*	Specifies the initial editor for preparing the message for distribution.
+*Folder*	Identifies the folder that contains the message to redistribute. If a folder is not specified, then *Current-Folder* is assumed.
-form *FormFile*	Determines the message form. The **dist** command treats each line in the specified form file.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-inplace	Forces annotation to be done in place in order to preserve links to the annotated message.
Message	Identifies the message to redistribute. Use the following references to specify messages: ***Number*** Number of the message. **cur or . (period)** Current message. This is the default. **first** First message in a folder. **last** Last message in a folder. **next** Message following the current message. **prev** Message preceding the current message.

Item	Description
-noannotate	Suppresses annotation. This flag is the default.
-nodraftfolder	Places the draft in the *UserMHDirectory***/draft** file.
-noedit	Suppresses the initial edit.
-noinplace	Prevents annotation in place. This flag is the default.
-nowhatnowproc	Suppresses interactive processing of the **dist** command. The **-nowhatnowproc** flag prevents any edit from occurring.
-whatnowproc *Program*	Starts the specified program to guide you through the distribution tasks. If you specify the **whatnow** command as the *Program* variable, the **dist** command starts an internal **whatnow** procedure instead of a program with the file name **whatnow**.

Profile Entries

The following entries are entered in the *UserMHDirectory***/.mh_profile** file:

Item	Description
`Current-Folder:`	Sets the default current folder.
`Draft-Folder:`	Sets the default folder for drafts.
`Editor:`	Sets the default editor.
`fileproc:`	Specifies the program used to refile messages.
`Path:`	Specifies the user's MH directory.
`whatnowproc:`	Specifies the program used to prompt `What now?` questions.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To redistribute the current message from the current folder, enter:

   ```
   dist
   ```

   The system prompts you for the header field values. After entering a value, press the Enter key. To skip an entry, press the Enter key without entering a value. You must fill in the `Resent-to:` field. After completing the headers, do not modify the body of the text. Press the Ctrl-D key sequence to exit the editor. The system prompts you with:

   ```
   What now?
   ```

   Press the Enter key to see a list of available options. If you want to redistribute this message, enter `send.` Your message is redistributed to the new list of addresses.
2. To redistribute a message to a new list of addresses when a message draft exists, enter:

   ```
   dist
   ```

   The system responds with a message similar to the following:

```
Draft "$HOME/Mail/draft" exists (43 bytes).
Disposition? _
```

To redistribute this draft, enter:

```
replace
```

The system prompts you for the header field values. After entering a value, press the Enter key. To skip an entry, press the Enter key without entering a value. You must fill in the `Resent-to:` field. After completing the headers, do not modify the body of the text. Press the Ctrl-D key sequence to exit the editor. The system prompts you with:

```
What now?
```

Press the Enter key to see a list of available options. If you want to redistribute the draft, enter `send`. Your message is redistributed to the new list of addresses.

3. To redistribute message 15 from the `schedules` folder, enter:

   ```
   dist  +schedules 15
   ```

   The system prompts you for the header field values. After entering a value, press the Enter key. To skip an entry, press the Enter key without entering a value. You must fill in the `Resent-to:` field. After completing the headers, do not modify the body of the text. Press the Ctrl-D key sequence to exit the editor. The system prompts you with:

   ```
   What now?
   ```

   Press the Enter key to see a list of available options. To redistribute the message, type `send` and press the Enter key.

Files

Item	Description
/etc/mh/distcomps	Contains the system default message format.
*UserMHDirectory***/distcomps**	Contains the user's default message format.
*UserMHDirectory***/draft**	Contains the current draft file.
/usr/bin/dist	Contains the executable form of the **dist** command.

dmpuncompress Command

Purpose

Restores dump compressed files.

Syntax

`/usr/bin/dmpuncompress` [`-f`] [`-p`] [*File*]

Description

The `dmpuncompress` command restores original dump files that were compressed at dump time.

Each compressed file specified by the *File* parameter is removed and replaced by an expanded copy. The expanded file has the same name as the compressed version, but without the `.BZ` extension. If the user has root authority, the expanded file retains the same owner, group, modes, and modification time as the

original file. If the user does not have root authority, the file retains the same modes and modification time, but acquires a new owner and group.

Flags

Item	Description
`-f` *File*	Forces expansion. Overwrites the file if it already exists. The system does not prompt the user that an existing file will be overwritten. File size might not actually shrink.
`-p` *File*	Preserves original `.BZ` file and uncompressed dump file. This overrides removal of the compressed file when there is a successful restoration of the original dump file. If restoration of the original dump file is incomplete because of an error, this option disables removal of the partial dump.

Exit Status

Item	Description
`0`	Successful completion.
`>0`	An error occurred.

Example

1. To uncompress the `dump.BZ` file, enter:

   ```
   /usr/bin/dmpuncompress dump.BZ
   ```

   The `dump.BZ` file is uncompressed and renamed `dump`.
2. To keep the `dump.BZ` file and the newly created dump file in the file system following completion, enter:

   ```
   /usr/bin/dmpuncompress -p dump.BZ
   ```

Location

`/usr/bin/dmpuncompress`

dnssec-keygen Command

Purpose

Domain name system security extensions (DNSSEC) key generation tool.

Syntax

dnssec-keygen [**-a** *algorithm*] [**-b** *keysize*] [**-n** *nametype*] [**-c** *class*] [**-e**] [**-f** *flag*] [**-g** *generator*] [**-h**] [**-k**] [**-p** *protocol*] [**-r** *randomdev*] [**-s** *strength*] [**-t** *type*] [**-v** *level*] [*name*]

Description

The **dnssec-keygen** command generates keys for DNSSEC (Secure DNS). It can also generate keys to use with Transaction Signatures (TSIG).

Flags

Item	Description
-a *algorithm*	Selects the cryptographic algorithm. The algorithm can have one of the following values: • RSAMD5 • DSA • DH (Diffie-Hellman) • HMAC-MD5 These values are case-sensitive. **Notes:** 1. For DNSSEC, RSASHA1 is a mandatory-implement algorithm, and DSA is preferred. For TSIG, HMAC-MD5 is mandatory. 2. HMAC-MD5 and DH automatically set the **-k** flag.
-b *keysize*	Specifies the number of bits in the key. The choice of key size depends on the algorithm used. RSAMD5 and RSASHA1 keys must be 512 - 4096 bits. DH keys must be 128 - 4096 bits. DSA keys must be 512 - 1024 bits and an exact multiple of 64. HMAC-MD5 keys must be 1 - 512 bits.
-n *nametype*	Specifies the owner type of the key. The value of *nametype* must either be ZONE (for a DNSSEC zone key), HOST, or ENTITY (for a key that is associated with a host), USER (for a key that is associated with a user), or OTHER (DNSKEY). These values are not case-sensitive.
-c *class*	Indicates that the Domain Name Server (DNS) record that contains the key must have the specified class. If not specified, class IN is used.
-e	If you are generating an RSAMD5 or RSASHA1 key, use a large exponent.
-f *flag*	Sets the specified flag in the *flag* field of the KEY or the DNSKEY record. The only recognized flag is KSK (Key Signing Key) DNSKEY.
-g *generator*	If you are generating a DH key, use this generator. The acceptable values are 2 and 5. If generator is not specified, a known prime from RFC 2539 is used if possible; otherwise the default is 2.
-h	Prints a short summary of the options and arguments to the **dnssec-keygen** command.
-k	Generates KEY records rather than DNSKEY records.
-p *protocol*	Sets the protocol value for the generated key. The protocol is a number 0 - 255. The default is 3 (DNSSEC).
-r *randomdev*	Specifies the source of randomness. If the operating system does not provide a /dev/random file or equivalent device, the default source of randomness is keyboard input. The *randomdev* argument specifies the name of a character device or a file that contains random data to be used instead of the default. The special value keyboard indicates that keyboard input must be used.
-s *strength*	Specifies the strength value of the key. The *strength* argument is a number 0 - 15, and currently has no defined purpose in DNSSEC.

Item	Description
-t *type*	Indicates the use of the key. The *type* must be one of AUTHCONF, NOAUTHCONF, NOAUTH, or NOCONF. The default is AUTHCONF. AUTH refers to the ability to authenticate data, and CONF the ability to encrypt data. No key is generated for these algorithms (DH, HMAC-MD5, HMAC-SHA1, HMAC-SHA224, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512) with key type as NOAUTHCONF.
-v *level*	Sets the debugging level.

Parameters

Item	Description
name	The name of the key that is specified on the command line. For DNSSEC keys, this name must match the name of the zone for which the key is being generated.

Generated Keys

When the **dnssec-keygen** command completes successfully, it prints a string of the form Knnnn.+aaa+iiiii to the standard output. It is an identification string for the key that it generated.

- nnnn is the key name.
- aaa is the numeric representation of the algorithm.
- iiiii is the key identifier (or footprint).

The **dnssec-keygen** command creates two files with names based on the printed string: Knnnn.+aaa+iiiii.key contains the public key, and Knnnn.+aaa+iiiii.private contains the private key.

The .key file contains a DNSKEY record that can be inserted into a zone file (directly or with a $INCLUDE statement). The .private file contains algorithm-specific fields. For security reasons, this file does not have general read permission. Both the .key and .private files are generated for symmetric encryption algorithm such as HMAC-MD5, even though the public key and the private key are equivalent.

Examples

To generate a 768 - bit DSA key for the domain example.com, type the following command:

```
dnssec-keygen -a DSA -b 768 -n ZONE example.com
```

The command prints a string of the form:

```
Kexample.com.+003+26160
```

In this example, **dnssec-keygen** creates the files Kexample.com.+003+26160.key and Kexample.com.+003+26160.private.

dnssec-makekeyset command

Purpose

Domain name system security extensions (DNSSEC) zone signing tool.

Syntax

dnssec-makekeyset [**-a**] [**-s** *start-time*] [**-e** *end-time*] [**-h**] [**-p**] [**-r** *randomdev*] [**-t** *ttl*] [**-v** *level*] {*key...*}

Description

The **dnssec-makekeyset** command generates a key set from one or more keys that are created by the **dnssec-keygen** command. It creates a file that contains a KEY record for each key, and self-signs the key set with each zone key. The output file is of the form `keyset-nnnn.`, where *nnnn* is the zone name.

Flags

Item	Description
-a	Verifies all generated signatures.
-s *start-time*	Specifies the date and time when the generated SIG records become valid. It can be either an absolute or relative time. An absolute start time is indicated by a number in YYYYMMDDHHMMSS notation; 20000530144500 denotes 14:45:00 UTC on May 30th, 2000. A relative start time is indicated by +N, which is *N* seconds from the current time. If no *start-time* is specified, the current time is used.
-e *end-time*	Specifies the date and time when the generated SIG records expire. As with the *start-time* value, an absolute time is indicated in YYYYMMDDHHMMSS notation. A time relative to the start time is indicated with +N, which is *N* seconds from the start time. A time relative to the current time is indicated with now+N. If no *end-time* is specified, 30 days time from the start time is used as a default.
-h	Prints a short summary of the options and arguments to the **dnssec-makekeyset** command.
-p	Uses pseudo-random data when you sign the zone. It is faster, but less secure, than using real random data. This option might be useful when you sign large zones or when the entropy source is limited.
-r *randomdev*	Specifies the source of randomness. If the operating system does not provide a `/dev/random` or equivalent device, the default source of randomness is keyboard input. The *randomdev* value specifies the name of a character device or file that contains random data to be used instead of the default. The special value keyboard indicates that keyboard input must be used.
-t *ttl*	Specifies the TTL (time to live) of the KEY and SIG records. The default is 3600 seconds.
-v *level*	Sets the debugging level.

Parameters

Item	Description
key	The list of keys to be included in the key set file. These keys are expressed in the form `Knnnn.+aaa+iiiii` as generated by the **dnssec-keygen** command.

Examples

The following command generates a key set that contains the DSA key for `example.com` generated in the **dnssec-keygen** man page.

```
dnssec-makekeyset -t 86400 -s 20000701120000 -e +2592000 Kexample.com.+003+26160
```

In this example, the **dnssec-makekeyset** command creates the file `keyset-example.com.`. This file contains the specified key and a self-generated signature. The DNS administrator for `example.com` can send `keyset-example.com.` to the DNS administrator for `.com` for signing, if the `.com` zone is DNSSEC-aware and the administrators of the two zones have some mechanism for authenticating each other and exchanging the keys and signatures securely.

dnssec-signkey Command

Purpose

Domain name system security extensions (DNSSEC) key set signing tool.

Syntax

dnssec-signkey [**-a**] [**-c** *class*] [**-s** *start-time*] [**-e** *end-time*] [**-h**] [**-p**] [**-r** *randomdev*] [**-v** *level*] *keyset key*

Description

The **dnssec-signkey** command signs a key set. Typically the key set is for a child zone, and is generated by the **dnssec-makekeyset** command. The child zone's key set is signed with the zone keys for its parent zone. The output file is of the form `signedkey-`*nnnn*`.`, where *nnnn* is the zone name.

Flags

Item	Description
-a	Verify all generated signatures.
-c *class*	Specifies the DNS class of the key sets.
-s *start-time*	Specify the date and time when the generated SIG records become valid. It can be either an absolute or relative time. An absolute start time is indicated by a number in YYYYMMDDHHMMSS notation; 20000530144500 denotes 14:45:00 UTC on May 30th, 2000. A relative start time is indicated by *+N*, which is *N* seconds from the current time. If no *start-time* is specified, the current time is used.
-e *end-time*	Specify the date and time when the generated SIG records expire. As with start-time, an absolute time is indicated in YYYYMMDDHHMMSS notation. A time relative to the start time is indicated with *+N*, which is *N* seconds from the start time. A time relative to the current time is indicated with now+N. If no *end-time* is specified, 30 days time from the start time is used as a default.
-h	Prints a short summary of the options and arguments to the **dnssec-signkey** command.
-p	Use pseudo-random data when you sign the zone. It is faster, but less secure, than using real random data. This option might be useful when you sign large zones or when the entropy source is limited.
-r *randomdev*	Specifies the source of randomness. If the operating system does not provide a `/dev/random` or equivalent device, the default source of randomness is keyboard input. *randomdev* specifies the name of a character device or file that contains random data to be used instead of the default. The special value keyboard indicates that keyboard input must be used.
-v *level*	Sets the debugging level.

Parameters

Item	Description
keyset	The file that contains the child's key set.
key	The keys that are used to sign the child's key set.

Examples

The DNS administrator for a DNSSEC-aware `.com` zone uses the following command to sign the key set file for `example.com` created by the **dnssec-makekeyset** command with a key generated by the **dnssec-keygen** command:

```
dnssec-signkey keyset-example.com. Kcom.+003+51944
```

In this example, **dnssec-signkey** creates the file `signedkey-example.com.`, which contains the `example.com` keys and the signatures by the `.com` keys.

dnssec-signzone Command

Purpose

Domain name system security extensions (DNSSEC) zone signing tool.

Syntax

dnssec-signzone [**-a**] [**-c** *class*] [**-d** *directory*] [**-e** *end-time*] [**-f** *output-file*] [**-g**] [**-h**] [**-k** *key*] [**-l** *domain*] [**-i** *interval*] [**-I** *input-format*] [**-j** *jitter*] [**-N** *soa-serial-format*] [**-o** *origin*] [**-O** *output-format*] [**-p**] [**-r** *randomdev*] [**-s** *start-time*] [**-t**] [**-v** *level*] [**-z**] *zonefile* [*key*...]

Description

The **dnssec-signzone** command signs a zone. It generates NSEC and RRSIG records and produces a signed version of the zone. The presence or absence of a key set file for each child zone determines the security status of delegations from the signed zone (that is, whether the child zones are secure or not).

Flags

Item	Description
-a	Verifies all generated signatures.
-c *class*	Specifies the DNS class of the zone.
-d *directory*	Looks for key set files in the directory that is specified by the *directory* argument.
-k *key*	Treats the specified key as a key-signing key ignoring any key flags. You can specify this option multiple times.
-l *domain*	Generates a DLV set in addition to the key (DNSKEY) and DS sets. The domain is appended to the name of the records.
-g	Generates DS records for child zones from key set files. This flag removes existing DS records.
-s *start-time*	Specifies the date and time when the generated RRSIG records become valid. It can be either an absolute or relative time. An absolute start time is indicated by a number in YYYYMMDDHHMMSS notation; 20000530144500 denotes 14:45:00 UTC on May 30th, 2000. A relative start time is indicated by +*N*, which is *N* seconds from the current time. If you do not specify the *start-time* argument, the command uses the current time minus 1 hour (to allow for clock skew).
-e *end-time*	Specifies the date and time when the generated RRSIG records expire. As with the *start-time* argument, an absolute time is indicated in YYYYMMDDHHMMSS notation. A time relative to the start time is indicated with +*N*, which is *N* seconds from the start time. A time relative to the current time is indicated with now+N. If you do not specify the *end-time* argument, the command uses 30 days from the start time as a default.

Item	Description
-f *output-file*	Specifies the name of the output file that contains the signed zone. The default is to append `.signed` to the input file name.
-h	Prints a short summary of the options and arguments of the **dnssec-signzone** command.
-i *interval*	When a previously signed zone is passed as input, records might be resigned. The interval option specifies the cycle interval as an offset from the current time (in seconds). If an RRSIG record expires after the cycle interval, it is retained. Otherwise, it is considered to be expiring soon, and it is then replaced. The default cycle interval is one quarter of the difference between the signature end and start times. If you specify neither the *end-time* argument or the *start-time* argument, the **dnssec-signzone** command generates signatures that are valid for 30 days, with a cycle interval of 7.5 days. Therefore, if any existing RRSIG records are due to expire in less than 7.5 days, they are replaced.
-I *input-format*	Specifies the format of the input zone file. Possible formats are text (default) and raw.
-j *jitter*	When you sign a zone with a fixed signature lifetime, all RRSIG records issued at the time of signing expire simultaneously. If the zone is incrementally signed, for example, a previously signed zone is passed as input to the signer and all expired signatures must be regenerated at about the same time. The *jitter* argument specifies a jitter window that is used to randomize the signature expire time, thus spreading incremental signature regeneration over time. Signature lifetime jitter can also benefit validators and servers by spreading out cache expiration. For example, if large numbers of RRSIGs do not expire at the same time from all caches, there is less congestion than if all validators must refetch at mostly the same time.
-n *ncpus*	Specifies the number of threads to use. By default, the command starts one thread for each detected processor.
-N *soa-serial-format*	Specifies the SOA serial number format of the signed zone. The *soa-serial-format* argument can be one of the following values: **keep** Does not modify the SOA serial number. It is the default value. **increment** Increases the SOA serial number by using RFC 1982 arithmetic. **unixtime** Sets the SOA serial number to the number of seconds since epoch.
-o *origin*	Specifies the zone origin. If not specified, the name of the zone file is assumed to be the origin.
-O *output-format*	Specifies the format of the output file that contains the signed zone. Possible formats are text (default) and raw.
-p	Uses pseudo-random data when you sign the zone. It is faster, but less secure, than using real random data. This option can be useful when you sign large zones or when the entropy source is limited.
-r *randomdev*	Specifies the source of randomness. If the operating system does not provide a `/dev/random` file or equivalent device, the default source of randomness is keyboard input. The *randomdev* argument specifies the name of a character device or file that contains random data to be used instead of the default. The special value keyboard indicates that keyboard input must be used.
-t	Prints statistics at completion.
-v *level*	Sets the debugging level.
-z	Ignores KSK flag on key when you determine what to sign.

Parameters

Item	Description
zonefile	The file that contains the zone to be signed.
key	The keys that are used to sign the key set. If no keys are specified, the defaults are all zone keys that have private key files in the current directory.

Examples

The following command signs the `example.com` zone with the DSA key generated by the **dnssec-keygen** command. The zone's keys must be in the zone. If there are key set files that are associated with this zone or any child zones, they must be in the current directory, `example.com`. You can issue the following command:

```
dnssec-signzone -o example.com db.example.com Kexample.com.+003+26160
```

In this example, the **dnssec-signzone** command creates the `db.example.com.signed` file. This file must be referenced in a zone statement in a `named.conf` file.

dodisk Command

Purpose

Initiates disk-usage accounting.

Syntax

/usr/sbin/acct/dodisk [**-X**] [**-o**] [*File ...*]

Description

The **dodisk** command initiates disk-usage accounting by calling the **diskusg** command and the **acctdisk** command. If you specify the **-o** flag with the **dodisk** command, a more thorough but slower version of disk accounting by login directory is initiated using the **acctdusg** command. Normally, the **cron** daemon runs the **dodisk** command.

By default, the **dodisk** command does disk accounting only on designated files with stanzas in the **/etc/filesystems** file and that contain the attribute **account=true**. If you specify file names with the *File* parameter, disk accounting is done on only those files.

If you do not specify the **-o** flag, the *File* parameter should contain the special file names of mountable file systems. If you specify both the **-o** flag and the *File* parameter, the files should be mount points of mounted file systems.

Note: You should not share accounting files among nodes in a distributed environment. Each node should have its own copy of the various accounting files.

Flags

Item	Description
-o	Calls the **acctdusg** command, instead of the **diskusg** command, to initiate disk accounting by login directory.
-X	Process all available characters of each user name instead of truncating to the first 8 characters.

Security

Access Control: This command should grant execute (x) access only to members of the **adm** group.

Examples

1. To start automatic disk-usage accounting, add the following to the **/var/spool/cron/crontabs/root** file:

   ```
   0 2 * * 4 /usr/sbin/acct/dodisk
   ```

   This example shows the instructions that the **cron** daemon will read and act upon. The **dodisk** command will run at 2 a.m. (`0 2`) each Thursday (4). This command is only one of the accounting instructions normally given to the **cron** daemon. See "Setting Up an Accounting System" in *Operating system and device management* for more information on typical **cron** accounting entries.
2. To run disk-usage accounting on a system that contains user names greater than 8 character, add the following line to the **/var/spool/cron/crontabs/root** file:

   ```
   0 2 * * 4 /usr/sbin/acct/dodisk -X
   ```

Files

Item	Description
/usr/sbin/acct	The path to the accounting commands
/etc/filesystems	Contains information about file system.

domainname Command

Purpose

Displays or sets the name of the current Network Information Service (NIS) domain.

Syntax

/usr/bin/domainname [*DomainName*]

Description

The **domainname** command displays or sets the name of the current NIS domain. If you do not specify a parameter, the **domainname** command displays the name of the current NIS domain. A domain typically encompasses a group of hosts under the same administration.

Only the root user can set the name of the domain by giving the **domainname** command an argument.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To join a new domain, enter:

   ```
   domainname caesar
   ```

In this example, the `domainname` command sets the NIS domain name to caesar.

2. To find out the name of the domain your machine belongs to, enter:

```
domainname
```

domlist Command

Purpose

Displays domain information for a user or process.

Syntax

domlist -p *pid*

Description

The **domlist** command provides domain information to the invoker about their current assigned domains. If no flags or arguments are specified, the **domlist** command displays the list of domains assigned to the invoker with the text description of each domain if one is provided in the domains database.

The **domlist** command also allows a privileged user to list the domain information for a process. Specifying a process ID with the **-p** flag allows a privileged user to display the domains associated with a process.

Flags

Item	Description
-p *PID*	Displays domain information of the specified process.

Examples

1. To display the list of domains that assigned to you and their text descriptions, use the following command:

```
domlist
```

2. To display the list of domains assigned to a process use the following command:

```
domlist -p <pid>
```

Files Accessed

Item	Description
Files	**Mode**
/etc/security/domains	r

dosdel Command

Purpose

Deletes DOS files.

Syntax

dosdel [**-v**] [**-D** *Device*] *File ...*

Description

The **dosdel** command deletes the DOS file specified by the *File* parameter. Use the **-v** flag to obtain format information about the disk.

DOS file-naming conventions are used with one exception. Since the \ (backslash) character can have special meaning to the operating system, use a / (slash) character as the delimiter to specify subdirectory names in a DOS path name. The **dosdel** command converts lowercase characters in the file or directory name to uppercase before it checks the disk. Because all file names are assumed to be full (not relative) path names, you need not add the initial / (slash).

Flags

Item	Description
-D*Device*	Specifies the name of the DOS device as **/dev/fd0** or **/dev/fd1**. The default device is **/dev/fd0.**
-v	Writes information to standard output about the format of the disk. Use this flag to verify that a device is a DOS disk.

Examples

To delete a DOS file on the default device, enter:

```
dosdel file.ext
```

Files

Item	Description
/usr/bin/dosdel	Contains the **dosdel** command.

dosdir Command

Purpose

Lists the directory for DOS files.

Syntax

dosdir [**-l** [**-e**]] [**-a**] [**-d**] [**-t**] [**-v**] [**-D** *Device*] [*File ...* | *Directory ...*]

Description

The **dosdir** command displays information about the specified DOS files or directories. If you specify a directory without also specifying the **-d** flag, the **dosdir** command displays information about the files in that directory.

DOS file-naming conventions are used with one exception. Since the \ (backslash) character can have special meaning to the operating system, use a / (slash) character as the delimiter to specify subdirectory names in a DOS path name. The **dosdir** command converts lowercase characters in the file or directory name to uppercase before it checks the disk. Because all file names are assumed to be full (not relative) path names, you need not add the initial / (slash).

Flags

Item	Description
-a	Writes information about all files. This includes hidden and system files as well as the . (dot) and .. (dot-dot) files.
-d	Treats the *File* value as a file, even if a directory is specified. When a directory is specified with the *Directory* parameter, information about the directory itself is listed instead of information about the files it contains.
-D*Device*	Specifies the name of the DOS device as **/dev/fd0** or **/dev/fd1**. The default device is **/dev/fd0**.
-e	Uses the **-l** flag to write the list of clusters allocated to the file.
-l	Produces a list of clusters that includes the creation date, size in bytes, and attributes of the file. The size of a subdirectory is specified as 0 bytes. The attributes have the following meanings: **A (Archive)** The file has not been backed up since it was last modified. **D (Directory)** The file is a subdirectory and not included in the normal DOS directory search. **H (Hidden)** The file is not included in the normal DOS directory search. **R (Read-only)** The file cannot be modified. **S (System)** The file is a system file and not included in the normal DOS directory search.
-t	Lists the entire directory tree starting at the named directory.
-v	Writes information to standard output about the format of the disk. Use this flag to verify that a device is a DOS disk.

Examples

To read a directory of the DOS files on **/dev/fd0**, enter:

```
dosdir
```

The command returns the names of the files and disk-space information.

```
PG3-25.TXT
PG4-25.TXT
PG5-25.TXT
PG6-25.TXT
Free space: 312320 bytes
```

To read a directory of the DOS files on **/dev/fd1**, enter:

```
dosdir -D/dev/fd1
```

The command returns the names of the files and disk-space information.

```
PG7-25.TXT
PG8-25.TXT
PG9-25.TXT
PG10-25.TXT
Free space: 312320 bytes
```

Files

Item	Description
/usr/bin/dosdir	Contains the **dosdir** command.

dosformat Command

Purpose

Formats a DOS diskette.

Syntax

dosformat [**-V** *Label*] [**-D** *Device* | **-4**]

Description

The **dosformat** command formats a diskette with the DOS format.

The default device and DOS diskette drive format is **/dev/fd0** for a 3.5-inch diskette. The density is usually either 1.44M-byte or 2.88M-byte, depending on the density that the drive supports. Other DOS diskette drive formats are implemented by using the **-D** or **-4** flags.

To include a volume label, use the **-V** flag.

Note: The purpose of this command is to facilitate file transfer between this operating system and DOS systems. Using this command to format a diskette that needs to have the DOS system startup files on it is not recommended.

Flags

Item	Description
-V	Write the *Label* parameter to the diskette as the DOS volume label.

Item	Description
-D*Device*	Specifies the diskette drive type and size. The *Device* parameter can be specified as: For a 3.5-inch, 1.44M drive: **/dev/fd0** 1.44MB (default) **/dev/fd0h** 1.44MB **/dev/fd0l** 720KB **/dev/fd0.18** 1.44MB **/dev/fd0.9** 720KB For a 3.5-inch, 2.88M drive: **/dev/fd0** 2.88MB (default) **/dev/fd0h** 2.88MB **/dev/fdol** 720KB **/dev/fd0.36** 2.88MB **/dev/fd0.18** 1.44MB **/dev/fd0.9** 720KB For a 5.25-inch, 1.2M drive: **/dev/fd0** 1.2MB (default) **/dev/fd0.15** 1.2MB **/dev/fd0.9** 360KB

Item	Description
-4	Specifies the lower density for the diskette size.

Examples

1. To format a 3.5-inch, 1.44M-byte diskette with the volume label "homework," type the following:

   ```
   dosformat -V homework
   ```

2. To format a 5.25-inch, 360K-byte diskette, type the following:

   ```
   dosformat -D /dev/fd1.9
   ```

   OR

```
dosformat -D /dev/fd1 -4
```

Files

Item	Description
/usr/bin/dosformat	Contains the **dosformat** command.

dosread Command

Purpose

Copies DOS files.

Syntax

dosread [**-a**] [**-v**] [**-D** *Device*] *File1* [*File2*]

Description

The **dosread** command copies the DOS file specified by the *File1* variable to standard output or to the file specified by the *File2* variable. If no pathname is specified for the *File2* variable, the DOS file is copied to the root directory.

Unless otherwise specified, the **dosread** command copies the number of bytes specified in the directory entry for the file specified by the *File1* variable. This means, in particular, that you cannot copy directories because, by convention, directories have a record size of 0.

You can use DOS file-naming conventions with one exception: the \ (backslash). Because the \ character can have special meaning in DOS, use a / (slash) character as the delimiter to specify subdirectory names in a DOS path name. The **dosdir** command converts lowercase characters in the file or directory name to uppercase before it checks the disk. Because all file names are assumed to be full (not relative) path names, you need not add the initial / (slash).

Notes:

1. The **dosread** command does not interpret the * and ? (asterisk and question mark) wildcard characters as having special meaning. If you do not specify a file-name extension, the file name is matched as if you had specified a blank extension.
2. You cannot customize the name of this command. The command must be named **dosread**.
3. The **dosread** command reads files from the default drive containing the DOS diskette. The **dosread** command then copies the files to the current directory as a file recognized by this operating system. If the DOS diskette contains subdirectories, the **dosread** command does not create corresponding new subdirectories in this operating system. You must create the subdirectory and specify each DOS file you want to copy into the new subdirectory.

Flags

Item	Description
-a	Replaces each CR-LF (carriage return, line-feed) key sequence with a new-line character and interprets a Ctrl-Z (ASCII SUB) key sequence as the end-of-line character.
-D*Device*	Specifies the name of the DOS device as **/dev/fd0** or **/dev/fd1**. The default value of the *Device* variable is **/dev/fd0**. This device must have the DOS disk format.
-v	Writes file information to standard output about the format of the disk. Use this flag to verify that a device is a DOS disk.

Examples

1. To copy a text file from a DOS, type:

   ```
   dosread  -a chap1.doc chap1
   ```

   This command sequence copies the DOS text file `\CHAP1.DOC` on default device **/dev/fd0** to `chap1` in the current directory.
2. To copy a binary file from a DOS diskette, type:

   ```
   dosread -D/dev/fd1 /survey/test.dta /home/fran/testdata
   ```

   This command sequence copies the DOS data file `\SURVEY\TEST.DTA` on **/dev/fd1** to `/home/fran/testdata`.
3. To copy every DOS file on a diskette, type:

   ```
   dosdir | awk '!/There are/ {print $1}'|xargs -t -i dosread {} {}
   ```

   This command sequence takes files from the default drive containing the DOS disk and copies them to the current directory.

Files

Item	Description
/usr/bin/dosread	Contains the **dosread** command.
/dev/fd0	Contains the device name for a diskette drive.

doswrite Command

Purpose

Copies files to DOS files.

Syntax

doswrite [**-a**] [**-v**] [**-D***Device*] *File1 File2*

Description

The **doswrite** command copies the file specified by the *File1* parameter to the DOS file specified by the *File2* parameter. The **doswrite** command copies files to a single DOS diskette. The **doswrite** command cannot copy files across multiple DOS diskettes.

The **doswrite** command writes the file specified by the *File2* parameter to the DOS device using standard DOS naming conventions. Because the DOS \ (backslash) character can have a special meaning for the DOS operating system, do not use a \ (backslash) when specifying subdirectory names in the *File2* parameter. Use the / (slash) character instead.

The **doswrite** command converts lowercase characters specified in the *File1* parameter to uppercase before it checks the DOS device. Because all file names are assumed to be full (not relative) path names, you do not need to add the initial / (slash).

If the file specified in the *File2* parameter contains a / (slash), each intervening component must exist as a directory and the last component (the named file) must not exist. Any existing file with the same name is overwritten.

Notes:

1. The wildcard characters * and ? (asterisk and question mark) are not treated in a special way by this command (although they are by the shell). If you do not specify a file-name extension, the file name is matched as if you had specified a blank extension.
2. This command must be named **doswrite**.
3. A DOS directory holds up to 244 files.

Flags

Item	Description
-a	Replaces NL (new-line) characters with the CR-LF (carriage return, line-feed) sequence. Ctrl-Z is added to the output at the end of file.
-D *Device*	Specifies the name of the DOS device as **/dev/fd0** or **/dev/fd1**. The default device is **/dev/fd0**. This device must have the DOS disk format.
-v	Writes information to standard output about the format of the disk. Use this flag to verify that a device is a DOS disk.

Examples

1. To copy a text file to a DOS diskette, enter:

   ```
   doswrite  -a chap1 chap1.doc
   ```

   This copies the file `chap1` in the current directory to the DOS text file `\CHAP1.DOC` on default device **/dev/fd0**.
2. To copy a binary file to a DOS diskette, enter:

   ```
   doswrite -D/dev/fd1 /home/fran/testdata /survey/test.dta
   ```

   This copies the data file `/home/fran/testdata` to the DOS file `\SURVEY\TEST.DTA` on **/dev/fd1**.
3. To copy every file in the current directory to a DOS diskette in your default drive, enter:

   ```
   for i in *
   do
   doswrite $i $i
   done
   ```

Files

Item	Description
/usr/bin/doswrite	Contains the **doswrite** command.
/dev/fd0	Contains the device name for diskette drive.

dp Command

Purpose

Parses and reformats dates.

Syntax

dp [**-form** *File* | **-format** *String*] [**-width** *Number*] *Date*

Description

The **dp** command parses and reformats dates. The **dp** command is not started by the user. The **dp** command is called by other programs, typically by its full path name, **/usr/lib/mh/dp**.

The **dp** command parses each mail header string specified as a date and attempts to reformat the string. The default output format for the **dp** command is the ARPA RFC 822 standard. For each string it is unable to parse, the **dp** command displays an error message.

Parameter

Item	Description
Date	Specifies the date to be parsed.

Flags

Item	Description
-form *File*	Reformats the date specified in the *Date* parameter to the alternate format described by the *File* variable.
-format *String*	Reformats the date specified in the *Date* parameter to the alternate format specified by the *String* variable. The default format string follows: `%<(nodate{text})error:%{text}%|%(putstr(pretty{text}))%>`
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.
-width *Number*	Sets the maximum number of columns the **dp** command uses to display dates and error messages. The default is the width of the display.

Files

Item	Description
$HOME/.mh_profile	Contains the MH user profile.
/etc/mh/mtstailor	Contains MH command definitions.

dpid2 Daemon

Purpose

Starts the **dpid2** Distributed Protocol Interface - SNMP multiplexer protocol (DPI-SMUX) converter daemon as a background process.

Syntax

dpid2 [**-d** [*Level*]]

Description

The **dpid2** command starts the **dpid2** DPI-SMUX converter daemon. This command can be issued only by a user with root privileges or by a member of the system group.

The **dpid2** DPI-SMUX converter daemon complies with the standard Simple Network Management Protocol (SNMP) DPI version 2.0 that is defined by RFC 1592 and SNMP SMUX protocol and Management Information Base (MIB) defined by RFC 1227.

The **dpid2** daemon acts as a DPI 2.0 to SMUX converter. It is used to allow DPI subagents, such as `/usr/sbin/hostmibd`, to communicate with the AIX SNMP version 1 agent. The converter changes DPI2 messages into SMUX protocol messages and vice versa. The **dpid2** daemon itself is implemented as SMUX peer. It connects with the TCP port 199 of the SMUX server that is part of the `snmpd` agent. To a DPI2 subagent (for example, `/usr/sbin/hostmibd`), the **dpid2** daemon behaves like a DPI2 agent. It listens on an arbitrary TCP port for a connection request from a DPI2 subagent. This port number is registered by the **dpid2** daemon with the `snmpd` agent through MIB variable dpiPortForTCP (1.3.6.1.4.1.2.2.1.1.1). The DPI2 subagent learns this port number from the `snmpd` agent by sending a get-request query for the dpiPortForTCP.0 (1.3.6.1.4.1.2.2.1.1.1.0) instance to the `snmpd` agent. After the DPI2 subagent knows the TCP port number, which the DPI2 agent is listening on, it then tries to connect to it.

The **dpid2** daemon is normally run during system startup when the `/etc/rc.tcpip` shell script is called.

The **dpid2** daemon must be controlled by using the System Resource Controller (SRC). Entering `dpid2` at the command line is not recommended.

Use the following SRC commands to manipulate the **dpid2** daemon:

startsrc
Starts a subsystem, group of subsystems, or a subserver.

stopsrc
Stops a subsystem, group of subsystems, or a subserver.

refresh
Causes a subsystem or group of subsystems to reread the appropriate configuration file.

lssrc
Gets the status of a subsystem, group of subsystems, or a subserver.

Note: The `snmpdv3` agent itself acts as a DPI2 agent and listens on the dpiPortForTCP.0 TCP port. Therefore, the **dpid2** daemon is not needed when you use the `snmpdv3` agent. Therefore, the **dpid2** daemon is not run in the system startup and the **dpid2** line in `/etc/rc.tcpip` is commented out.

Flags

Item	Description
-d *Level*	Specifies tracing or debug level. **8** DPI level 1 **16** DPI level 2 **32** Internal level 1 **64** Internal level 2 **128** Internal level 3 Add the numbers for multiple trace levels. **Note:** If the **-d** flag is specified, but the level number is not specified, the default level is 56. If **-d** flag is not specified, the default level is 0.

Examples

1. To start the **dpid2** daemon, enter a command similar to the following command:

   ```
   startsrc -s dpid2 -a "-f /tmp/dpid2.log"
   ```

   This command starts the **dpid2** daemon and logs information to the `/tmp/dpid2.log` file at debug level 0.
2. To stop the **dpid2** daemon normally, enter the following command:

   ```
   stopsrc -s dpid2
   ```

   This command stops the **dpid2** daemon. The **-s** flag specified the subsystem that follows to be stopped.
3. To get the short status from the **dpid2** daemon, enter the following command:

   ```
   lssrc -s dpid2
   ```

   This command returns the name of the daemon, the process ID of the daemon, and the state of the daemon (active or inactive).

Files

Item	Description
`/etc/snmpd.conf`	Specify SMUX peer entry in `snmpd` version 1 agent configuration file.
`/etc/snmpd.peers`	Specify the configuration for SMUX peer.
`/etc/mib.defs`	Defines the MIB variables the SNMP agent and manager must recognize and handle.

dping Command

Purpose

Pings nodes or devices in parallel.

Syntax

dping [**-h**] [**-v**] [**-a**] [**-s**] [**-S**] [**-r**] [**-i** *interface...]* [**-w** " **selectstr**"] [**-H** *host_list*] [**-f** *filename*] [**-N** *nodegroup...*] [**-d** *devicename...*] [**-D** *devicegroup*] [[**-n**] *node_list*]

Description

The **dping** command pings the specified servers. The command can be used to retrieve node status or when you suspect a problem with Rational® Method Composer (RMC) and its heartbeating. The **dping** command is used to ping nodes or devices in parallel. **- Ping** nodes pings every second node interface in the series. For example, **eth1** or **mryi0** and **- Direct** nodes pings other nodes.

Keywords

Item	Description
-a	Specifies to ping all the nodes. This flag cannot be combined with the **-n**, **-N**, **-f**, **-d**, **-H**or **-w** flags.
-c	Collapses identical output from more than 1 node and displays it 1 time.

Item	Description
-d *devicename...*	Specifies one or more comma-separated devices to ping. The asterisk character (*) indicates all devices.
-D *devicegroup...*	Specifies one or more comma-separated device groups to ping.
-f *filename*	Specifies a file that contains a list of nodes. If the file name specified is the dash character (**-**), then the list is read from the standard input. The file can contain multiple lines, and each line can list one or more comma or space-separated node host names or node ranges.
-h	Displays command usage information.
-H *host_list*	Specifies the host name that is separated by comma or space to be pinged. These host names must not be defined NIM nodes. Space separated host names must be specified within double quotation marks. The **-H** flag cannot be specified with the **-n, -N**, **-f, -d, -w,** or the**-a** flags.
-i *interface..*	Specifies one or more comma-separated network interfaces to ping for each node specified. The flag assumes that *nodename-interface.domain* resolves to the IP address of that network adapter on the node. You must set up host name resolution before running **dping**. If one of the host names is an empty string, for example **"eth1,eth2"**, the flag also pings the primary host name.
-n *node_list*	Specifies one or more comma or space-separated node host names or node ranges to ping. This flag can be used simultaneously with **-N** and **-f**. The host name value can be specified without **-n**, if it is the last argument specified. See the noderange file for information about node ranges.
-N *nodegroup..*	Specifies one of more comma-separated node groups on which to run the command.
-r	Pings recursively. This flag runs **dsh** command to the nodes that pinged successfully. It pings all remaining nodes specified with the **dping** command, from those nodes.
-s	Pings the nodes serially instead of in parallel. This flag cannot be used with the **-S** flag.
-S	Displays a summary of the ping results only. This flag cannot be used with the **-s** flag.
-v	Specifies verbose mode.
-w *selectstr*	Displays the nodes that match the "where" clause of the select string. Specifying the entire string within double quotation marks will allow you to specify attribute value strings within single quotation marks. The asterisk character (*) indicates all nodes, as if a "where" clause were not specified. The **-w** flag cannot be specified with the **-n**, **-N**, **-f**, **-d**, **-H**, or **-a** flags.

Security

The command requires root access to the cluster management server.

Examples

1. To ping all nodes, enter:

   ```
   dping -a
   ```

   Output is similar to:

   node1.localdomain: ping (alive)

   node2.localdomain: noping (unreachable)

   node3.localdomain: ping (alive)

2. To ping the **group1** *nodegroup* and **eth1** node interface, enter:

```
dping -N group1 -i eth1
```

Output is similar to:

node1-eth1.localdomain: ping (alive)

node2-eth1.localdomain: noping (unreachable)

3. To ping the hostname **node1-eth2.clusters.com**, enter

```
dping -i eth2 node1.clusters.com
```

Output is similar to:

Exit Status 0 The command completes successfully. **1** The command failed. **10** No nodes or devices were specified.

drmgr Command

Purpose

The **drmgr** command is used to install and configure dynamic logical partitioning (DLPAR) scripts.

Syntax

drmgr { **-i** *script_name* [**-w** *minutes*] [**-f**] | **-u** *script_name* } [**-D** *hostname*]

drmgr [**-b**]

drmgr [**-R** *script_install_root_directory*]

drmgr [**-S** *syslog_ID*]

drmgr [**-l**]

Description

DLPAR scripts are provided by system administrators and vendors to coordinate the consumption of resources (for example, specific processors and large amounts of pinned memory) by applications and middleware with the addition or removal of those resources with respect to the operating system. DLPAR scripts are run both before and after DLPAR operations. DLPAR scripts are provided so that applications can be cleanly quiesced and restarted.

Note: The specified action flags cannot be combined. That is, a user cannot combine **-R** and **-S** flags, **-l** and **-R** flags, and so on.

Flags

Flag	Description
-b	Rebuilds the scripts' information file that is managed by the **drmgr** command. In general, this option must be used only when restoring scripts from another systems.
-D *hostname*	Specifies the hostname of the system on which the script can be started.
-f	Forces the replacement of an existing script.
-i *script_name*	Installs the *script_name* script. The *script_name* must have complete path. If the path is not specified, the current directory is assumed. If any name conflicts, the **drmgr** command issues a warning and does not install the script. Any existing script can be overwritten by specifying the **-f** flag.

Flag	Description
-l	Displays the details on the DLPAR scripts that are currently installed.
-R *base_script_directory*	Changes the base script installation directory.
-S *syslog_ID*	Logs syslog messages with the specified syslog ID string. This ID string is appended to every entry logged in syslog by the **drmgr** command.
-u *script_name*	Uninstalls a DLPAR script. If the script was installed with the **-D** option, the same parameter must be used to uninstall it. If no directory is specified, the **drmgr** command removes the DLPAR script from "all" installation directory.
-w *minutes*	Overrides the time limit value specified by the vendor for the script. The script is stopped if it exceeds the specified time limit.

Exit Status

0
Successfully completed the requested operation.

>0
The command failed. The cause of failure can be:

- File or directory does not exist.
- The length of the parameter exceeds the system limit (PATH_MAX).
- Too many arguments were specified.
- You do not have root authority to run this command.

drslot Command

Purpose

Manages a dynamically reconfigurable slot, such as, a hot plug slot.

Syntax

To Identify a Hot Plug Slot

drslot -i { -s *Slot* | **-l** *DeviceName* } **-c** *ConnectorType*

To Prepare a Hot Plug Slot for Configuring Devices

drslot -a -s *slot* **-c** *ConnectorType* [**-I**]

To Prepare a Hot Plug Slot for Removal of a Device

drslot -r { **-s** *slot* | **-l** *DeviceName*} **-c** *ConnectorType* [**-I**]

To Prepare a Hot Plug Slot for Removal and Replacement of a Device

drslot -R { **-s** *slot* | **-l** *DeviceName* } **-c** *ConnectorType* [**-I**]

Description

The **drslot** command manages dynamically reconfigurable slots, that is, hot plug slots. Hot plug slots are the plug-in points for connecting entities which can be configured without turning the system power off or rebooting the operating system. For the add (**-a**) operation, the slot must be specified directly by using the **-s** flag, giving the unique identifier for the slot. For the identify (**-i**), the remove (**-r**), and the replace (**-R**) operations, the slot may be specified directly with the **-s** flag, or indirectly. The slot may be specified indirectly by using the **-l** flag giving the logical name for a device connected to the slot. The **drslot** command determines to which slot the specified device is connected and manages that slot.

Notes:

1. The remove and replace operations fail unless the device connected to the identified slot has been unconfigured. For more information on how to successfully unconfigure a device, see Managing Hot Plug Connectors in *Operating system and device management*.
2. After an add or replace operation, you must run the **cfgmgr** command in order to make the new device active and ready for use by the operating system.

Flags

Note: Do not use the **-a**, **-i**, **-r**, **-R** flags together.

Item	Description
-a	Prepares a hot plug slot for configuring the device(s) connected to it. The slot is first identified to you and you are prompted for confirmation of the slot. Next, you are prompted for confirmation that the device has been connected to the slot. Upon confirmation that the device has been connected, the slot is prepared and the device is made ready for configuration.
-c *ConnectorType*	Specifies the *ConnectorType* of the *Slot* on which you are operating. For example, the *ConnectorType* for a hot plug PCI slot is `pci`. This flag is must be specified with the **-a**, **-i**, **-r**, and **-R** flags.
-i	Identifies a hot plug slot. The identification of the slot is hardware dependent. For example, if a slot has an LED associated with it, issuing the **drslot -i** command may cause the LED to flash.
-I	Specifies that the identification step should be skipped when using the **-a** (add), **-r** (remove), and **-R** (replace) flags. This flag should only be used when you are sure you have already identified the proper slot.
-l *DeviceName*	Specifies the *DeviceName*, which is the logical device name of the device connected to the slot to be managed. This flag must be used for the **-i** (identify), **-r** (remove) or **-R** (replace) flags if the **-s** flag is not used.
-r	Prepares a hot plug slot for removal of a device that has been previously unconfigured with the **rmdev** command, or the SMIT equivalent. The slot is identified and you are prompted for confirmation of the slot. If a visual indicator is associated with the slot, it is turned off. Finally, the slot is prepared for device removal and you are prompted for confirmation that the device has been removed from the slot.
-R	Prepares a hot plug slot for the removal of a device that has been previously unconfigured and the replacement with an identical device. The device must be unconfigured with the **rmdev** command, or the SMIT equivalent. **drslot** identifies the slot and you are prompted for confirmation of the slot. Next, the slot is prepared for the replacement of the device. You are then prompted to confirm that the device has been replaced. Upon confirmation that the device has been replaced in the hot plug slot, the slot is prepared and the device is made ready for configuration.
-s *Slot*	Specifies the *Slot* on which **drslot** should operate. This flag is required for the add (**-a**) operation. This flag must be used for the identify (**-i**), remove (**-r**) or replace (**-R**) operations if the **-l** flag is not used. The format of *Slot* is Platform/*ConnectorType* dependent.

Examples

1. To identify a specific PCI hot plug slot, enter:

```
drslot  -i  -c pci  -s U0.1-P1-I3
```

In this example, there is an LED associated with this slot. The system may display a message similar to the following:

```
The visual indicator of the specified PCI slot has been set to the identify
state. Press Enter to continue or enter x to exit.
```

The LED for the slot specified by `U0.1-P1-I3` flashes until the you press the Enter key.

2. To add a hot pluggable Ethernet adapter to a hot plug slot without confirmation of the slot, enter:

```
drslot  -a  -I  -c pci  -s U0.1-P1-I3
```

No confirmation prompt is given for identifying the slot. There will be a confirmation prompt displayed when it is time to put the new adapter into the slot, and a message similar to the following displays:

```
The visual indicator for the specified PCI slot has been set to the action
state. Insert the PCI card into the identified slot, connect any devices to
be configured, and press Enter to continue. Enter x to exit.
```

After connecting the adapter, press Enter, and the slot is prepared.

3. To identify a particular PCI slot before replacing the scsi card in it, enter the following:

```
drslot  -R  -c pci  -s U0.2-P1-I3
```

The system displays messages similar to the following:

```
The visual indicator of the specified PCI slot has been set to the identify
state. Press Enter to continue or enter x to exit.
```

The LED for the PCI slot blinks to identify the slot. Pressing any key but the Enter key exits the command. Pressing Enter continues with this slot. If continuing, the LED for the PCI slot is changed to the action state and the system displays a message similar to the following:

```
The visual indicator for the specified PCI slot has been set to the action
state. Replace the PCI card in the identified slot, reconnect any devices
to be configured, and press Enter to continue. Enter x to exit. Exiting now
leaves the PCI slot in the removed state.
```

Files

/usr/sbin/drslot

dscrctl Command

Purpose

Sets the default prefetch characteristics of an operating system.

Syntax

To query the characteristics of the hardware streams on the computer:

dscrctl -q

To set the default prefetch depth of the operating system on the computer, temporarily (for the current session) or permanently (after each restart operation):

dscrctl [**-n**] [**-b**] **-s** *dscr_value*

To cancel a permanent setting of the default prefetch depth of the operating system at start time:

dscrctl -c

Description

The **dscrctl -q** subcommand displays the number of hardware streams, and default prefetch depth of the platform and the operating system. Any user can run this subcommand.

The **dscrctl -s** subcommand sets the default prefetch depth of the operating system. You must have root authority to run this subcommand. This default value can be changed either for the current session by using **-n** flag, at start time by using **-b** flag, or for both current session and at start time by using **-n -b** flags together with the **dscrctl** command.

The **dscrctl -c** option cancels the default prefetch depth setting of the operating system at start time. This option removes the **dscrctl** command from the **/etc/inittab** file, and therefore takes effect after the next restart operation.

Flags

-q
Displays the number of hardware streams that are supported by the platform and also displays the values of the default prefetch depth of the firmware and the operating system.

-c
Cancels a permanent setting of the default prefetch depth at start time by removing the **dscrctl** command from the **/etc/inittab** file.

-n
Changes the run time value of the default prefetch depth of the operating system. This flag is used in conjunction with the **-s** flag. The change is not persistent from one boot operation to the next.

-b
Makes the change persistent across boot operations by adding the **dscrctl** command to the **/etc/inittab** file. This flag is used in conjunction with the **-s** flag.

-s *dscr_value*
Defines the value for the new default prefetch depth of the operating system. The value is treated as a decimal number, unless it starts with `0x` in which case it is treated as a hexadecimal number.

Examples

1. To set the value of the default prefetch depth of the operating system to 13 for the current session, enter:

```
# dscrctl -n -s 13
```

2. To show the current settings of the hardware stream mechanism, enter:

```
# dscrctl -q
```

The following output is displayed:

```
Current DSCR settings:
Data Streams Version = V2.06
number_of_streams = 16
platform_default_pd = 0x5 (DPFD_DEEP)
os_default_pd = 0xd (DSCR_SSE | DPFD_DEEP)
```

dscreen Command

Purpose

Starts the Dynamic Screen utility.

Syntax

dscreen [**-i** *InfoFile*] [**-t** *TermType*]

Description

The **dscreen** command starts the Dynamic Screen utility, which allows a single physical terminal to be connected to several virtual sessions, or screens, at one time.

If no flags are specified, the **dscreen** command reads the description for the terminal specified in the **TERM** environment variable from the file specified in the **DSINFO** environment variable. If the **DSINFO** environment variable is not specified, the terminal description is read from the **/etc/dsinfo** file. A terminal description typically contains the following configuration information:

- Keys used with the Dynamic Screen utility and their function
- Number of pages of screen memory the terminal has available
- Code sequences that must be sent or received to access and use Dynamic Screen features

Flags

Item	Description
-i *InfoFile*	Specifies the file that contains alternate key mappings for use with the Dynamic Screen utility. This option is useful when the originally defined Dynamic Screen keys conflict with one of your applications. If this flag is not specified, terminal configuration information is read from the file specified in the **DSINFO** environment variable, if set. Otherwise, information is read from the **/etc/dsinfo** file.
-t *TermType*	Identifies the terminal description to be read from the file containing the key mappings. This option is useful when the desired terminal type does not match the setting of the **TERM** environment variable.

Examples

1. To start the Dynamic Screen utility using key mapping defaults, enter:

   ```
   dscreen
   ```

   This sets the **DSINFO** and **TERM** environment variables as designated in the default **/etc/dsinfo** file.
2. To start the Dynamic Screen utility and specify a file that contains alternate key mappings and also identifies a terminal description to be read from the file, enter:

   ```
   dscreen -i myfile -t myterm
   ```

   This uses information from a user-created **dsinfo-**type file named `myinfo` to handle unusual key mapping needs. The `myinfo` file also contains a terminal definition named `myterm`.
3. To start the Dynamic Screen utility and specify an alternate terminal setup, enter:

   ```
   dscreen -t wy60-wp
   ```

   This terminal definition (maintained in the **/etc/dsinfo** file) sets **dscreen** assigned key actions so they do not conflict with control key command sequences in the word processing application being used.

Files

Item	Description
/etc/dsinfo	Contains the terminal descriptions for the Dynamic Screen utility.

dshbak Command

This command is part of the IBM Distributed Shell Management (DSM) software. The command is located at the */opt/ibm/sysmgt/dsm/bin/dshbak* location.

Purpose

Presents formatted output from the dsh command.

Syntax

dshbak [**-c** | **-x**]

Description

The **dshbak** command formats output from the dsh command. The syntax of the **dshbak** command is as follows:

host_name: line of output from **remote** command

The **dshbak** command formats the lines and writes them to the standard output as follows. The assumption is that the output from *host_name3* and *host_name4* are identical, and the **-c** flag is specified.

```
HOSTS -------------------------------------------------------
          host_name1
          ---------------------------------------------------------------
          .
          .
          lines from dsh with host_names stripped off
          .
          .
          HOSTS -------------------------------------------------------
          host_name2
          ---------------------------------------------------------------
          .
          .
          lines from dsh with host_names stripped off
          .
          .
          HOSTS -------------------------------------------------------
          host_name3             host_name4
          ---------------------------------------------------------------
          .
          .
          lines from dsh with host_names stripped off
          .
          .
```

The host names are displayed alphabetically, if the output is displayed from more than one node in a collapsed form. The output is sorted alphabetically by host name, if the output is not collapsed. The **dshbak** command writes "." for each 1000 lines of output filtered.

If the **-x** flag is specified, the extra header lines that **dshbak** command displays for each node is excluded. The **dshbak** command sorts the output using the node name to view the content:

```
host_name1: lines from dsh started
          .
          .
          lines from dsh continued
          .
          .
          lines from dsh ended
          host_name2: lines from dsh started
```

```
.
.
lines from dsh continued
.
.
lines from dsh ended
```

Flags

Item	Description
-c	Collapses identical output from more than 1 node to display the output at one time.
-x	Excludes the extra header lines that **dshbak** displays for each node. This flag provides compact output, and **dshbak** command sorts the output by node name to view the content. The flag must not be used with **-c**.

Security

Note: You must run the **kinit** command to obtain a ticket-grant-ticket before running the Kerberos Version 5 **remote** commands. The additional security considerations are like that of the **remote shell** command.

Examples

1. To display the results of a command issued on several nodes, in the format used in the Description, enter the following command:

   ```
   dsh -n node1,node2,node3 cat /etc/passwd | dshbak
   ```

2. To display the results of a command issued on several nodes with identical output, enter the following command:

   ```
   dsh -w host1,host2,host3 pwd | dshbak -c
   ```

3. To display the results of a command issued on several nodes with compact output, enter the following command:

   Note: The output is sorted alphabetically by host name.

   ```
   dsh -w host1,host2,host3 date | dshbak -x
   ```

Standard Error

The error messages on standard error is displayed before all standard output messages, if the **dshbak** filter is used. This behaviour is true with and without the **-c** flag.

dsh Command

Purpose

Runs commands concurrently on multiple nodes and hardware devices.

Syntax

dsh -h dsh -V dsh -q dsh [**-a**] [**--all-nodes** *context_list*]

[**-A**] [**--all-devices** *context_list*] [**-n**

node_list] [**-N** *nodegroups*] [**-d** *device_list*]

[**-D** *devicegroups*] [**-C** *context*]

[**-c**] [**-e**] [**-E** *environment_file*

] [**-f** *fanout*] [**-F** *output_path*]

[**-i**] [**-l** *user_ID]* [**-L**]

[**--log** *log_file*] [**-m**] [**-o**

node_options] [**-O** *device_options*] [**-Q**]

[**-r** *node_remote_shell*] [**--device-rsh**

device_remote_shell] [**-s**] [**-S**

csh | **ksh**] [**-t** *timeout*]

[**-T**] [**-v**] [**-X**

env_list] [**-z**] [**--report**

report_path] [**--report-name** *report_name*] [*command_list*]

Description

The **dsh** command runs commands concurrently on remote targets - nodes, hardware devices, or both. Targets can be selected from multiple contexts. A context is a target database that contains node and device definitions, such as the NIM database. The **dsh** command issues a **remote shell** command for each target specified. It returns output from all targets in a formatted manner to enable the command results from all nodes to be managed easily. The**/usr/bin/rsh** is the model for syntax and security and the dsh command is a DSM Distributed Shell Utility.

Note: The **dsh** command is supported only on IPv4 target devices. IPv6 addresses are not supported currently.

Parameters

Item	Description
TARGET CONTEXT	The **dsh** command target context is the database where the target or target group is defined. A default context can be configured using the **-C** context flag or the **DSH CONTEXT** environment variable. If either parameter is not specified, the default context is **NIM** when the dsh command is run from a NIM management server, else the default context is **DSH** (see **DSH CONTEXT**). A context is used with a **DSH Utilities** command by installing a context extension file in the **/opt/ibm/sysmgt/dsm/pm/Context** directory. The target or target group context can be explicitly specified by qualifying a target name with a context name, or implicitly defined by specifying a default context for unqualified target names (see **Target list**).
DSH CONTEXT	The **DSH CONTEXT** is the in-built context for all the **DSH Utilities** commands. It permits a user-defined node group database contained in the local file system. The **DSH_NODEGROUP_PATH** environment variable specifies the path to the node group database. Each file in this directory represents a node group, and contains one host name or TCP/IP address for each node that is a group member. Blank lines and comment lines beginning with a # symbol are ignored. If all nodes are requested for the **DSH CONTEXT**, a full node list is built from all groups in the **DSH_NODEGROUP_PATH** directory, and cached in **/var/ibm/sysmgt/dsm/dsh/$DSH_NODEGROUP_PATH/AllNodes**. This file is recreated each time a group file is modified or added to the **DSH_NODEGROUP_PATH** directory. Device targets are not supported in the **DSH** context.

Item	Description
TARGET SPECIFICATION	A target is a node or hardware device where a remote command is issued. Node targets are specified using the **-a, --all-nodes** *context_list*, **-n** *node_list* and **-N** *nodegroups* flags, or the **DSH_NODE_LIST** environment variable. If both the **-N** flag and the **DSH_NODE_LIST** environment variable are used, the groups and lists are merged eliminating any duplicates. **Note:** The **DSH_NODE_LIST** environment variable has replaced **WCOLL**. Device targets are specified using the **-A, --all-devices** *context_list*, **-d** *device_list* and **-D** *devicegroups flags*, or the **DSH_DEVICE_LIST** environment variable. If the local host is included as part of the targets, the *command_list* is run directly on the local host and not through the configured remote shell, unless a *user_ID* is specified for execution with the local host (see Remote user). The **DSH_NODE_LIST** and **DSH_DEVICE_LIST** environment variables specify files listing target nodes and devices. The file format is 1 target per line. Blank lines and comment lines beginning with a # symbol are ignored. Both node and device targets can be specified simultaneously, but the same target name cannot be used for both a device and a node. If a similar name is used, the program skips the duplicate targets, and continues execution on the other targets. Node and device targets can also be specified using node ranges; see the **noderange** file for details. If the same target is specified more than one time, the **remote** command is run one time on the specified target.
TARGET LIST	Target and target groups are specified using the following format: [*context*:] [*user_ID*@] *target* [, [*context*:] [*user_ID*@] *target*]... where context is the explicit context specification for the target, *user_ID* is the optional user name to use when remotely running the command on the target, and *target* is the name or TCP/IP address of the target as permitted by the target's context. For a **noderange** expression, the *user_ID* is used for each target in the list that results from the evaluation of the **noderange** expression. If the target list is only specified with a **dash (-)**, targets can be specified interactively. For nodes, the prompt is **dsh node>**; and for devices the prompt is **dsh device>**. Specify the target list using the following syntax on 1 line at a time: [*context*:] [*user_ID*@] *target* where context is the explicit context specification for the target, *user_ID* is the optional user name to use when remotely running the command on the target, and target is the name or TCP/IP address of the target as permitted by the context of the target. For a **noderange** expression, the *user_ID* is used for each target in the list that results from the evaluation of the **noderange** expression. If the target list is only specified with a dash (-), targets can be specified interactively. For nodes, the prompt is **dsh node>**; for devices the prompt is **dsh device>**. Specify the target list using the following syntax on 1line at a time: [*context:*][*user_ID*@]*target* When you are finished, press Ctrl-d to continue.

Item	Description
COMMAND SPECIFICATION	The commands to run on the remote targets are specified by the *command_list* dsh parameter, entering commands on the command line in interactive mode, providing a *command_list* through standard input, or running a local script using the **-e** flag. The syntax for the *command_list* dsh parameter is as follows: "**command[; command]...**" where command is the command to run on the remote target. Quotation marks are required to ensure that all commands in the list are run remotely, and that any special characters are interpreted correctly on the remote target. A script file on the local host is run on each of the remote targets by using the **-e** flag. If **-e** is specified, *command_list* is the script name and arguments to the script. For example: `dsh -e[flags] script_filename [arguments]...` The *script_filename* file is copied to a random file name in the **/tmp** directory on each remote target and then run on the targets. If the *command_list* parameter is not specified, dsh enters interactive command-line mode and prompts the **dsh>** prompt. Enter commands at the **dsh>** prompt using the following syntax: **[!] "command"** where command is the command to run on the remote target. An exclamation point (!) preceding a command causes the command to run on the local host only, and not on remote targets. The dsh command runs command on resolved target, and the result is displayed. It then returns to the dsh> prompt. To exit command-line mode, enter exit at the dsh> prompt. The dsh command does not work with any interactive commands, including those commands read from standard input.
REMOTE USER	The *user_ID* to use for a remote target can be specified as part of the target syntax (see Target lists) or using the **-l** (lowercase L) flag. If both methods are used, the *user_ID* is determined as follows: 1. For targets specified as *user_ID@target*, *user_ID* is used for remote execution on the target, and the *-l* flag is ignored. 2. For targets not specified using *user_ID@target*, the *user_ID* used for remote execution on the target is determined as follows: - The *user_ID* specified with the **-l** flag. If **-l** is not specified, the current user running the command is specified. If a *user_ID* is specified for the local host included in a target list, the **remote shell** command runs the *command_list* on the local host to ensure a secure login ID.
REMOTE SHELL COMMAND	The commands to run on the remote targets are specified by the *command_list* dsh parameter, entering commands on the command line in interactive mode, providing a *command_list* through standard input, or executing a local script using the **-e** flag. The syntax for the *command_list* dsh parameter is as follows: "**command[; command]...**" where command is the command to run on the remote target. Quotation marks are required to ensure that all commands in the list are run remotely, and that any special characters are interpreted correctly on the remote target. A script file on the local host can be run on each of the remote targets by using the **-e** flag. If **-e** is specified, **command_list** is the script name and arguments to the script. For example: dsh *-e[flags] script_filename [arguments]...* The *script_filename* file is copied to a random file name in the **/tmp** directory on each remote target and then run on the targets. If the **command_list** parameter is not specified, dsh enters interactive command-line mode and the dsh> prompt is displayed. Enter commands at the dsh> prompt using the following syntax: **[!] "command"**, where command is the command to run on the remote target. An exclamation point (!) preceding a command causes the command to run on the local host only, and not to any remote targets. The dsh command runs each command to each resolved target, the results are displayed and returns to the dsh> prompt. To exit command-line mode, enter exit at the dsh> prompt. The dsh command does not work with any interactive commands, including those commands that read from standard input.

Item	Description
REMOTE USER	The **user_ID** to use for a remote target can be specified as part of the target syntax (see Target lists) or using the **-l** (lowercase L) flag. If both methods are used, the *user_ID* is determined as follows: 1. For targets specified as *user_ID@target*, *user_ID* is used for remote execution on the target, and the *-l* flag is ignored. 2. For targets not specified using *user_ID@target*, the *user_ID* used for remote execution on the target is determined as follows: - The *user_ID* specified with the *-l* flag. If **-l** is not specified, the current user running the command. If a **user_ID** is specified for the local host included in a target list, the **remote shell** command runs the *command_list* on the local host to ensure a secure login.
REMOTE SHELL ENVIRONMENT	The shell environment used on the remote target defaults to the shell defined for the **user_ID** used for remote command execution. The command syntax used for remote command execution can be specified using the **-S** flag. If **-S** is not specified, the syntax defaults to ksh syntax. When commands are run on the remote target, the path used is determined by the **DSH_PATH** environment variable defined in the shell of the current user. If **DSH_PATH** is not set, the path used is the remote shell default path. For example, to set the local path for the remote targets, use: **DSH_PATH=$PATH**. The **-E** flag exports a local environment definition file to each remote target. Environment variables specified in this file are defined in the remote shell environment before the *command_list* is run.
COMMAND EXECUTION	The maximum concurrent **remote shell** command processes (the fanout) can be specified with the **-f** flag or with the **DSH_FANOUT** environment variable. The fanout is restricted by the number of **remote shell** commands that can be run in parallel. You can experiment with the **DSH_FANOUT** value on your management server to see if higher values are appropriate. A timeout value for remote command execution can be specified with the **-t** flag or with the **DSH_TIMEOUT** environment variable. If any remote target does not provide output to either standard output or standard error within the timeout value, an error message is displayed by the dsh command and exits. If streaming mode is specified with the **-s** flag, the output is returned as it becomes available from each target. This process does not wait for the *command_list* to complete on all targets before returning output. This can improve performance but causes the output to be unsorted. The **-z** flag is used to display the exit code from the last command issued on the remote node in *command_list*. **Note:** OpenSSH returns the exit status of the last remote command issued as its exit status. This affects the behavior of dsh and requires using the **-c** flag. If the command issued on the remote node is run in the background, the command does not display the exit status. The **-m** flag monitors execution of the dsh command by printing status messages to standard output. Each status message is preceded by dsh>. The **-T** flag provides diagnostic trace information for the execution of the dsh command. Default settings and the actual **remote shell** commands run on the remote targets are shown. No error detection or recovery mechanism is provided for remote targets. The dsh command output to standard error and standard output can be analyzed to determine the appropriate course of action. In interactive mode, if a command cannot be run on a remote target (for example, a **remote shell**l command resulting in a non-zero return code), subsequent commands are not sent to this node on this invocation of the dsh command unless the **-c** flag is specified.

Item	Description
COMMAND OUTPUT	The **dsh** command waits to display the output from each remote shell process and then initiates a new remote shell process. This default behavior is overridden by the **-s** flag. The dsh command output consists of standard error and standard output from the remote commands. The dsh standard output is the standard output from the remote shell command. The dsh standard error is the standard error from the remote shell command. Each line is prefixed with the host name of the node that produced the output. The host name is followed by the: character and a command output line. A filter to display identical outputs grouped by node is provided separately. See the **dshbak** command for more information. Output for each target can be copied to a file using the **-F** *output_path* flag. Standard output for each target is written to the *target.output* file in the *output_path* directory, and standard error for each target is written to the *target.error* file in the **output_path** directory. The **-F** flag does not suppress output on the console. A command can be run silently using the **-Q** flag; no output from each target's standard output or standard error is displayed. If the **-F** flag is specified, output continues to be written to output files.
REPORTING	Output from the dsh command can be saved to a report on the local host. The **--report** *report_path* flag enables report generation to the specified *report_path* directory. Reporting is activated by defining the **DSH_REPORT** environment variable with the *report_path*. The **--report flag** overrides the **DSH_REPORT** environment variable. The **--report-name** flag defines a report name, if reporting is activated. The **report name** is also the subdirectory of *report_path* that contains the report files. A numeric index is appended to the subdirectory name to allow multiple reports with the same name. If the **--report-name** flag is not used, the name defaults to Unspecified. Summary HTML and XML report files are created, in addition to an XML results file. SIGNALS: Signal 2 (INT), Signal 3 (QUIT), and Signal 15 (TERM) are propagated to the commands executing on the remote targets. Signal 19 (CONT), Signal 17 (STOP), and Signal 18 (TSTP) default to dsh; the dsh command responds normally to these signals, but the signals does not affect on the remotely executing commands. Other signals are determined by dsh and have their default effects on the dsh command; all current child processes, through propagation to remotely running commands, are terminated (SIGTERM). Parameters *command_list* Specifies a list of commands to execute on the remote targets. The syntax for the *command_list* parameter is as follows: " **command[; command...**"

Keywords

Item	Description
-a	Includes in the target list all nodes that are defined in the default context. The default context can be set using the **-C** flag or the **DSH_CONTEXT** environment variable.
-A	Includes in the target list all devices that are defined in the default context. The default context can be set using the **-C** flag or the **DSH_CONTEXT** environment variable. This flag is disabled on HMCs.
--all-nodes*context_list*	Includes in the target list all nodes defined in the contexts listed in *context_list.* The default context is not implicitly included in this list. This flag is disabled on HMC. **--all-nodes**.
--all-devices *context_list*	Includes in the target list all devices that are defined in the contexts listed in *context_list*. The default context is not implicitly included in this list. This flag is disabled on HMCs.
-C | **--continue**	In interactive mode only, keeps a node in the target list even if the remote shell command for the host has a non-zero exit value.

Item	Description
-C| **--context***context*	The default context to use when resolving target names. The context value must correspond to a valid context extension module in the **/opt/ibm/sysmgt/dsm/pm/Context** directory. For example, the **/opt/ibm/sysmgt/dsm/pm/Context/DSH.pm** file is the module for the DSH context.
-d | **--devices** *device_list*	Specifies a list of device targets to include in the target list. The *device_list* syntax is: [*context:*][*user_ID@]device_name*[,[*context:*]\ [*user_ID@*]*device_name*]... This flag is disabled on HMCs.
--device-rsh*device_remote_shell*	Specifies the full path of the remote shell command used for remote command execution on device targets. A remote shell for a specific context can be defined by including context: before the path. The *device_remote_shell* syntax is: [*context:*]*path*[,[*context:*]*path*]... This flag is disabled on HMCs. -
-D | **--devicegroups** *devicegroups*	Includes in the target list all devices defined in the device groups specified in the *devicegroups* list. The *devicegroups* syntax is: [*context:*] [*user_ID@*]*devicegroup*[,[*context:*]\ [*user_ID@*]*devicegroup*]... This flag is disabled on HMCs.
-e | **--execute**	Indicates that *command_list* specifies a local script filename and arguments to be executed on the remote targets. The script file is copied to the remote targets and then remotely executed with the given arguments. The *DSH_NODE_RCP* and *DSH_DEVICE_RCP* environment variables specify the remote copy command to use to copy the script file to node and device targets, respectively.
-E | **--environment** *environment_file*	Specifies that the *environment_file* contains environment variable definitions to export to the target before executing the*command_list*. The *DSH_NODE_RCP* and *DSH_DEVICE_RCP* environment variables specify the remote copy command to use to export the file to node and device targets, respectively.
-f | **--fanout** *fanout_value*	Specifies a fanout value for the maximum number of concurrently executing remote shell processes. Serial execution can be specified by indicating a fanout value of 1. If **-f** is not specified, a default fanout value of **64** is used.
-F | **--output** *output_path*	Copies standard output to *output_path*/*target_name.output* and standard error to *output_path*/*target_name.error*. Output continues to be sent to standard output and standard error. Use the **-Q** flag to suppress standard output and standard error.
-i | **--notify**	Indicates that a target is not responding and prompts to continue remote execution for the target. Specify the **-v** flag with the **-i** flag.
-l (lowercase L) |**--user** *user_ID*	Specifies a remote user name to use for remote command execution.
-h | **--help**	Displays command usage information.

Item	Description
-n | **--nodes** *node_list*	Specifies a list of node targets to include in the target list. The *node_list* syntax is: [*context:*] [*user_ID@*]*node_name*[, [*context*:]\ [*user_ID@*]*node_name*]...
-L | **--no-locale**	Specifies to not export the locale definitions of the local host to the remote targets. Local host locale definitions are exported by default to each remote target. Output is appended to the file for each execution of the dsh command.
--log *log_file*	Enables logging to the specified *log_file*
-m | **--monitor**	Monitors remote shell execution by displaying status messages during execution on each target.
-N | **--nodegroups** *nodegroups*	Includes in the target list all nodes defined in the node groups specified in the *nodegroups* list. The syntax of *nodegroups* is: [*context:*] [*user_ID@*] *nodegroup* [, [*context*:]\ [*user_ID@*]*nodegroup*]...
-o--node-options *node_options*	Specifies options to pass to the **remote shell command** for node targets. The options must be specified within double quotation marks ("") to distinguish them from dsh options. Options for nodes in a specific context can be defined by including a context: before the option list. The syntax for *node_options* is: [*context:*]" *options* "[,[*context:*]"*options*"]...
-O | **--device**-options *device_options*	Specifies options to pass to the **remote shell command** for device targets. The options must be specified within double quotation marks to distinguish them from dsh options. Options for devices in a specific context can be defined by including context: before the option list. The syntax for *device_options* is [*context*:]"*options*"[,[*context*:]"*options*"]... This flag is disabled on HMCs.
-Q |**--silent**	Specifies silent mode. No target output is written to standard output or standard error. Monitoring messages are written to standard output.
-q | **--show-config**	Displays the current environment settings relevant to all DSH Utilities commands. This includes the values of all environment variables and settings for all currently installed and valid contexts. Each setting is prefixed with context: to identify the source context of the setting.
-r | **--node-rsh** *node_remote_copy*	Specifies the full path of the **remote shell command** used to copy files to or from node targets. A **remote shell command** for a specific context can be defined by including context: before the path. The *node_remote_copy* syntax is: [*context:*]*path*[,[*context:*]*path*]... If path contains **rsync**, it is assumed that the **rsync** command performs the remote copy.

Item	Description
--report *report_path*	Enables report generation and specifies the path to the directory where reports are saved. **--report-name** *report_name* Specifies the name to use when generating the report. If not specified, the name defaults to Unspecified. This flag can only be used with the **--report** flag.
-s| **--stream**	Specifies to return output as it becomes available from each target. It does not wait for the *command_list* to complete on a target before returning output.
-S | **--syntax csh** | **ksh**	Specifies the shell syntax to be used on the remote target. If not specified, the ksh syntax is used.
-t |***--timeout****timeout*	Specifies the time, in seconds, to wait for output from any currently executing remote targets. If no output is available from any target in the specified timeout, an error message is displayed by the dsh command and terminates execution to the remote targets that failed to respond. If timeout is not specified, dsh waits indefinitely to continue processing output from all remote targets. When specified with the **-i** flag, the user is prompted for an additional timeout interval to wait for output.
-T | **--trace**	Enables trace mode. The dsh command prints diagnostic messages to standard output during execution to each target.
-v | **--verify**	Verifies each target before executing any remote commands on the target. If a target is not responding, execution of remote commands for the target is canceled. When specified with the -i flag, the user is prompted to retry the verification request.
-X *env_list*	Ignores dsh environment variables. This option can take an argument which is a comma-separated list of environment variable names that must NOT be ignored. If there is no argument to this option, or the argument is an empty string, all dsh environment variables are ignored. This flag cannot be specified as the last flag.
-V | **--version**	Displays version information for the dsh command.

Item	Description
-z | **--exit-status**	Displays the exit status for the last remotely executed non-asynchronous command on each target. If the command issued on the remote node is run in the background, the exit status is not displayed. Exit Status Exit values for each remote shell execution are displayed in messages from the dsh command, if the remote shell exit values are non-zero. A non-zero return code from a remote shell indicates that an error was encountered in the remote shell. This return code is unrelated to the exit code of the remotely issued command. If a remote shell encounters an error, execution of the remote command on that target is bypassed. The dsh command exit code is 0 if the command executed without errors and all remote shell commands finished with exit codes of 0. If internal dsh errors occur or the remote shell commands do not complete successfully, the dsh command exit value is greater than 0. The exit value is increased by 1 for each successive instance of an unsuccessful remote command execution. If the remotely issued command is run in the background, the exit code of the remotely issued command is 0. Environment Variables **DSH_CONTEXT** Specifies the default context to use when resolving targets. This variable is overridden by the **-C** flag. **DSH_DEVICE_LIST** Specifies a file that contains a list of device targets.

Item	Description
	This variable is overridden by the **-d** flag. This environment variable is ignored on HMCs. **DSH_DEVICE_OPTS** Specifies the options to use for the remote shell command with device targets only. This variable is overridden by the **-O** flag. This environment variable is ignored on HMCs. **DSH_DEVICE_RCP** Specifies the full path of the remote copy command used to copy local scripts and local environment configuration files to device targets. This environment variable is ignored on HMCs. **DSH_DEVICE_RSH** Specifies the full path of the remote shell to use for remote command execution on device targets. This variable is overridden by the **--device-rsh f**lag. This environment variable is ignored on HMCs. **DSH_ENVIRONMENT** Specifies a file that contains environment variable definitions to export to the target before executing the remote command. This variable is overridden by the **-E** flag. **DSH_FANOUT** Specifies the fanout value. This variable is overridden by the **-f** flag. **DSH_LOG** Specifies the full path of the file to use for logging. This variable is overridden by the **--log** flag. **DSH_NODE_LIST** Specifies a file containing a list of node targets. The **DSH_NODE_LIST** variable has replaced WCOLL.Hel DSH_NODE_OPTS . Specifies the options to use for the remote shell command with node targets only. This variable is overridden by the **-o** flag. **DSH_NODE_RCP** Specifies the full path of the remote copy command to use to copy local scripts and local environment configuration files to node targets. **DSH_NODE_RSH** Specifies the full path of the remote shell to use for remote command execution on node targets. This variable is overridden by the **-r** flag. **DSH_NODEGROUP_PATH** Specifies a colon-separated list of directories that contain node group files for the DSH context. When the **-a** flag is specified in the DSH context, a list of unique node names is collected from all node group files in the path. **DSH_OUTPUT** . Specifies the base file name for standard output and standard error copies. Output continues to be sent to standard output and standard error. This variable is overridden by the **-F** flag. **DSH_PATH** Sets the command path to use on the targets. If **DSH_PATH** is not set, the default path defined in the profile of the remote *user_ID* is used. **DSH_PATH** cannot be used to run a dsh command to an HMC. **DSH_REPORT**. Enables reporting when set to the absolute path of the directory where reports are saved. This variable is overridden by the **--report** flag. **DSH_SYNTAX** Specifies the shell syntax to use on remote targets; ksh or csh. If not specified, the ksh syntax is assumed. This variable is overridden by the **-S** flag. **DSH_TIMEOUT** Specifies the time, in seconds, to wait for output from each remote target. This variable is overridden by the **-t** flag. Security The dsh command has no security configuration requirements. All remote command security requirements - configuration, authentication, and authorization - are imposed by the underlying remote command configured for dsh. The command assumes that authentication and authorization are configured between the local host and the remote targets. Interactive password prompting is not supported; an error is displayed and execution is bypassed for a remote target if password prompting occurs, or if either authorization or authentication to the remote target fails. Security configurations as they pertain to the remote environment and remote shell command are user-defined. When the remote command is configured as **/usr/bin/rsh** and this command is configured to use Kerberos Version 5, you must first run the Kerberos **kinit** command to obtain a ticket-granting ticket, and you must ensure that your Kerberos principal is in the. **k5login** file in the home directory of the remote user on the targets.

Examples

1. To run the ps command on node targets **node1** and **node2**, enter:

```
dsh -n node1,node2 "ps"
```

2. To run the **ps** command on each node target listed in the **myhosts** file, enter:

```
DSH_NODE_LIST=./myhosts; dsh ps
```

3. To enter commands in interactive mode for execution on the node targets defined in **NodeGroup1**, enter:

```
dsh -N NodeGroup1
```

4. To display the number of users on all NIM Managed nodes and in the DSH context node group **NodeGroup2**, enter:

```
dsh --all-nodes NIM -N DSH:NodeGroup2 "who | wc -l"
```

5. To enter a list of node targets and device targets interactively and then execute the date command in interactive mode, enter:

```
dsh -n - -d -
```

   Additional input and the output similar to the following is displayed:

```
dsh node> node1
dsh node> gregb@node2
dsh node>
dsh device> CSM:kathyc@device1
dsh device>
dsh> date  node1: Wed Apr 13 17:15:59 EDT 2005
gregb@node2: Wed Apr 13 17:15:59 EDT 2005
kathyc@device1: Wed Apr 13 17:15:59 EDT 2005
dsh> exit #
```

6. To run the **ls** command on all the nodes in the cluster and ignore all the dsh environment variables, enter:

```
dsh -X -a ls
```

7. To run the **ps** command on **node1** and ignore all the dsh environment variables except the **DSH_NODE_OPTS**, enter:

```
dsh -n node1 -X 'DSH_NODE_OPTS' ps
```

dslpaccept Command

Purpose

Accept print queue requests for directory-enabled System V print systems.

Syntax

dslpaccept *PrintQueueName*

Description

The **dslpaccept** and **dslpreject** commands are used to set a print queue so that it will accept or reject print requests being queued for it. Unlike the **accept** and **reject** commands, the directory-enabled commands can control remote print systems, so long as they are directory-enabled. This is because they write directly to the print queue object on the directory server.

The user of this command must be directory-enabled and have permissions set for write, modify, search and read on the directory, in the directory context in which the user is administrator.

Parameters

Item	Description
PrintQueueName	The *PrintQueueName* parameter is the relative distinguished name (RDN) of the print queue object. Multiple print queue names may be specified in a comma-separated list.

Exit Status

0
Indicates success.

1
Indicates invalid options.

2
Indicates that the specified print queue is unknown.

3
Indicates that this user does not have modify permissions.

4
Indicates that an invalid RDN was supplied.

5
Indicates that the value is already set.

6
Indicates that the command is unable to contact the directory service

7
Indicates any other error.

Examples

1. To set the print queue "hpcolor" to accept requests:

```
dslpaccept hpcolor
```

dslpaccess Command

Purpose

Allow or deny non-directory enabled users and systems access to a print queue for a System V print subsystem.

Syntax

dslpaccess **-q** *QueueName* **-a** *AllowList* | **-d** *DenyList*

Description

The **dslpaccess** command either allows or denies users and systems access to a directory-enabled print queue. It is modeled on the **lpadmin** command's **-u** option.

Allow and deny lists consist of a comma-separated list of entries, each of which may specify a login ID, or a system name and login ID, as follows:

```
[[LoginID]|[System!LoginID]],[[LoginID]|[System!Login-ID]],...
```

LoginID or *System*, or both, can be set to the wildcard **all**, allowing or denying all appropriate entries. Use **all** with care. When the **all** entry is added to one list, all non-**all** entries are removed from the other list, for the appropriate value of *LoginID* or *System*. The default for *System* is the local host.

The user of this command must be directory-enabled and have permissions set for write, modify, search and read on the directory, in the directory context in which they are administrator.

Flags

Item	Description
-a *AllowList*	Specifies a list of users to add to the allow list. If present, these are deleted from the deny list. This option can not be used with the **-d** option.
-d *DenyList*	Specifies a list of users to add to the deny list. If present, these are deleted from the allow list. This option can not be used with the **-a** option.
-q *QueueName*	The queue-name parameter is the Relative Distinguished Name (RDN) of the print queue. If the print queue name does not exist in the directory context, the command fails.

Exit Status

0
Indicates success.

1
Indicates invalid options.

2
Indicates that the specified print queue is unknown.

3
Indicates that the user does not have appropriate access control permissions.

4
Indicates that an invalid RDN was supplied.

5
Indicates that the value is already set.

6
Indicates any other error.

Examples

1. The following grants user fredb access to print queue printq1 on host systemX:

```
dslpaccess -q printq1 -a systemX!fredb
```

2. The following denies access to print queue printq1 to user tomt for all hosts:

```
dslpaccess -q printq1 -d all!tomt
```

dslpadmin Command

Purpose

Configure directory-enabled print service for a System V print subsystem.

Syntax

dslpadmin [[**-q** *PrintQueueName* [**-D** *QueueDescription*] [**-n** *LocalQueueName*] [**-o banner** | **nobanner**] [**-A mail** | **none**] [**-F** *FaultRecovery*] [[**-P** *PhysicalPrinterName*] [**-s** *NetworkEntityName*]]] [**-P** *PhysicalPrinterName* [**-T** *PrinterType*] [**-l** *Location*] [**-L** *PDLList*]] [**-q** *PrintQueueName* **-P** *PhysicalPrinterName* [**-I** *ContentType*] [[**-i** *InterfaceScript*] | [**-m** [**Standard** | **PS**]]] [**-o** *PrintOptions*]] [**-q** *PrintQueueName* [**-I** *ContentType*]]] [**-q** *PrintQueueName* **-s** *NetworkEntityName* [**-a** *PrintSystemDNSName* | *PrinterSystemAddress*] [**-t BSD** | **HPNP**]]

dslpadmin [**-q** *PrintQueueName* [**-u** *PhysicalPrinterName*] [**-U** *ObjectRDN*]]

dslpadmin [**-x** *PrintQueueName*] [**-X** *PhysicalPrinterName*] [**-r** *NetworkEntityName*]

dslpadmin [**-h**]

Description

The **dslpadmin** command is used to perform the following functions in order to configure a directory-enabled print service:

- Add print queues and physical printers to the system.
- Modify print queues and physical printers.
- Remove print queues and physical printers from the system.
- Add and delete network entity objects for networked printers.

The dslpadmin command provides directory-aware versions of the functionality supplied by **lpadmin** (which is not directory-aware), and continues to use the traditional ``flat file'' configuration system. Note that where both systems are in use, the printer subsystem employs information found in the directory first. It is the responsibility of the administrator to ensure that naming conflicts do not arise between the two configuration systems.

The directory-enabled commands use Relative Distinguished Names (RDNs), rather than Distinguished Names (DNs). For example, to create a directory-enabled queue with a DN of "`cn=test,ou=printq,ou=print,cn=aixdata`", only the RDN "test" is to be used for the *PrintQueueName*.

When configuring a print queue where the administrator is not on the system that is to host the print queue, the *InterfaceScript* parameter of **-i** and the *PrinterType* parameter of **-T** are not checked. This is because the remote system cannot be accessed in order to do the checks. It is therefore the administrator's responsibility to ensure that the specified *InterfaceScript* and *PrinterType* exist on the remote hosting system.

A command line can contain any combinations of the **-q** , **-P** and **-s** flags, or any combination of the **-x**, **-X** and **-r** flags, but only one of each flag. When multiple directory objects are simultaneously created or modified, appropriate links are set up between the three object types (printers, print queues and network entities).

Flags

Item	Description
-a *PrinterSystemDNSName* | *PrinterSystemAddress*	Associates a DNS name or network address with the system. If the argument given can be interpreted as an IPv4 or IPv6 address, it is an address, if not it is assumed to be a DNS name. The -a flag causes the network entity object specified by **-s** to be modified, or else created if it does not already exist. The administrator should ensure that network entity objects are given unique names, so as to avoid modifying existing UNIX system objects instead of adding new print system objects. This flag requires the **-s** flag.

Item	Description
-A [**mail** | **none**]	Instructs the print system to generate a mail message if a print request fails. The mail is sent to the owner of the physical printer, or to the root user of the system hosting the print queue, if the printer has no owner or the user has no mail address. The default is **none**. This flag requires the **-q** flag.
-D *QueueDescription*	Defines a description comment for the print queue object specified with the **-q** flag. This description is displayed whenever a user asks for a full description of a print queue using the **lpstat** command. Strings containing whitespace should be double quoted. This flag requires the **-q** flag.
-F *FaultRecovery*	Defines the print queue's fault recovery mode. This flag specifies the recovery to be used if the printer on a print queue fails while printing a print request. The value of *FaultRecovery* can be any of the following: **continue** Continue printing on the top of the page where printing stopped. This requires a filter to wait for the fault to clear before automatically continuing. **beginning** Start printing the request again from the beginning. **wait** Disable printing on *PhysicalPrinterName* and wait for the administrator or a user to enable printing again. During the wait the administrator or the user who submitted the stopped print request can issue a change request that specifies where printing should resume. If no change request is made before printing is enabled, printing resumes at the top of the page where it stopped, if the filter allows; otherwise, the request is printed from the beginning. The default value of *FaultRecovery* is **beginning**. This flag requires the **-q** flag.
-h	Displays a brief help screen.
-i *InterfaceScript*	Pathname for the printer's *InterfaceScript* when accessed through the specified print queue. This flag is not valid if the **-P** flag has not been specified. The interface scripts are usually supplied by the user. This flag cannot be used when **-m** has also been specified. This flag requires both the **-q** and the **-P** flags.
-I *ContentType*[, *ContentType*, ...]	Specifies the print queue's content types. Allows the print queue to handle print requests with the content types in the list. If the list contains more than one *ContentType*, the *ContentType* parameters must be separated by commas. See the **lpadmin** manual page for a full description of the format. This also requires the **-P** flag and the **-q** flag.
-l *Location*	Defines the printer's location. This is a string identifying where a printer is physically located, for example "Building X, Room 6". It can be searched on by the **dslpsearch** command. Once set, this value can only be overwritten, not removed. This flag requires the **-P** flag.

Item	Description
-L *PDL*[, *PDL*, ...]	Specifies the list of Page Description Languages (PDLs) supported by the printer. This is used to advertise any PDL the printer supports, and can be searched on, using the **dslpsearch** command. The **AUTOSW**, **PCL**, **PCLXL**, **POSTSCRIPT**, **TEXT**, **ESCP**, **PJL**, **SIMPLE**, and **OTHER** PDLs are supported. If the **-L** flag is used to modify an existing physical printer object, the list replaces the existing list. This flag requires the **-P** flag.
-m [**standard** | **PS**]	Model interface program for the printer when accessed through the specified print queue. It selects the model interface script to be used by the print queue. When a physical printer object is being created, and neither the **-m** nor the **-i** flag has been specified, the default is **standard**. This flag cannot be used when **-i** has also been specified. This flag requires both the **-q** and the **-P** flags.
-n *LocalQueueName*	Defines the local name of a print queue. This name normally only differs from the queue's RDN when the queue is on a non-directory-enabled host. It is used by incoming remote network connections to identify the print queue on the receiving system. The default value is the print queue's RDN. This flag requires the **-q** flag.
-o [**banner** | **nobanner**]	Defines if a banner page will always be produced by this print queue. The default value, **banner**, forces a banner page to be printed for all print requests, whereas **nobanner** allows the user to submit a print job specifying that no banner page is to be printed. This flag requires the **-q** flag.
-o *PrintOption*=*Value*[, ...]	Specifies values for print options. See the **lpadmin** documentation for a detailed description of the print options available with the **-o** flag. This flag requires both the **-q** and the **-P** flags.
-P *PhysicalPrinterName*	Create or modify a physical printer object. The *PhysicalPrinterName* argument specifies the RDN of a printer object. If the object does not already exist, **dslpadmin** creates it.
-q *PrintQueueName*	**dlspadmin** Creates or modifies a print queue object. The *PrintQueueName* argument specifies the RDN of a print queue object. When adding a new print queue, you must specify the **-s** and **-P** flags so the command knows the *NetworkEntityName* and *PhysicalPrinterName* for the print queue being added. If the print queue object does not exist, **dslpadmin** creates it. A command line can contain any combinations of the **-q**, **-P** and **-s**flags , or any combination of the **-x**, **-X** and **-r** flags, but only one of each flag. When multiple directory objects are simultaneously created or modified, appropriate links are set up between the three object types (printers, print queues and network entities).
-r *NetworkEntityName*	Delete the network entity system object. Care needs to be taken not to delete a non-printer system object. It is the responsibility of the administrator to ensure that the correct object is deleted.
-s *NetworkEntityName*	Specifies the network entity system object that hosts the print queue. If **-a** is also given, the object is created or modified. The *NetworkEntityName* argument specifies the RDN of an object in the current directory context. The network entity object defines the network address that remote clients need to use to access the print queue.

Item	Description
-t [**BSD** | **HPNP**]	Defines the print protocol used by this "networked printer" print queue. Retry and timeout values are set to their default values for a networked printer. To change these values, the **dslpprotocol** command should be used. Note that this flag should only be used for networked printers supporting the BSD or HPNP protocol. This flag requires the **-q** flag.
-T *PrinterType*[, *PrinterType*, ...]	List of printer types. It identifies the printer as being of one or more printer types, for example "hplaserjet". See the **lpadmin** manual page for details. This flag requires the **-P** flag.
-u *PhysicalPrinterName*	Unlinks the named physical printer from the print queue (specified with the **-q** flag) without deleting its object. This flag requires the **-q** flag.
-U *ObjectRDN*	Unlinks either the physical printer or the print queue object (specified by *ObjectRDN*) from the print queue (specified with the **-q** flag), without deleting its object. This flag requires the **-q** flag.
-x *PrintQueueName*	Delete a print queue object.
-X *PhysicalPrinterName*	Delete a physical printer object.

Exit Status

0
Indicates success

255 (or -1)
Indicates an error in configuration. Error messages are displayed to explain the error or failure.

Examples

The following examples illustrate use of the dslpadmin command, when the user is logged on to a directory-enabled UNIX system.

1. The following adds an HP LaserJet network printer that uses the BSD remote print protocol, with a print queue RDN of "denlj5n", and a physical printer RDN of "denplj5n". It gives the print queue a description of "HP JetDirect (PostScript)", the printer type "PS-b", and the model interface script as "PS". The printer has a network address of "p_hplj.ibm.com":

   ```
   dslpadmin -q denlj5n -P denplj5n -T PS-b -D "HP JetDirect (Postscript)" \
   -I PS -m PS -A mail -o nobanner  -s denslj5n -a p_hplj.ibm.com -t BSD
   ```

   The print system will allow print requests of content type PS for this print queue, and allow disabling of banner pages.

2. The following adds an HP LaserJet PostScript network printer, using the HPNP remote print protocol, with a print queue RDN of "dehpnp", and a physical printer RDN of "dephpnp". It gives the print queue a description of "HPNP (PCL)", the printer type "hplaserjet", and the model interface script as "standard". The printer has a network address of "p_hplj.ibm.com":

   ```
   dslpadmin -q dehpnp -P dephpnp -T hplaserjet -D "HPNP (PCL)" -I pcl \
   -m standard -A mail -s deshpnp -a p_hplj.ibm.com -t HPNP
   ```

   The print system will allow print requests of content type PCL for this print queue, and reject requests if no banner page is requested. If a printer fault occurs, the print system will mail the owner of the printer.

3. The following deletes an HP LaserJet PostScript printer:

   ```
   dslpadmin -x delj5n  -X deplj5n
   ```

4. The following deletes an HPNP printer:

```
dslpadmin -x dehpnp  -X dephpnp -r deshpnp
```

dslpdisable Command

Purpose

Disable print queue requests for a System V print subsystem.

Syntax

dslpdisable [**-r** *Reason*] *PrintQueueName*

Description

The **dslpenable** and **dslpdisable** commands are used to enable or disable a print queue from processing print requests that have been queued for it. Unlike the **enable** and **disable** commands, the directory-enabled commands can control remote print systems, so long as they are directory-enabled. This is because they write directly to the print queue object on the directory server.

Flags

Item	Description
`-r` *Reason*	Assign the reason for disabling the print queue. Strings containing whitespace should be double quoted. *Reason* is a string that is displayed by the lpstat command. No default reason is set when one is not specified.

Parameters

Item	Description
PrintQueueName	The *PrintQueueName* parameter is the RDN of the print queue. This could be a list of print queues. If the print queue name does not exist in the directory context, the command fails.

Exit Status

0
Indicates success.

1
Indicates invalid options.

2
Indicates that the specified print queue is unknown.

3
Indicates that this user does not have modify permissions.

4
Indicates that an invalid RDN was supplied.

5
Indicates that the value is already set.

6
Indicates that the command is unable to contact the directory service

7
Indicates any other error.

Example

To disable print queue "printer1", specifying the reason "routine maintenance", enter the following:

```
dslpdisable -r "routine maintenance" printer1
```

dslpenable Command

Purpose

Enable print queue requests for a System V print subsystem.

Syntax

dslpenable *PrintQueueName*

Description

The **dslpenable** and **dslpdisable** commands are used to enable or disable a print queue from processing print requests that have been queued for it. Unlike the **enable** and **disable** commands, the directory-enabled commands can control remote print systems, so long as they are directory-enabled. This is because they write directly to the print queue object on the directory server.

Parameters

Item	Description
PrintQueueName	The *PrintQueueName* parameter is the RDN of the print queue. This could be a list of print queues. If the print queue name does not exist in the directory context, the command fails.

Subcommands

Exit Status

0
Indicates success.

1
Indicates invalid options.

2
Indicates that the specified print queue is unknown.

3
Indicates that this user does not have modify permissions.

4
Indicates that an invalid RDN was supplied.

5
Indicates that the value is already set.

6
Indicates that the command is unable to contact the directory service

7
Indicates any other error.

Examples

1. To enable print queue "hpcolor", enter the following:

```
dslpenable hpcolor
```

dslpprotocol Command

Purpose

Configure the remote print protocol of print queue for a System V print subsystem.

Syntax

dslpprotocol **-t** *RemoteProtocol* [**-T** *TimeOut*] [**-R** *Retry*] [**-r**] *PrintQueueName*

dslpprotocol **-l** [**-S**] *PrintQueueName*

Description

The **dslpprotocol** command is used to configure the "remote print protocol" that a remote print client can use when sending print requests to a print queue.

In directory-enabled printing, to print to a remote print queue, the client must first get the remote print protocol it can use. This is obtained from the print queue object in the directory. This can be one or both of BSD and HPNP. Where more than one protocol is configured for a print queue, the UNIX print system uses the first value it reads, so a queue will normally only have a single protocol configured.

The *PrintQueueName* parameter is the Relative Distinguished Name (RDN) of the print queue. If the value assigned to *PrintQueueName* does not exist, the command fails.

The user of this command must be directory-enabled and have permissions set for write, modify, search and read on the directory, in the directory context in which they are administrator.

Flags

Item	Description
-l	Print out a description of the remote print protocol parameters associated with the print queue.
-t *RemoteProtocol*	Specifies the remote print protocol that can be used when sending print requests to this print queue. The protocol type values supported are **bsd** and **hpnp**. The default value is **bsd**.
-T *TimeOut*	Set the network connection timeout value for the specified protocol, that is, the time a network connection should stay alive in an idle condition before disconnection. The value n can also be specified in order to disable timing out. The value 0 causes the connection to be dropped as soon as it becomes idle. The default value is 10 minutes, and there is no practical upper limit. See the **lpsystem** manual page for a full definition of the **-T** option.
-r	This option is used to remove a specified protocol from the print queue object. This option requires that the **-t** option also be specified.

Item	Description
-R *Retry*	Set the network connection retry time for the specified protocol, that is, the time in minutes to wait before trying to re-establish the network connection after a failure. The default value is 2 minutes. A value of 0 causes the connection to be retried immediately. Note that this value must be shorter than the timeout value specified using the **-T** option. The value n can also be specified in order to prevent dropped connections being retried when no work is available. There is no practical upper limit on the value. For "networked printers", the retry time should be set to 0. See the **lpsystem** manual page for a full definition of the **-R** option.
-S	Used with the **-l** option to display the print queue's protocol setup in a simple format.

Parameters

Item	Description
PrintQueueName	The *PrintQueueName* parameter is the Relative Distinguished Name (RDN) of the print queue. If the value assigned to *PrintQueueName* does not exist, the command fails.

Exit Status

0
Indicates success.

1
Indicates invalid options.

2
Indicates that the specified print queue is unknown.

3
Indicates that this user does not have modify permissions.

4
Indicates that an invalid RDN was supplied.

5
Indicates that the value is already set.

6
Indicates any other error.

Examples

1. To set print queue "printq1" to allow the BSD remote print protocol, enter the following:

```
dslpprotocol -t BSD printq1
```

2. To remove the BSD protocol from print queue "hpcolor", enter the following:

```
dslpprotocol -r -t BSD hpcolor
```

dslpreject Command

Purpose

Reject print queue requests for directory-enabled System V print systems.

Syntax

dslpreject [**-r** *Reason*] *PrintQueueName*

Description

The **dslpaccept** and **dslpreject** commands are used to set a print queue so that it will accept or reject print requests being queued for it. Unlike the **accept** and **reject** commands, the directory-enabled commands can control remote print systems, so long as they are directory-enabled. This is because they write directly to the print queue object on the directory server. Print requests that are already queued are not affected by the **dslpreject** command.

The user of this command must be directory-enabled and have permissions set for write, modify, search and read on the directory, in the directory context in which the user is administrator.

Flags

Item	Description
-r *Reason*	Assigns a reason for the rejection. Strings containing whitespace should be within double quotes. *Reason* is a string that is displayed by the **lpstat** command. No default reason is set when one is not specified.

Parameters

Item	Description
PrintQueueName	The *PrintQueueName* parameter is the RDN of the print queue object. Multiple print queue names may be specified in a comma-separated list.

Exit Status

0
Indicates success.

1
Indicates invalid options.

2
Indicates that the specified print queue is unknown.

3
Indicates that this user does not have modify permissions.

4
Indicates that an invalid RDN was supplied.

5
Indicates that the value is already set.

6
Indicates that the command is unable to contact the directory service

7
Indicates any other error.

Examples

1. To set a print queue to reject requests and specify the reason that there is no toner, enter the following:

```
dslpreject -r "no toner" printer1
```

dslpsearch Command

Purpose

Search directory for print system objects on a System V print subsystem.

Syntax

dslpsearch [**-q** [**-p**]] | [**-P**] [**-o** *SearchOptions*]

Description

The **dslpsearch** command allows users and administrators to search the directory for print system objects. For example, a user could search for any printer that can print color PostScript files. The main use of this command will be to search for print queues that match the search string.

The **dslpsearch**command returns the Distinguished Name (DN) of any objects that match the search string. However, the Relative Distinguished Name (RDN) is required for use in the other directory-enabled commands. For example, if the DN "`cn=testqueue,ou=printq,ou=print,cn=aixdata`" is returned by the **dslpsearch** command, only the RDN "`testqueue`" is used to refer to the print queue.

Flags

Item	Description
-q	Search for print queues that match the search options. The search is done on the physical printer objects but the print queues that service those printers are displayed. This is the default search type. The **-q** option cannot be specified with **-P**.
-p	This option is used with the **-q** option, and causes a list of physical printers servicing the print queue also to be displayed.
-P	Search for physical printers that match the search string. The **-P** option cannot be specified with **-q**.
-o *SearchOptions*	Multiple search options may form a comma-separated list. Each option may be constructed from the following: • one or more of the following Page Description Languages (PDLs): **AUTOSW**, **PCL**, **PCLXL**, **POSTSCRIPT**, **TEXT**, **ESCP**, **PJL**, **SIMPLE**, **OTHER** • any of the following printer facilities: **COLOR**, **DUPLEX**, **TRAYS**, **FINISH** • one or more physical printer locations, specified by `location=xxxxxxxx` or `location='aaaa bbbbb'` • The string value defined by location= is searched on with wildcards placed at both ends of the string, so `location=Room1` would find any printer with "Room1" in its location, such as "Building X, Room1, Bay6". The string value can also have wildcards (*) embedded in it, for example `location="Building X*Bay6"`. Multiple location values are OR'd in the search. • The following are valid command lines containing search strings: `dslpsearch -q -o PCL,ESCP,location=room2,COLOR` `dslpsearch -q -p -o "PS, location='Building 1, Room1', DUPLEX"`

Exit Status

0
Indicates success.

1
Indicates invalid options.

2
Indicates that the search on the directory tree failed.

3
Indicates invalid directory context.

4
Indicates the command is unable to contact the directory service.

Examples

1. The following command line searches for any print queues that match the search options:

```
dslpsearch -q -o search-options
```

2. The following searches for any physical printers that match the search options:

```
dslpsearch -P -o search-options
```

dspcat Command

Purpose

Displays all or part of a message catalog.

Syntax

To Display Messages in a Catalog

dspcat *CatalogName* [*SetNumber* [*MessageNumber*]]

To Format Output for the gencat Command

dspcat -g *CatalogName* [*SetNumber*]

Description

The **dspcat** command displays a particular message, all the messages in a set, or all the messages in a catalog. The **dspcat** command directs the messages to standard output.

Note: The **dspcat** command looks for the catalog files under the **NLSPATH** environment variable if the **LC__FASTMSG** attribute is set to `False` in C or POSIX locale environment.

The **LC__FASTMSG** attribute specifies that default messages are used for the C and POSIX locales and that the **NLSPATH** environment variable is ignored when the **LC__FASTMSG** attribute is set to `True`.

The default value for the **LC__FASTMSG** attribute is `True` in the `/etc/environment` path.

The *CatalogName* parameter specifies a message catalog. The *SetNumber* parameter specifies a set in the catalog specified by the *CatalogName* parameter. The *MessageNumber* parameter specifies a particular message in the set specified by the *SetNumber* parameter. If you include all three parameters, the **dspcat** command displays the specified message. If you do not include the *MessageNumber* parameter, the **dspcat** command displays all the messages in the set. If you specify a nonexistent value for the *SetNumber* or *MessageNumber* parameter, the **dspcat** command displays an error message and returns a nonzero return value. If you specify only the *CatalogName* parameter, the **dspcat** command displays all the messages in the catalog. You must include the *SetNumber* parameter if you include the *MessageNumber* parameter.

The **dspcat** command uses the **NLSPATH** environment variable and the **LC_MESSAGES** category to find the specified message catalog if you do not use / (slash) characters in the value of the *CatalogName* parameter.

Flags

Item	Description
-g	Formats output to be used as input to the **gencat** command. The *MessageNumber* parameter is not valid when you use the **-g** flag.

Examples

To display message number 2 in set number 1 of the `test.cat` file, enter:

```
dspcat test.cat 1 2
```

Files

Item	Description
/usr/bin/dspcat	Contains the **dspcat** command.

dspmsg Command

Purpose

Displays a selected message from a message catalog.

Syntax

dspmsg [**-s** *SetNumber*] *CatalogName MessageNumber* [**'***DefaultMessage***'** [*Arguments*]]

Description

The **dspmsg** command displays either the text of a particular message from a message catalog generated with the **gencat** command or, if the message cannot be retrieved, a default message supplied as a parameter to the command. The **dspmsg** command directs the message to standard output. This command is intended for use in shell scripts as a replacement for the **echo** command.

Note: The **dspmsg** command looks for the catalog files under the **NLSPATH** if the **LC_FASTMSG** is set to False in C or POSIX locale environment.

LC__FASTMSG specifies that default messages are used for the C and POSIX locales and that **NLSPATH** is ignored when **LC__FASTMSG** is set to True.

The default value for **LC__FASTMSG** will be True in **/etc/environment**.

The **NLSPATH** environment variable and the **LC_MESSAGES** category are used to find the specified message catalog if no / (slash) characters are used in the value of the *CatalogName* parameter. If the catalog named by the *CatalogName* parameter is not found or if the message named by the *MessageNumber* parameter (and optional *SetNumber* value) is not found, then the supplied *DefaultMessage* value is displayed. If a *DefaultMessage* value is not specified, a system-generated error message is displayed.

The **dspmsg** command allows up to ten string arguments to be substituted into the message if it contains the **%s**, %n$s, **%ld**, or %n$ld **printf** subroutine conversion specification. Missing arguments for

conversion specifications result in a **dspmsg** error message. Normal **printf** subroutine control character escapes (for example, **\n**) are recognized.

The use of **printf** subroutine format strings is recommended in the catalog. This format provides for correct insertion of arguments even if the format strings in the message are in a different order than the default message. You must enclose the default message in single quotation marks if using the **%n$s** notation for message inserts.

Flags

Item	Description
-s *SetNumber*	Specifies an optional set number. The default value for the *SetNumber* variable is 1.

Examples

To display set number 1, message number 2 of the `test.cat` catalog, enter:

```
dspmsg    -s    1    test.cat    2    'message    %s    not    found'    2
```

If the message is not found, `message 2 not found` is displayed.

Files

Item	Description
/usr/bin/dspmsg	Contains the **dspmsg** command.

dtaction Command

Purpose

Invokes a CDE action with specified arguments.

Syntax

`dtaction` [`-contextDir` *context_dir*]\

[`-execHost` *host_name*] [`-termOpts` *terminal_arguments*]

[`-user` *user_name*] *action_name*

[*action_arg*] ...

Description

The `dtaction` command allows applications or shell scripts, which are otherwise not connected into the CDE development environment, to invoke action requests.

The action called *action_name* is called with the *action_arg* provided on the command line.

A single *action_name* is required; the user can provide any number of *action_args*.

Interpretation of the *action_name* and *action_args* depends on the definition of the action in the action database.

The action might be defined in one of the system action database files, or in one of the user's private action database files.

The *action_args* are absolute or relative path names of files. The `dtaction` command passes this list of files on to the specified action.

Error dialogs are posted when the following conditions are detected:

- Desktop environment could not be initialized
- Invalid user or password
- Unable to change ID to the requested user
- No action name specified

Flags

Item	Description
`contextDir` *context_dir*	Specifies a default directory context if the definition of *action_name* does not define a current working directory for command actions.
`execHost` *host_name*	Specifies an alternative execution host, *host_name*, for a command action. If the action is not a command action, the `dtaction` command ignores this option. The action is attempted on *host_name* instead of the hosts specified in the action's EXEC_HOST specification. An error is posted if it is not possible to invoke the specified action on any eligible host.
`termOpts` *terminal_arguments*	Specifies arguments intended for the terminal emulator that is provided for command actions that are not of type NO_STDIO. If there are white-space characters in the *terminal_arguments* string, that string must be quoted to protect it from the shell. These arguments are passed unchanged to the terminal emulator, so the user must ensure that the strings are reasonable. In particular, *terminal_arguments* does not allow the argument that specifies the command to be run in a terminal emulator window (that is, using `dtterm1` with the `-e` flag).
`user` *user_name*	Specifies a user name. If `dtaction` is not currently running as that user, a prompt dialog collects the specified user password or the root user password. After a valid password is entered, the `dtaction` command changes so that it is running as the requested user and then starts the requested action.

Parameters

Item	Description
action_name	Specifies the name of the action to be invoked.
action_arg	Specifies the absolute or relative file names of files.

Environment Variables

Item	Description
`DTDATABASESEARCHPATH`	A comma-separated list of directories (with optional host: prefix) that tells the action service where to find the action databases.

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

The dtaction command is an application enabled by PAM with service name dtaction.

If the user name specified by user *user_name* option is different from the login user name,

The dtaction command authenticates the user before invoking the specified action. It is capable of performing PAM authentication as well as traditional authentication.

To use PAM for authentication system-wide, establish root user permissions and modify the value of the *auth_type* attribute in the usw stanza of the /etc/security/login.cfg file toPAM_AUTH.

The authentication mechanisms used when PAM is enabled depend on the configuration for the login service in /etc/pam.conf.

The dtaction command requires an /etc/pam.conf entry for the auth module type.

The following configuration is recommended in /etc/pam.conf for the dtaction service:

```
dtaction        auth            required        /usr/lib/security/pam_aix
```

Examples

1. To invoke an action, enter:

   ```
   dtaction Xterm
   ```

   This launches X Windows terminal emulator (Xterm).
2. To invoke an action on a remote host, enter:

   ```
   dtaction -execHost hostname Xterm
   ```

   This executes Xterm on the specified remote host.
3. To invoke an action as a different user, enter:

   ```
   dtaction -user username Xterm
   ```

   This executes Xterm as the specified user.

Location

/usr/dt/bin/dtaction

Standard Error

The dtaction command writes diagnostic error messages to standard error, which is redirected to $HOME/.dt/errorlog.

Files

Item	Description
/etc/pam.conf	Determines PAM authentication mechanisms.

Item	Description
`/etc/security/login.cfg`	Determines PAM authentication system-wide.

dtappintegrate Command

Purpose

The Common Desktop Environment application integration tool.

Syntax

dtappintegrate -s *ApplicationRoot* [**-t** *TargetPath*] [**-l** *Language*] [**-u**]

Description

The **dtappintegrate** command links the application CDE configuration files from application-specific locations to system locations and updates the system's Browser help volumes for the languages affected. The **dtappintegrate** command is used during the installation process of an application. The application installation script should invoke the **dtappintegrate** command at the end.

There are four key subdirectories under the application root (referred to as **$APP_ROOT**) dictated by CDE policy. The directories are:

Item	Description
$APP_ROOT/dt/appconfig/types/ *Language*	For filetype, Front Panel, and action files.
$APP_ROOT/dt/appconfig/appmanager/ *Language*	For application group files.
$APP_ROOT/dt/appconfig/icons/ *Language*	For icons used by the CDE managers.
$APP_ROOT/dt/appconfig/help/ *Language*	For application help. For example, the default-language application **SpreadSheet** would load its desktop icons under: **/opt/SpreadSheet/dt/appconfig/icons/C/*.bm** and **/opt/SpreadSheet/dt/appconfig/icons/C/*.pm**, where **/opt/SpreadSheet** is the value of **$APP_ROOT**.

Note: $APP_ROOT is a syntactical convention of this document and is not used by the runtime environment.) All of these CDE configuration files and subdirectories are placed under a common top and should always include the default language subdirectory **C**.

In the simplest case, the command takes as input the application root, for example, **/opt/thisapp**. The outputs from this operation are corresponding subdirectories and files on the application server that contain relative symbolic links to the applications CDE configuration files described above, under the following system locations:

Item	Description
/etc/dt/appconfig	Top-level application configuration subdirectory, consists of following subdirectories: **/etc/dt/appconfig/types/*Language*/** Contains the ***.dt** and any ***.fp** links. **/etc/dt/appconfig/appmanager/*Language*/** Contains links to the application group subdirectory and the action script files to appear as actions under the Application Manager. **/etc/dt/appconfig/help/*Language*/** Contains symbolic links to the help files installed under the application's root. **/etc/dt/appconfig/icons/*Language*/** Contains symbolic links to the CDE icons for the application.

Flags

Item	Description
-s *ApplicationRoot*	Integrates the application located at *ApplicationRoot*. This flag is required.
-t *TargetPath*	Links the application CDE configuration files from the application-specific location to *TargetPath* rather than to the system locations. This flag is optional. If the **-t** flag is supplied, the files are linked under the specified subdirectory. For example, specifying **-t /etc/dt/private** would cause the application help files to be symbolically linked under **/etc/dt/private/help/***Language*. This flag is typically used only by system administrators who want to create separate applications and not by the application post-installation script. By default (with no **-t** specified), the application subdirectory root is global to the application host. All applications installed on the host will have their configuration files copied to the same place for merging with other application configuration files.
-l *Language*	Specifies the language to integrate. Basically, this flag indicates the directories under which to find the application CDE configuration files. If this parameter is not specified, all languages will be integrated. This parameter is optional.
-u	Integration of application is canceled. This flag is optional.

dtlogin Command

Purpose

Performs a CDE login service.

Syntax

`dtlogin` [`-config` *configuration_file*] [`-daemon`] [`-debug` *debug_level*] [`-error` *error_log_file*] [`-nodaemon`] [`-resources` *resource_file*] [`-server` *server_entry*] [`-session` *session_program*] [`-udpPort` *port_number*]

Description

The dtlogin command supports the following key tasks:

- Launching dtgreet login screen for explicitly managed local and remote displays and XDMCP-managed remote displays.
- Accessing traditional terminal (character) login from GUI login screen
- Authenticating and logging in system-dependent users
- Launching the selected session

The dtlogin command provides services similar to those provided by init, getty, and login on character terminals, which include prompting for login and password, authenticating the user, and running a session. A *session* is defined by the lifetime of a particular process. In the traditional character-based terminal world, a session is the user's login shell process; in the DT context, it is the DT Session Manager. If the DT Session Manager is not used, the typical substitute is either a window manager with an exit option, or a terminal emulator running a shell, where the lifetime of the terminal emulator is the lifetime of the shell process that it is running. This reduces the X session to an emulation of the character-based terminal session. When the session is terminated, dtlogin resets the X server and (optionally) restarts the whole process.

The dtlogin command supports management of remote displays using the X Display Manager Control Protocol, Version 1.0. (XDMCP). When dtlogin receives an indirect query from XDMCP, it can run a chooser process to perform an XDMCP BroadcastQuery (or an XDMCP Query to specified hosts) on behalf of the display and offer a menu of possible hosts that offer XDMCP display management. This feature is useful with X terminals that do not offer a host menu.

Because dtlogin provides the first interface that users see, it is designed to be simple to use and easy to customize according to the needs of a particular site.

Login Window

The Login window allows users to enter a user ID and password, select a startup session, and select a startup locale. Users can also reset the X server or temporarily suspend the X server to access the character login prompt.

The contents of the Login window are as follows:

login field
Provides an entry field in which users enter their IDs.

password field
Provides an entry field in which users enter their passwords (no-echo).

OK button
Authenticates a user and launches a session.

Clear button
Clears login and password fields.

Options
Lets users select a locale name and login session type. It also lets users restart the X server or switch to a character login prompt (for local displays). The contents of the Options menu are as follows:

Languages
Displays the Languages menu. Selecting the language from the login screen Options menu immediately localizes the login screen and sets the LANG variable for the next session. Login screen localization and LANG return to the default value upon conclusion of the session. The contents of this menu can vary depending upon the locales installed on the system. They can be overridden by using the languageList resource. The default locale of C can be overridden using the language resource. The system or languageList locales specified are displayed as menu items in the Languages menu. Alternate text to be displayed can be specified for a given locale name by using the languageName resource.

No-windows
Displays character login prompt (local displays only).

Reload Login
Restarts the X Server and returns to login screen.

Resources
Lists resources to be used.

Sessions
Displays Sessions menu. Allows users to select which session type should be started upon login. Menu items include the following:

DT Session
Starts a regular desktop session (Xsession).

Fail-safe Session
Starts a fail-safe session (Xfailsafe).

Help
Displays help messages.

Controlling the Server

The `dtlogin` command controls local servers using POSIX signals. The SIGHUP signal is expected to reset the server, closing all client connections and performing other clean up duties. The SIGTERM signal is expected to terminate the server. If these signals do not perform the expected actions, the `resetSignal` and `termSignal` resources can specify alternate signals.

To control remote servers that are not using XDMCP, `dtlogin` searches the window hierarchy on the display and uses the KillClient X protocol request in an attempt to clean up the terminal for the next session. This might not actually kill all of the clients, because only those that have created windows are noticed. XDMCP provides a more sure mechanism; when `dtlogin` closes its initial connection, the session is over and the terminal is required to close all other connections.

Controlling dtlogin

The `dtlogin` command responds to two signals: SIGHUP and SIGTERM. When it is sent a SIGHUP, `dtlogin` rereads the configuration file and the file specified by the servers resource, and determines whether entries have been added or removed. If a new entry has been added, `dtlogin` starts a session on the associated display. Entries that have been removed are disabled immediately, meaning that any session in progress is terminated without notice, and no new session is started. When sent a SIGTERM, `dtlogin` terminates all sessions in progress and exits. This can be used when shutting down the system.

Internationalization

All labels and messages are localizable. The `dtlogin.cat` message catalog contains the localized representations of the default labels and messages. The `dtlogin` command reads the appropriate message catalog indicated by the LANG environment variable and displays the localized strings. An option on the authentication screen allows the user to override the default language for the subsequent session. If the authentication screen has been localized for the selected language, the screen is redisplayed in that language; otherwise, it is displayed in the default language. In either case, the LANG environment variable is set appropriately for the resulting session.

The resource language is available in the dtlogin configuration file to change the default language for a display. The `languageList` resource is available in the `dtlogin` configuration file to override the default set of languages displayed on the authentication screen. The `languageName` resource is available to provide a mapping from locale names to the text displayed on the Language menu.

Authentication and Auditing

The `dtlogin` command is a login service enabled by PAM with service name `dtlogin`. The `dtlogin` client supports PAM authentication in addition to traditional local UNIX login and auditing. Additional authentication or auditing functions, such as Kerberos or B1 can be added by individual vendors.

To use PAM for system-wide authentication, establish root user permissions and modify the value of the *auth_type* attribute in the `usw` stanza of the `/etc/security/login.cfg` file to PAM_AUTH.

The authentication mechanisms used when PAM is enabled depend on the configuration for the login service in `/etc/pam.conf`. The `dtlogin` command requires an `/etc/pam.conf` entry for the `auth`,

account, password, and session module types. The following configuration is recommended in /etc/pam.conf for the dtlogin service:

```
dtlogin        auth              required       /usr/lib/security/pam_aix
dtlogin        account           required       /usr/lib/security/pam_aix
dtlogin        password          required       /usr/lib/security/pam_aix
dtlogin        session           required       /usr/lib/security/pam_aix
```

X Server Security

The X server provides both user-based and host-based access control. By default, dtlogin uses user-based access control to the X server (MIT-MAGIC-COOKIE-1). This level of security allows access control on a per-user basis. It is based on a scheme where if a client passes authorization data that matches what the server has, the client is allowed access. When a user logs in, this authorization data is by default stored and protected in the $HOME/.Xauthority file.

However, using host-based access control mechanisms might be preferable in environments with unsecure networks, because user-based access control allows any host to connect if the host has discovered the private key. Another drawback to user-based access control is that R2 or R3 clients are unable to connect to the server.

The authorize resource controls whether user-based or host-based access control is used by dtlogin. See the xhost, and xauth commands for more information.

Resources

The dtlogin command is controlled by the contents of the dtlogin configuration file, which defaults to /usr/dt/config/Xconfig. Some resources control the behavior of dtlogin in general, and others can be specified for a particular display.

General Resources

The following dtlogin general resources are not display-specific and apply to all displays where appropriate.

Item	Description
accessFile	**Class:** AccessFile **ClassType:** String **Default:** Null **Description:** To prevent unauthorized XDMCP service and to allow forwarding of XDMCP IndirectQuery requests, this file contains a database of host names that are either allowed direct access to this machine or have a list of hosts to which queries should be forwarded to. Refer to the Xaccess file section for a description of the format. If this resource is not set, all hosts will be allowed XDMCP service.
authDir	**Class:** AuthDir **ClassType:** String **Default:** /var/dt **Description:** The directory name that dtlogin uses to temporarily store authorization files for displays using XDMCP.

Item	Description
autoRescan	**Class:** AutoRescan **ClassType:** Boolean **Default:** True **Description:** Controls whether dtlogin rescans the configuration file and server file after a session terminates and the files have changed. You can force dtlogin to reread these files by sending a SIGHUP signal to the main process.
daemonMode	**Class:** DaemonMode **ClassType:** Boolean **Default:** False **Description:** The dtlogin command can make itself into an unassociated daemon process. This is accomplished by forking and leaving the parent process to exit, then closing file descriptors and releasing the controlling terminal. This is inconvenient when attempting to debug dtlogin. Setting this resource to False disables daemonMode.
debugLevel	**Class:** DebugLevel **ClassType:** Int **Default:** 0 **Description:** A nonzero value specified for this integer resource enables debugging information to be printed. It also disables daemon mode, which redirects the information into the normally unuseful bit-bucket.
errorLogFile	**Class:** ErrorLogFile **ClassType:** String **Default:** NULL **Description:** Error output is normally directed at the system console. To redirect it, set this resource to any file name. This file contains any output directed to stderr by Xsetup, Xstartup, and Xreset.

Item	Description
errorLogSize	**Class:** errorLogSize **ClassType:** Int **Default:** 50 **Description:** This resource specifies the maximum size of the error log file in kilobytes. When the limit is reached, dtlogin deletes the oldest entries in the file until the file size is reduced to 75 percent of the maximum. After the file is truncated, any user who is accessing the error log file (for example, using cat or tail) will need to close the file and reopen it for access in order to see subsequent information that is logged to the file.
exportList	**Class:** ExportList **ClassType:** String **Default:** NULL **Description:** Contain a set of variable names separated by a space or tab. Each variable named is obtained from the dtlogin environment and loaded into the environment of the server and session. See the Environment section for details.
fontPathHead	**Class:** FontPathHead **ClassType:** String **Default:** NULL **Description:** Value that is prepended to the default X server font path.
fontPathTail	**Class:** fontPathTail **ClassType:** String **Default:** NULL **Description:** Value that is appended to the default X server font path.

Item	Description
keyFile	**Class:** KeyFile **ClassType:** String **Default:** /usr/dt/config/Xkeys **Description:** XDM-AUTHENTICATION-1 style XDMCP authentication requires that a private key be shared between dtlogin and the terminal. This resource specifies the file containing those values. Each entry in the file consists of a display name and the shared key. By default, dtlogin does not include support for XDM-AUTHENTICATION-1 because it requires DES, which is not generally distributable.
lockPidFile	**Class:** LockPidFile **ClassType:** Boolean **Default:** True **Description:** Controls whether dtlogin uses file locking to prevent multiple instances of dtlogin from executing concurrently.
networkDevice	**Class:** NetworkDevice **ClassType:** String **Default:** /dev/dtremote **Description:** For remote connections, the value for line in /etc/utmp must also exist as a device in the /dev directory for commands such as finger to operate properly. This resource specifies the path name of the /dev file dtlogin creates when a remote display connects. For most platforms, the file is created as a symbolic link to /dev/null. The specified value must start with /dev/, or else the value is discarded and no file is created.
pidFile	**Class:** PidFile **ClassType:** STring **Default:** NULL **Description:** The filename specified is created to contain an ASCII representation of the process-ID of the main dtlogin process. This can be used when sending signals to dtlogin. The dtlogin client also uses file locking to attempt to prevent more than one dtlogin from running on the same machine. See the lockPidFile resource for more information.

Item	Description
removeDomainname	**Class:** RemoveDomainname **ClassType:** Boolean **Default:** True **Description:** When computing the display name for XDMCP clients, dtlogin typically creates a fully qualified host name for the terminal. Because this is sometimes confusing, dtlogin removes the domain name portion of the host name if it is the same as the domain name for the local host when this variable is set.
requestPort	**Class:** RequestPort **ClassType:** int **Default:** 177 **Description:** Indicates the UDP port number that dtlogin uses to listen for incoming XDMCP requests. Unless the system needs to be debugged the system, the default value for this resource should remain.

Item	Description
servers	**Class:** Servers **ClassType:** String **Default:** :0 Local local /system_dependent_path/X :0 **Description:** Either specifies a file name full of server entries, one per line (if the value starts with a slash), or a single server entry. Each entry indicates a display that should be managed constantly and that is not using XDMCP. The general syntax for each entry is as follows: *DisplayName DisplayClass DisplayType*[@ite] [Command [options]] where: ***DisplayName*** A value that can be passed in the `-display` option to any X program. This string is used in the display-specific resources to specify the particular display, so caution must be taken to match the names. For example, use `:0 local /usr/bin/X11/X :0` instead of `localhost:0 local /usr/bin/X11/X :0` if your other resources are specified as `Dtlogin._0.session`). A asterisk (*) in this field expands to `hostname:0` by `dtlogin`. ***DisplayClass*** The display class portion is also used in the display-specific resources as the class portion of the resource. This is useful if you have a large collection of similar displays (a group of X terminals, for example) and want to set resources for groups of them. When using XDMCP, the display is required to specify the display class. Refer to your X terminal documentation for information on a reasonably standard display class string for your device. ***DisplayType*** If specified as `local`, indicates that an X server should be started for this entry. A value of `remote` indicates that an existing X server should be attached. **@ite** On local bitmaps, the user can choose a `Command Line Login` option using the login screen, which temporarily suspends the X-server and presents the traditional character `login:` prompt. The user can then log in and perform non-X related tasks. When the user finishes and logs out, the X-server is restarted, and the login screen is redisplayed. In order to support `Command Line Login` mode, the display must have an associated Internal Terminal Emulator (ITE) device. By default, `dtlogin` associates the ITE device "console" (`/dev/console`) with display `:0`. If your configuration does not match this default, specify `@device` for any displays with an associated ITE, and specify `@none` for all other displays listed in the servers file. **Command [options]** The string that starts the X server. The `dtlogin` client will always connect to the X server using the *DisplayName* specified, so you might need to specify an explicit connection number as an option to your X server (`:0` in the preceding example).

Item	Description
sysParmsFile	**Class:** SysParmsFile **ClassType:** String **Default:** /system_dependent_path **Description:** Specifies a file containing shell commands, one of which sets the time zone environment variable (TZ) for the system. If the time zone is set using the shell syntax TZ=, dtlogin can use this information to set the time zone for the user session.
timeZone	**Class:** TimeZone **ClassType:** String **Default:** NULL **Description:** Specifies the local time zone for dtlogin. It is loaded into the environment of dtlogin as the value of the TZ variable and inherited by all subsequent sessions. Some systems maintain a configuration file that contains the time zone setting (for example, /etc/src.sh). See also the sysParmsFile resource.
wakeupInterval	**Class:** WakeupInterval **ClassType:** Int **Default:** 10 **Description:** If the user selects Command Line Login mode from the login screen, dtlogin terminates the X-server and allows the traditional character-based login prompt login: to become visible. If the user does not log in within 2 times the wakeupInterval seconds, the X-server is restarted. After the user has logged in, dtlogin checks every wakeupInterval seconds to see if the user has logged out. If so, the X-server is restarted and the login screen is redisplayed.

Display Resources

The dtlogin command display resources can be specified for all displays or for a particular display. To specify a particular display, the display name is inserted into the resource name between Dtlogin and the final resource name segment. For example, Dtlogin.expo_0.startup is the name of the resource defining the startup shell file on the expo:0 display. The resource manager separates the name of the resource from its value with colons, and separates resource name parts with dots, so dtlogin uses underscores (_) for the dots (.) and colons (:) when generating the resource name.

Resources can also be specified for a class of displays by inserting the class name instead of a display name. A display that is not managed by XDMCP can have its class affiliation specified in the file referenced by the servers resource. A display using XDMCP supplies its class affiliation as part of the XDMCP packet.

The following dtlogin general resources are not display-specific and apply to all displays where appropriate.

Item	Description
authorize	**ClassClass:** Authorize **Type:** Boolean **Default:** False **Description:** Authorize is a Boolean resource that controls whether dtlogin generates and uses authorization for the server connections. Refer also to the authName resource.
authName	**ClassClass:** AuthName **Type:** String **Default:** MIT-MAGIC-COOKIE-1 **Description:** If the authorize resource is used, authName specifies the type of authorization to be used. Currently, dtlogin supports only MIT-MAGIC-COOKIE-1 authorization. XDM-AUTHORIZATION-1 could be supported, but DES is not generally distributable. XDMCP connections state which authorization types are supported dynamically, so authName is ignored in this case. Refer also to the authorize resource.)
authFile	**ClassClass:** AuthFile **Type:** String **Default:** NULL **Description:** Communicates the authorization data from dtlogin to the server, using the -auth server command line option. Keep this resource in a write-protected directory to prevent its erasure, which would disable the authorization mechanism in the server. If NULL, dtlogin generates a file name.
chooser	**ClassClass:** Chooser **Type:** **Default:** **Description:** Specifies the program run to offer a host menu for indirect queries redirected to the special host name CHOOSER. The default is /usr/dt/bin/dtchooser. See the Xaccess file section.

Item	Description
cpp	**ClassClass:** Cpp **Type:** String **Default:** system dep. **Description:** Specifies the path of the C preprocessor that is used by xrdb.
environment	**ClassClass:** Environment **Type:** String **Default:** system dep. **Description:** Contains a set of *name=value* pairs separated by a space or tab. Each item is loaded into the environment of the server and session. See the Environment section for more information.
failsafeClient	**ClassClass:** FailsafeClient **Type:** String **Default:** /system_dep./xterm **Description:** If the default session fails to execute, dtlogin falls back to this program. This program is executed with no arguments, but executes using the same environment variables as the session would have had.
grabServer	**ClassClass:** GrabServer **Type:** Boolean **Default:** True **Description:** To improve security, dtlogin grabs the server and keyboard while reading the name and password. The grabServer resource specifies if the server should be held while the name and password is read. When FALSE, the server is ungrabbed after the keyboard grab succeeds; otherwise, the server is grabbed until just before the session begins.

Item	Description
grabTimeout	**ClassClass:** GrabTimeout **Type:** Int **Default:** 3 seconds **Description:** Specifies the maximum time dtlogin will wait for the grab to succeed. The grab can fail if another client has the server grabbed, or possibly if the network latencies are very high. The grabTimeout resource has a default of 3 seconds; use this resource with care, because a user can be deceived by a look-alike window on the display. If the grab fails, dtlogin kills and restarts the server (if possible) and session. Some X-terminals cannot display their login screens while the server is grabbed. Setting grabServer to FALSE allows the screen to be displayed but opens the possibility that a user's login name can be stolen by copying the contents of the login screen. Because the keyboard is still grabbed and the password is not echoed, the password cannot be stolen.
language	**ClassClass:** Language **Type:** String **Default:** system dep. **Description:** Specifies the default setting for the LANG environment variable. If the dtlogin screen is localized for that language, it is displayed appropriately; otherwise, it is displayed in the C language The user can temporarily override this setting using an option on the login screen. When the subsequent session terminates, the LANG variable reverts to this setting.
languageList	**ClassClass:** LanguageList **Type:** String **Default:** NULL **Description:** Allows the user to override the default set of languages displayed in the Language menu of the login screen. It is useful if the set of languages actually used on a particular display is smaller than the set installed on the system. The resource value is a list of valid values for the LANG environment variable. Language values should be separated by one or more spaces or tabs.

Item	Description
languageName	**ClassClass:** LanguageName **Type:** String **Default:** NULL **Description:** Allows the user to override the default locale name displayed in the Language menu of the login screen with alternate text. This way, instead of users seeing a En_US item, they could see an English (United States) item instead. This resource is specified as Dtlogin*_local_name_.languageName: *text* as follows: `Dtlogin*En_US.languageName: English (United States)` `Dtlogin*Fr_CA.languageName: French (Canadian)`
openDelay	**ClassClass:** OpenDelay **Type:** Int **Default:** 5 seconds **Description:** Specifies the duration (in seconds) between successive attempts to open reluctant servers.
openRepeat	**ClassClass:** OpenRepeat **Type:** Int **Default:** 5 seconds **Description:** Specifies the number of successive attempts to open reluctant servers.
openTimeout	**ClassClass:** OpenTimeout **Type:** Int **Default:** 30 seconds **Description:** Specifies the amount of time to wait while actually attempting to open reluctant servers. This time is the same as the maximum time spent in the connect system call.

Item	Description
pingInterval	**ClassClass:** PingInterval **Type:** Int **Default:** 5 minutes **Description:** To discover when remote displays disappear, dtlogin occasionally pings them, using an X connection and sending XSync requests. The pingInterval resource specifies the time (in minutes) between successive ping attempts.
pingTimeout	**ClassClass:** PingTimeout **Type:** int **Default:** 5 minutes **Description:** Specifies the maximum wait time (in minutes) for the terminal to respond to the request. If the terminal does not respond, the session is terminated. The dtlogin client does not ping local displays. A local session should never be terminated as a result of the server waiting (for remote file system service, for example) and not responding to the ping.
reset	**ClassClass:** Reset **Type:** String **Default:** NULL **Description:** specifies a program that is run (as root) after the session terminates. If this resource is not set, no program is run. The conventional name is Xreset. See the Xreset File.
resetForAuth	**ClassClass:** ResetForAuth **Type:** Boolean **Default:** False **Description:** During the original implementation of authorization in the sample server, the authorization file was reread at server reset time instead of when checking the initial connection. Because dtlogin generates the authorization information just before connecting to the display, an old server does not get current authorization information. This resource causes dtlogin to send SIGHUP to the server after setting up the file, causing an additional server reset to occur, during which time the new authorization information is read.

Item	Description
resetSignal	**ClassClass:** Signal **Type:** Int **Default:** 1 SIGHUP **Description:** Specifies the signal dtlogin sends to reset the server.
resources	**ClassClass:** Resource **Type:** String **Default:** NULL **Description:** Specifies the name of the file to be loaded by xrdb as the resource database onto the root window of screen 0 of the display. This resource database is loaded just before the authentication procedure is started, so it can control the appearance of the login window. See the section on the authentication screen, which describes the various resources that are appropriate to place in this file. There is no default value for this resource, but the conventional name is Xresources.
session	**ClassClass:** Session **Type:** String **Default:** /usr/dt/bin/Xsession **Description:** Specifies the session to be executed for the authenticated user. By default, the /usr/dt/bin/Xsession file is run. The conventional name is Xsession. Refer to the Xsession file.
setup	**ClassClass:** Setup **Type:** String **Default:** NULL **Description:** Specifies a program that is run (as root) prior to the display of the authentication screen. By default, no program is run. The conventional name is Xsetup. Refer to the Xsetup file.

Item	Description
startAttempts	**ClassClass:** StartAttempts **Type:** Int **Default:** 4 **Description:** Four numeric resources control the behavior of dtlogin when attempting to open reluctant servers: openDelay, openRepeat, openTimeout, and startAttempts. This resource specifies the number of times the entire process occurs before giving up on the server. After openRepeat attempts have been made, or if openTimeout seconds elapse in any particular attempt, dtlogin terminates and restarts the server, attempting to connect again. This process is repeated startAttempts time, at which point the display is declared dead and disabled.
startup	**ClassClass:** Startup **Type:** String **Default:** NULL **Description:** Specifies a program that is run (as root) after the authentication process succeeds. By default, no program is run. The conventional name for a file used here is Xstartup. See the Xstartup file section.
systemPath	**ClassClass:** SystemPath **Type:** String **Default:** system_dep._path **Description:** The dtlogin client sets the PATH environment variable for the startup and reset scripts to the value of this resource. Note the conspicuous absence of "." from this entry. This is a good practice to follow for root because it avoids many system penetration schemes.
systemShell	**ClassClass:** SystemShell **Type:** String **Default:** /bin/sh **Description:** The dtlogin client sets the SHELL environment variable for the startup and reset scripts to the value of this resource.

Item	Description
terminateServer	**ClassClass:** TerminateServer **Type:** Boolean **Default:** False **Description:** Specifies whether the X server should be terminated when a session ends (instead of resetting it). This option can be used if the server tends to grow indefinitely over time in order to limit the amount of time the server is run continuously.
termSignal	**ClassClass:** Signal **Type:** Int **Default:** 15 (SIGTERM) **Description:** Specifies the signal dtlogin sends to terminate the server.
userAuthDir	**ClassClass:** UserAuthDir **Type:** String **Default:** /var/dt **Description:** When dtlogin cannot write to the usual user authorization file ($HOME/.Xauthority), it creates a unique file name in this directory and points the environment variable XAUTHORITY at the created file.
userPath	**ClassClass:** UserPath **Type:** String **Default:** system_dep._path **Description:** The dtlogin client sets the PATH environment variable for the session to this value. It should be a colon-separated list of directories.
xdmMode	**ClassClass:** XdmMode **Type:** Boolean **Default:** False **Description:** If True, the $HOME/.xsession file will be executed from Xsession upon user authentication, rather than from dtsession.

Item	Description
xrdb	**ClassClass:** Xrdb **Type:** String **Default:** /system_dep./xrdb **Description:** Specifies the program used to load the resources. The authentication screen reads a *name-password* pair from the keyboard. Because this is a Motif toolkit client, colors, fonts and some layout options can be controlled with resources. General resources for this screen should be put into the file named by the resources resource (Xresources is the default). Specify language-specific values, such as text or fonts, in the Dtlogin app-defaults file.

Logo Resources

The default logo on the authentication screen can be replaced with a bitmap or pixmap of the user's choice. The resources should be prefaced with the string Dtlogin*logo* when specified.

Item	Description
bitmapFile	**ClassClass:** BitmapFile **Type:** String **Default:** NULL **Description:** Specifies the absolute path name to the bitmap or pixmap file to be used for the logo.
background	**ClassClass:** Background **Type:** Pixel **Default:** #a8a8a8 **Description:** Specifies the background color for the logo.
topShadowPixmap	**ClassClass:** topShadowPixmap **Type:** String **Default:** 25_foreground **Description:** Specifies the pixmap to use for the logo border shadow.

The following resources describe the greeting string used on the login screen. The resources should be prefaced with the string Dtlogin*greeting* when specified.

Item	Description
foreground	**ClassClass:** Foreground **Type:** Pixel **Default:** black **Description:** Specifies the foreground color for the welcome message.
background	**ClassClass:** Background **Type:** Pixel **Default:** dynamic **Description:** Specifies the background color for the welcome message. The default is light gray for color systems or white for monochrome systems.
fontlist	**ClassClass:** FontList **Type:** FontList **Default:** -*-*schoolbook-medium-i-normal--18-* **Description:** Specifies the font to use for the welcome message.
labelString	**ClassClass:** LabelString **Type:** String **Default:** Welcome to %LocalHost% **Description:** Specifies the string to use for the welcome message. Multiple lines can be specified by including newline characters (0 in the text. If the token %LocalHost% is included in the text, it will be replaced with the name of the host providing login service. If the token %DisplayName% is included in the text, it will be replaced with the display name.
perLabelString	**ClassClass:** LabelString **Type:** String **Default:** Welcome %s **Description:** Specifies the string to use for the personalized welcome message. This is the message displayed after the user name has been entered. The %s will be replaced with the user name entered.

Item	Description
alignment	**ClassClass:** Alignment **Type:** String **Default:** ALIGNMENT_CENTER **Description:** Specifies the string to use for the alignment of the Welcome message. Valid values are ALIGNMENT_BEGINNING, ALIGNMENT_CENTER and ALIGNMENT_END.

Matte Resources

The following resources describe the matte layout used on the login screen. The resources should be prefaced with the `Dtlogin*matte.` string when specified.

Item	Description
width	**ClassClass:** Width **Type:** Int **Default:** 806 for high-resolution displays 755 for medium-resolution displays 585 for low-resolution displays **Description:** Specifies the width to use for the `login_matte`.
height	**ClassClass:** Height **Type:** Int **Default:** 412 for high-resolution displays 385 for medium-resolution displays 300 for low-resolution displays **Description:** Specifies the height to use for the `login_matte`.

Label Resources

The following resources describe the fonts layout used on the login screen. The resources should be prefaced with the `string Dtlogin*.` when specified.

Item	Description
`labelFont`	**ClassClass:** LabelFont **Type:** String **Default:** -*-swiss 742-medium-r-normal-*-140-*-p-110-* for high-resolution displays -*-swiss 742-bold-r-normal-*-140-*-p-100-* for low-resolution displays **Description:** Specifies the `labelFont` to use for the push buttons and labels.
`textFont`	**ClassClass:** TextFont **Type:** String **Default:** -*-prestige-medium-r-normal-*-128-72-* for high-resolution displays -*-helvetica-bold-r-normal-*-100-* for low-resolution displays **Description:** Specifies the `textFont` to use for the push buttons and labels.

Flags

All flags, except `-config`, specify values that can also be specified in the configuration file as resources. Typically, customization is done using the configuration file rather than command line options. These flags are most useful for debugging and one-shot tests.

Item	Description
`-config` *configuration_file*	Specifies a resource file that specifies the remaining configuration parameters. This replaces the `dtlogin` default `Xconfig` file. See the Xconfig file section for more information.
`-daemon`	Specifies `true` as the value for the `daemonMode` resource. This makes `dtlogin` close all file descriptors, disassociate the controlling terminal, and put itself in the background when it first starts up (just like the host of other daemons).
`-debug` *debug_level*	Specifies the numeric value for the *debug_level* resource. A nonzero value causes `dtlogin` to print debugging statements to the terminal; it also disables the `daemonMode` resource, forcing `dtlogin` to run synchronously.
`-error` *error_log_file*	Specifies the value for the *error_log_file* resource. See the Xerrors file section for more information.
`-nodaemon`	Specifies `false` as the value for the resources.
`-resources` *resource_file*	Specifies the value for the *resource_file* resource. See the the Xresources file section for more information.

Item	Description
`-server` *server_entry*	Specifies the value for the *server_entry* resource. See the the Xservers file section for more information.
`-udpPort` *port_number*	Specifies the value for the `requestPort` resource. This sets the port number that `dtlogin` monitors for XDMCP requests. Because XDMCP uses the well-known registered udp port 177, avoid changing this resource except for debugging.
`-session` *session_program*	Specifies the value for the *session_program* resource. See the Xconfig file section for more information.

Environment Variables

The `dtlogin` command invokes the user's session with the following default environment:

Item	Description
`DISPLAY`	Set to the associated display name.
`EDITOR`	Set to `/usr/dt/bin/dtpad`.
`HOME`	Set to the home directory of the user.
`KBD_LANG`	Set to the value of `LANG` for applicable languages.
`LANG`	Set to the current NLS language (if any).
`LC_ALL`	Set to the current NLS language (if any).
`LC_MESSAGES`	Set to the current NLS language (if any).
`LOGNAME`	Set to the user name.
`MAIL`	Set to `/usr/mail/$USER` (system dependent).
`PATH`	Set to the value of the `userPath` resource.
`USER`	Set to the user name.
`SHELL`	Set to the user's default shell (from `/etc/passwd`).
`TERM`	Set to `dtterm`.
`TZ`	Set to the value of the *timeZone* resource or system default.
`XAUTHORITY`	Set to authority file.

Adding to the Environment List

Four methods are available to modify or add to the preceding list depending on the desired scope of the resulting environment variable:

- The `exportList` resource is available to allow the export of variables provided to the `dtlogin` process by its parent. Variables specified by this method are available to both the display's X server process and the user's session, and they override any default settings. The resource accepts a string of *name*=*value* separated by at least one space or tab.
- The `environment` resource is available in the `dtlogin` configuration file to allow setting of environment variables on a global or per-display basis. Variables specified by this method are available to both the display's X server process and the user's session, and they override any default settings. The

resource accepts a string of *name*=*value* separated by at least one space or tab. The values specified must be constants because no shell is used to parse the string. For example:

```
Dtlogin*environment:MAIL_HOST=blanco MAIL_SERVER=pablo
```

Note: The LANG and TZ environment variables have their own dedicated resources in the configuration file and should not be set by the environment.

- Environment variables that require processing by a shell or are dependent on the value of another environment variable can be specified in the startup script Xsession. These variables are loaded into the environment of all users on the display, but not to the X server process. They override any previous settings of the same variable. The Xsession script accepts ksh syntax for setting environment variables. For example:

```
MAIL=/usr/mail/$USER
```

- Personal environment variables can be set on a per-user basis in the $HOME/.dtprofile script file. The dtlogin command accepts either sh, ksh, or csh syntax for the commands in this file. The commands should only be those that set environment variables, not any that perform terminal I/O, with the exception of tset or stty. If the first line of .dtprofile is #!/bin/sh, #!/bin/ksh or #!/bin/csh, dtlogin uses the appropriate shell to parse .dtprofile. Otherwise, the user's default shell ($SHELL) is used.

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To start the CDE login service as a daemon, enter:

```
/usr/dt/bin/dtlogin -daemon
```

2. To start the CDE login service in debug mode, enter:

```
/usr/dt/bin/dtlogin -debug 1
```

Location

/usr/dt/bin/dtlogin

Standard Errors

The dtlogin command returns the following error messages:

- Login incorrect; please try again.
- Unable to change to home directory.
- Sorry. Maximum number of users already logged in.
- Login error, invalid user ID.
- Login error, invalid group ID.
- Login error, invalid audit ID.
- Login error, invalid audit flag.
- Logins are currently disabled.

• Your current password has expired.

Files

The dtlogin command is designed to operate in a wide variety of environments and provides a suite of configuration files that can be changed to suit a particular system. The default dtlogin configuration files can be found in /usr/dt/config with the exception of Xsession, which is stored in /usr/dt/bin. They are as follows:

Item	Description
/usr/dt/config/Xconfig	Specifies other dtlogin configuration files and dtlogin behavior.
/usr/dt/config/Xaccess	Controls access from displays requesting XDMCP service.
/usr/dt/config/Xservers	Contains the list of displays for dtlogin to explicitly manage.
/usr/dt/config/Xresources	Contains resource definitions specifying the appearance of the login screen.
/usr/dt/config/Xsetup	A script executed as root prior to display of the login screen.
/usr/dt/config/Xstartup	A script executed as root after the user has successfully authenticated.
/usr/dt/bin/Xsession	A script executed as the authenticated user that starts the user's session.
/usr/dt/config/Xfailsafe	A script executed as the authenticated user that starts a fail-safe session.
/usr/dt/config/Xreset	A script executed as root after the user's session has exited.

The Xconfig File

The Xconfig file contains the general resources for dtlogin and is at the top of the dtlogin configuration file tree. Xconfig specifies the location of other dtlogin configuration and log files and specifies dtlogin behavior. The location of other dtlogin configuration and log files are specified by resource definitions. The defaults are as follows:

Dtlogin.errorLogFile
/var/dt/Xerrors

Dtlogin.pidFile
/var/dt/Xpid

Dtlogin.accessFile
Xaccess

Dtlogin.servers
Xservers

Dtlogin*resources
%L/Xresources

Dtlogin*setup
Xsetup

Dtlogin*startup
Xstartup

Dtlogin*reset
Xreset

Dtlogin*failsafeClient
Xfailsafe

Dtlogin*session
/usr/dt/bin/Xsession

If the path specified for `accessFile`, `servers`, `resources`, `setup`, `startup`, `reset`, `failsafeClient`, or `session` is relative, `dtlogin` will first look for the file in directory `/etc/dt/config`, then `/usr/dt/config`.

Note: Some of the resources are specified with * separating the components. These resources can be made unique for each different display, by replacing the * with the display-name. Refer to Display Resources for more information.

The default `Xconfig` file is `/usr/dt/config/Xconfig`. A system administrator can customize `Xconfig` by copying `/usr/dt/config/Xconfig` to `/etc/dt/config/Xconfig` and modifying `/etc/dt/config/Xconfig`. The default `Xconfig` file contains the preceding configuration and log file entries in addition to a few vendor specific resource definitions and examples.

The Xaccess File

The database file specified by the `accessFile` resource provides information which `dtlogin` uses to control access from displays requesting XDMCP service. This file contains three types of entries: entries which control the response to Direct and Broadcast queries, entries which control the response to Indirect queries, and macro definitions.

The format of a Direct entry is either a host name or a pattern. A pattern is distinguished from a host name by the inclusion of one or more meta characters (* matches any sequence of 0 or more characters, and ? matches any single character) which are compared against the host name of the display device. If the entry is a host name, all comparisons are done using network addresses, so any name which converts to the correct network address can be used. For patterns, only canonical host names are used in the comparison, so ensure that you do not attempt to match aliases. Putting an exclamation point (`!`) character before either a host name or a pattern causes hosts that match that entry to be excluded.

An Indirect entry also contains a host name or pattern, but follows it with a list of host names or macros to which indirect queries should be sent. Indirect entries can also specify to have `dtlogin` run `dtchooser` to offer a menu of hosts to which a login screen can be displayed.

A macro definition contains a macro name and a list of host names and other macros that the macro expands to. To distinguish macros from host names, macro names start with a % character. Macros can be nested.

When the access for a particular display host is checked, each entry is scanned in turn and the first matching entry determines the response. Direct and Broadcast entries are ignored when scanning for an Indirect entry and vice-versa. Blank lines are ignored, # is treated as a comment delimiter causing the rest of that line to be ignored, and `\newline` causes the newline to be ignored, allowing indirect host lists to span multiple lines.

The following example shows an Xaccess file:

```
#
# Xaccess - XDMCP access control file
#

#
# Direct/Broadcast query entries
#
!xtra.lcs.mit.edu # disallow direct/broadcast service for xtra
bambi.ogi.edu     # allow access from this particular display
*.lcs.mit.edu     # allow access from any display in LCS

#
# Indirect query entries
#

#define %HOSTS macro
%HOSTS                expo.lcs.mit.edu xenon.lcs.mit.edu \
                      excess.lcs.mit.edu kanga.lcs.mit.edu
```

```
#force extract to contact xenon
extract.lcs.mit.edu xenon.lcs.mit.edu

#disallow indirect access by xtra
!xtra.lcs.mit.edu   dummy

#all others get to choose among %HOSTS
*.lcs.mit.edu       %HOSTS
```

If XDMCP access is granted, a temporary file can be created in the directory specified by `authDir` which contains authorization information for the X-terminal. It is deleted when the session starts.

For X terminals that do not offer a host menu for use with Broadcast or Indirect queries, the `chooser` program can do this for them. In the `Xaccess` file, specify CHOOSER as the first entry in the Indirect host list. The `chooser` program sends a Query request to each of the remaining host names in the list and displays a menu of all the hosts that respond. The list might consist of the word BROADCAST, in which case `chooser` sends a Broadcast instead, again displaying a menu of all hosts that respond. On some operating systems, UDP packets cannot be broadcast, so this feature will not work.

An example of an `Xaccess` file using the `chooser` program is as follows:

```
#offer a menu of these hosts to extract
extract.lcs.mit.edu CHOOSER %HOSTS

#offer a menu of all hosts to xtra
xtra.lcs.mit.edu    CHOOSER BROADCAST
```

The program to use for `chooser` is specified by the `chooser` resource. Resources for this program can be put into the file named by resources. The default `Xaccess` file is `/usr/dt/config/Xaccess`. A system administrator can customize `Xaccess` by copying `/usr/dt/config/Xaccess` to `/etc/dt/config/Xaccess` and then modifying `/etc/dt/config/Xaccess`. The default `Xaccess` file contains no entries.

The Xservers File

The `Xservers` file contains the list of displays to manage. The default `Xservers` file is `/usr/dt/config/Xservers`. A system administrator can customize `Xservers` by copying `/usr/dt/config/Xservers` to `/etc/dt/config/Xservers` and then modifying `/etc/dt/config/Xservers`. The default `Xservers` file contains an entry for one local display.

The Xresources File

The `Xservers` file contains the resource definitions specifying the appearance of the login screen. The default `Xresources` file is `/usr/dt/config/Xresources`. A system administrator can customize `Xresources` by copying `/usr/dt/config/Xresources` to `/etc/dt/config/Xresources` and then modifying `/etc/dt/config/Xresources`.

The Xsetup File

The `Xsetup` file typically a shell script. Only root users can run it, and they should be very careful about security. This script is run before the login screen is displayed. No arguments of any kind are passed to the script. The `dtlogin` command waits until this script exits before displaying the login screen.

The default `Xsetup` file is `/usr/dt/config/Xsetup`. A system administrator can customize `Xsetup` by copying `/usr/dt/config/Xsetup` to `/etc/dt/config/Xsetup` and then modifying `/etc/dt/config/Xsetup`. The default `Xsetup` file contains vendor specific code but typically contains code that sets up the X server prior to the display of the login screen, such as setting up keyboard maps.

The Xstartup File

The `Xstartup` file typically a shell script. Only root users can run it, and they should be very careful about security. This is the place to put commands that display the message of the day or do other system-level functions on behalf of the user. The following environment variables are set for the use of this script:

DISPLAY
Set to the associated display name.

HOME
Set to the home directory of the user.

PATH
Set to the value of the systemPath resource.

USER
Set to the user name.

SHELL
Set to the value of the systemShell resource.

No arguments of any kind are passed to the script. The dtlogin command waits until this script exits before starting the user session. If the exit value of this script is nonzero, dtlogin discontinues the session immediately and starts another authentication cycle.

The default Xstartup file is /usr/dt/config/Xstartup. A system administrator can customize Xstartup by copying /usr/dt/config/Xstartup to /etc/dt/config/Xstartup and then modifying /etc/dt/config/Xstartup. The default Xstartup file contains code to change ownership of /dev/console to the user whose session is running on the console.

The Xsession File

The Xsession script initializes a user's session and invokes the desktop session manager. It is run with the permissions of the authorized user, and has several environment variables preset. See Environment Variables for a list of the preset variables.

The default Xsession file is /usr/dt/bin/Xsession. A system administrator can customize Xsession by copying /usr/dt/bin/Xsession to /etc/dt/config/Xsession and then modifying /etc/dt/config/Xsession. The session resource defined in Xconfig must also be changed to reference the customized Xsession file. See The Xconfig File for information on how to update the Xconfig file. The default Xsession file contains session initialization code. It does contain some vendor specific code, but its general function is as follows:

- Sources the user's $HOME/.dtprofile
- Sources any /etc/dt/config/Xsession.d/* scripts
- Sources any /usr/dt/config/Xsession.d/* scripts
- Launches the desktop welcome client, dthello, in the background
- Sources the application search path setup script, dtsearchpath
- Launches the help setup client, dthelpgen, in the background
- Launches the application manager directory setup client, dtappgather, in the background
- Execs the desktop session manager, dtsession
-

System administrators are discouraged from customizing the Xsession file.

The Xreset File

Symmetrical with Xstartup, the Xreset script is run after the user session has terminated. Because it is run by a root user, the Xreset script should contain commands that undo the effects of commands in Xstartup, such as unmounting directories from file servers. The collection of environment variables that were passed to Xstartup are also given to Xreset.

The default Xreset file is /usr/dt/config/Xreset. A system administrator can customize Xreset by copying /usr/dt/config/Xreset to /etc/dt/config/Xreset and then modifying /etc/dt/config/Xreset. The default Xreset file contains code change ownership of /dev/console back to root.

The Xerrors File

The Xerrors script contains error messages from dtlogin and anything output to stderr by Xsetup, Xstartup or Xreset. The system administrator can use the contents of this file for dtlogin troubleshooting. The errorLogSize resource limits the size of the Xerrors file and can prevent it from growing without bound. If the file does grow larger than the requested size and is truncated by dtlogin,

any user who is accessing the file (for example, using `cat` or `tail`) will need to close the file (after the file is truncated) and reopen it for access in order to see subsequent information that is logged to the file.

A system administrator can change the path name of the `Xerrors` by setting the `errorLogFile` resource in the `Xconfig` file.

The Xpid File

The `Xpid` script contains the process ID of the master `dtlogin` process, which can be used when sending signals to `dtlogin`. A system administrator can change the path name of the `Xpid` by setting the `pidFile` resource in the `Xconfig` file.

dtscript Command

Purpose

Builds simple dialogs used in the X Window System environment.

Syntax

dtscript [**-xrm** *options*] [**-dir** *Path*] [**-file** *FileName*] [**-workspace** *WorkspaceName*]

Note: The **-xrm** *options* must be specified, if used, before any other flag.

Description

Desktop Script supports a subset of Motif widgets you drag and drop from the palette into your dialog. You can move or resize any widget in a dialog. You can also edit widget properties using the specialized editors provided.

You can enter callbacks to give widgets desired behavior. When a dialog is complete, Desktop Script generates dtksh code for it.

Flags

Item	Description
-dir *Path*	Sets Desktop Script's current directory shown in the File Select dialog to *Path*.
-file *FileName*	Loads an existing dialog called: *FileName*. The *FileName* argument can be an absolute path name, a path name relative to the current directory, or a path name relative to the **-dir** value.
-workspace *WorkspaceName*	Loads Desktop Script into the corresponding CDE workspace.
-xrm *options*	Enables you to enter any of the specifications (*options*) that you would otherwise put into a resource file.

Examples

To invoke the Desktop Script from a window, enter:

```
dtscript
```

Files

Item	Description
/usr/dt/bin/dtscript	Contains the **dtscript** command.

dtsession Command

Purpose

Manages a CDE session.

Syntax

`dtsession` [*options*] ...

Description

The `dtsession` command provides session management functionality, compliant with ICCCM 1.1, during a user session, from login to logout. It starts a window manager and allows users to save a session, restore a session, lock a session, start screen savers, and allocate colors for desktop-compatible clients.

Note: The desktop login manager `dtlogin` automatically invokes the `dtsession` client through the `Xsession` script. The `dtsession` client can also be started through the `Xsession` script on an existing X server. The `dtsession` session manager automatically starts a window manager.

The `dtsession` command supports the following tasks:

- Initializing a session
- Starting a window manager
- Restoring a home or current session
- Providing session lock on command or timeout
- Providing session screen saver on command or timeout
- Acting as a color allocation server for other desktop clients
- Saving a home or current session
- Displaying confirmation dialog at logout
- Displaying session selection dialog at logout
- Terminating a session

Sessions

A session is the collection of applications, settings, and resources that are present on the user desktop. Session management is a set of conventions and protocols that allow a special session manager, such as `dtsession`, to save and restore a user session. A user can log in to a system and be presented with the same set of running applications, settings, and resources that were present when the user logged off. When a user logs in to the desktop for the first time, a default initial session is loaded. Afterward, `dtsession` supports the notion of a current and a home session.

The following sessions are defined:

Initial session
: When a user logs in to the desktop for the first time, `dtsession` generates the user's initial session by using system default values. For more information, refer to Session Resource Management and Session Application Management.

Current session
: The user session that is running is always considered the current session, whether restored upon login from a saved home session, a saved current session, or the system default initial session. Based on the user's Style Manager Startup settings, when the user exits the session, the current session is automatically saved. When the user next logs in to the desktop, the previously saved current session is restarted. The desktop is restored to the same state it was in when the user last logged out.

Home session
Another option restores the desktop to the same state every time the user logs in, regardless of its state when the user logged out. The user can save the state of the current session, then sets the Style Manager Startup so that the desktop starts that session every time the user logs in.

Display-specific sessions
To run a specific session for a specific display, users can create a display-specific session. To do so, users can copy the `$HOME/.dt/sessions` directory to `$HOME/.dt/`*display*, where display is the real, unqualified host name (for example, `pablo:0` is valid, but `pablo.gato.com:0` or `local:0` is not). When the user logs in on display `pablo:0`, that display-specific session takes precedence.

ICCCM Session Management Protocol

For an application to be saved upon logout and restarted upon login, it must participate in a simple session management protocol. The `dtsession` command supports the ICCCM 1.1 Session Management Protocol.

Applications that want to save their state can take part in the WM_SAVE_YOURSELF protocol. To do so, an application requires to set the WM_SAVE_YOURSELF property on only one of its top-level windows. When a session is saved, `dtsession` sends the application's top-level window a WM_SAVE_YOURSELF client message. The application proceeds to quietly save its state. The application cannot interact with the user in any way while it is saving its state. Because an application likely saves its state into a file, the session manager provides a convenience function, `DtSessionSavePath`, which returns a full path name of a file in which an application can save its state. While the application is saving its state, `dtsession` awaits notice from the application that it is finished. To tell `dtsession` that the save is complete, the application must update the WM_COMMAND property on its top-level window.

The WM_COMMAND property on an application's top-level window serves two purposes. First, a change of this property indicates to `dtsession` that an application is finished saving its state and `dtsession` can proceed to the next application. Second, the WM_COMMAND property value is expected to contain the command line that `dtsession` uses to restart the application at session startup. If an application is started with a full path name, it must use the full path name when setting the WM_COMMAND value. Applications that do not require to save their state but want to be restarted can set the WM_COMMAND value once during application startup.

Restoring a Session

At session startup time, `dtsession` determines which session to restore. The following list describes the order of precedence:

1. Display-specific current of home session
2. Current or home session
3. Initial session

Session Resource Management

The session manager uses the X Server RESOURCE_MANAGER property on which to make available desktop resources to all applications. The session manager loads the RESOURCE_MANAGER in the following manner:

1. Loads the system default resources.
2. Merges any system administrator-specified resources.
3. Merges any user-specified resources.

The desktop default resources can be found in the `/usr/dt/config/$LANG/sys.resources` file. These resources are made available to each user session through the RESOURCE_MANAGER property. Do not edit this file because it is unconditionally overwritten during subsequent desktop installations.

By creating a `/etc/dt/config/$LANG/sys.resources` file, a system administrator can override system default resources or specify more resources. Because this file is merged into the desktop default

resources during session startup, only new or updated resource specifications must be placed in this file. It is preferable to making a copy of the desktop default resource file. Resources that are specified in this file are made available to each user session through the RESOURCE_MANAGER property. Resources that are specified in this file take precedence over those resources that are specified in the desktop default resource file.

By editing the `$HOME/.Xdefaults` file, a user can override the desktop default and system administrator resources. Resources that are specified in this file are made available to only that user session through the RESOURCE_MANAGER property and take precedence over those resources that are specified in the desktop default or system administrator resource files.

Note: The X Toolkit Intrinsics specify that it loads application resources from either RESOURCE_MANAGER or from `$HOME/.Xdefaults`, but not both. Ordinarily, it means that the user's `$HOME/.Xdefaults` file is ignored. However, the session manager accommodates `$HOME/.Xdefaults` by merging it into the RESOURCE_MANAGER at session startup. When users change their `$HOME/.Xdefaults` files, their changes are not visible to new applications until the users invoke the `ReloadResources` action.

The `ReloadResources` action instructs the session manager to reload the RESOURCE_MANAGER with the system-specified, system administrator-specified, and user-specified resources. It makes available to new applications changes that were made to system administrator-specified or user-specified resource files.

Session Application Management

At session startup, the session manager restarts any applications that were saved as part of the session. The system's default set of applications to be restored as part of the user's initial session can be found in the `/usr/dt/config/$LANG/sys.session` file. Do not edit this file because it is unconditionally overwritten during subsequent desktop installations.

A system administrator can replace the set of applications that are restored as part of the user's initial session by creating a `/etc/dt/config/$LANG/sys.session` file. Unlike the resource files, this file is used as a complete replacement for the desktop default file, so you can make a copy of the system default file and make any necessary modifications.

The Window Manager

The `dtsession` command starts the window manager. By default, `/usr/dt/bin/dtwm` is started. An alternative window manager can be specified by using the `wmStartupCommand` resource. For more information, refer to the Workspace Manager specification.

The Style Manager

The style manager provides the interface by which a user can change various desktop and X server settings for the current session. For more information, refer to the Style Manager specification.

The Color Server

The `dtsession` command serves as the color server for the desktop and provides the following set of resources that can be used to configure it:

foregroundColor
Controls whether a pixel is allocated for the foreground color.

dynamicColor
Specifies whether read-only colors are allocated.

shadowPixmaps
Specifies whether colors are allocated for top shadow or bottom shadow.

colorUse
Limits color allocation.

writeXrdbColors
Specifies whether the *background and *foreground resources are placed in the resource database.

For more information, see the Color Server Resources section.

Session Lock

The dtsession command provides locking of session. The current session can be locked directly by pressing the lock icon on the front panel. If supported by the X server, the current session can be locked after a specified period of inactivity. To unlock the session, users must enter their login password, the login password for the root user, or the login password for any of the users specified by the keys resource. See Screen Lock and Screen Save Resources for more information on the keys resource.

The dtsession command is a PAM-enabled session manager with service name dtsession. It supports traditional local UNIX authentication as well as PAM authentication for unlocking the session. Additional reauthentication functionality, such as that required by DCE, can be added by individual vendors.

System-wide configuration to use PAM for authentication is set by establishing root user permissions and modifying the value of the *auth_type* attribute in the usw stanza of the /etc/security/login.cfg file to PAM_AUTH.

The authentication mechanisms that are used when PAM is enabled depend on the configuration for the login service in /etc/pam.conf. The dtsession command requires an /etc/pam.conf entry for the auth module type. The following configuration is recommended in /etc/pam.conf for the dtsession service:

```
dtsession       auth            required        /usr/lib/security/pam_aix
```

Screen Savers

The dtsession command provides support for the launching of external screen savers as a part of session locking from the front panel or, if supported by the X server, after a specified period of inactivity. Refer to the Screen Saver specification for information as to how screen savers are integrated into the desktop.

X Server Screen Saver Extensions

The dtsession command's ability to provide session lock or screen saver launch after a specified period of inactivity depends upon the availability of an X server screen saver extension. The dtsession command supports the X Consortium Sample X11 Screen Saver Extension 1.0 and the HP X Screen Saver Extension. The ability of the dtsession command to recognize both, either, or none of these extensions is vendor-specific.

Starting the Session Manager

The dtsession command must be started from the Xsession script. Xsession is described in the login manager specification. Although starting Xsession from dtlogin as part of the default login sequence is recommended, some systems allow proxy programs, such as **xinit**, **x11start**, or **startx**, to start Xsession.

Color Server Resources

Item	Description

Item	Description
colorUse	**ClassClass:** ColorUse **Type:** String **Default:** DEFAULT **Description:** Specifies the number of colors to use for the user interface. Color server determines type of monitor based upon number of display planes of the screen as follows: **1, 2, or 3 planes (B_W)** Specifies a black-and-white system. The color palettes use 2 color cells for the user interface. In this configuration, only 2 color palettes are available: BlackWhite and WhiteBlack. These palettes cannot dynamically change. To change a palette, all applications that use the color palette must be restarted. This resource value forces ShadowPixmaps to True, and ForegroundColor to either black or white (depending on the palette chosen). **4 or 5 planes (LOW_COLOR)** Specifies a low-color system. The color palettes have two color sets and use a maximum of 12 color cells for the user interface, including black and white (color cells 0 and 1). The number of color cells can be reduced by using the resources ShadowPixmaps and ForegroundColor. **6 planes (MEDIUM_COLOR)** Specifies a medium-color system. The color palettes have four color sets and use a maximum of 22 color cells for the user interface, including black and white (color cells 0 and 1). The number of color cells can be reduced by using the resources ShadowPixmaps and ForegroundColor. **7+ planes (HIGH_COLOR)** Specifies a high-color system. The color palettes have eight color sets and use a maximum of 42 color cells for the user interface, including black and white (color cells 0 and 1). The number of color cells can be reduced by using the resources ShadowPixmaps and ForegroundColor.
dynamicColor	**ClassClass:** DynamicColor **Type:** Boolean **Default:** True **Description:** This resource can have values of True or False. The dynamicColor resource is used to reduce the number of color cells that are being used. After a palette is selected and it is not likely to be changed, dynamicColor can be set to False. If set to False, colors cannot be dynamically changed by using the desktop style manager. A selected palette takes effect the next session. The next time that the session comes up, the color server uses Read Only color cells that can be shared by all clients, reducing the number of color cells used.

Item	Description
foregroundColor	**ClassClass:** ForegroundColor **Type:** String **Default:** DYNAMIC **Description:** This resource can have values of White, Black, or Dynamic. The foregroundColor resource causes all text (foreground) to use either pixel 0 or 1 (Black or White) or to have a color cell that is dedicated to foreground and changes in response to the background color (Dynamic) for each ColorSet. If set to White or Black, the number of color cells that are used per ColorSet is reduced by 1.
shadowPixmaps	**ClassClass:** ShadowPixmaps **Type:** String **Default:** DEFAULT **Description:** For color systems, this resource can have a value of True or False. If True, topShadowColor and bottomShadowColor use the same pixel as background and topShadowPixmap and bottomShadowPixmap are specified instead of solid color to create the 3-D look. It reduces the number of color cells per ColorSet by 2. This resource defaults to True for systems with four or less color planes (16 or less color cells), and False for systems with more than four color planes.
writeXrdbColors	**ClassClass:** WriteXrdbColors **Type:** Boolean **Default:** True

Screen Lock and Screen Save Resources

Item	Description

Item	Description
keys	**ClassClass:** Keys **Type:** unsigned char **Default:** NULL **Description:** Lists key holders that can unlock the screen any time it is locked by the user. The list is a list of user IDs separated by commas. For example, if user `kim` has the following resource active during a session, users `fred` and `keith` can unlock the display when `kim` locks it: `Dtsession*keys: fred,keith`
passwordTimeout	**ClassClass:** passwordTimeout **Type:** unsigned int **Default:** 10 **Description:** Specifies (in seconds) the amount of time before the password dialog is removed from the screen. When the display is locked, the pointer shows a lock cursor, and a dialog is displayed that asks for the user password. If no activity from the pointer or keyboard is detected for *passwordTimeout* seconds, the dialog is removed from the screen. The dialog is redisplayed as soon as a pointer or keyboard event is detected. A *passwordTimeout* of 0 leaves the password dialog in place for the entire time the display is locked. The default value is 10 seconds.

Miscellaneous Resources

Item	Description
queryServerSettings	**ClassClass:** QueryServerSettings **Type:** Boolean **Default:** False **Description:** Specifies whether the `dtsession` command queries the server at logout for all its settings, or whether it saves only those settings that are set by using the desktop Style Manager. Querying the server ensures that all settings are saved; however, there is a degradation in performance when a full query is done. The default value is False, which means that the server is not queried.

Item	Description
saveFontPath	**ClassClass:** SaveFontPath **Type:** Boolean **Default:** False
wmStartupCommand	**ClassClass:** WmStartupCommand **Type:** executable path **Default:** NULL **Description:** Allows for an alternative window manager to be started at login. If this resource is NULL, dtsession starts /usr/dt/bin/dtwm. An alternative startup might look like: `Dtsession*wmStartupCommand: /usr/bin/X11/mwm` The command must not have any commands to a shell in it, and it must not be surrounded by quotes. If any other window manager other than /usr/dt/bin/dtwm is used, clients are restored but might not be restored to the correct position. By default, this resource contains a NULL value.

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To start the session manager from the command line without restoring the previous session, enter:

```
dtsession -norestore
```

Location

/usr/dt/bin/dtsession

Files

Item	Description
/usr/dt/config/$LANG/sys.session	The desktop default set of applications for the user's initial session.
/etc/dt/config/$LANG/sys.session	System administrator-specified set of applications for the user's initial session.

Item	Description
`/usr/dt/config/$LANG/sys.resources`	The desktop default resources.
`/etc/dt/config/$LANG/sys.resources`	The system administrator-specified resources.
`$HOME/.Xdefaults`	The user-specified resources.
	Note: The `dtsession` command stores session information in `$HOME/.dt/display` or `$HOME/.dt/sessions`. The content of these directories must not be directly edited by the user.
`/usr/dt/app-defaults/$LANG/Dtsession`	Default `dtsession` resources.

dtterm Command

Purpose

Provides runtime support of existing applications.

Syntax

dtterm [Flags...]

Description

The **dtterm** client provides runtime support of existing applications that are written for ANSI X3.64-1979 and ISO 6429:1992(E) conformant character terminals.

Flags

Note: The **dtterm** terminal emulator accepts all of the standard X Toolkit command line flags along with additional flags, all of which are listed below (if the flag begins with a **+** instead of a **-**, the flag is restored to its default value):

Item	Description
-132	Causes the DECCOLM escape sequence to be recognized, and the **dtterm** window will resize appropriately. Normally the DECCOLM escape sequence that switches between 80 and 132 column mode is ignored. Associated resource: c132.
+132	Causes the DECCOLM escape sequence to be ignored. This is the default behavior. Associated resource: c132.
-aw	Indicates that auto-wraparound should be allowed. This allows the cursor to automatically wrap to the beginning of the next line when it is at the right-most position of a line and text is output. This is the default behavior. Associated resource: autoWrap.
+aw	Indicates that auto-wraparound should not be allowed. Associated resource: autoWrap.
-background *background_color*	Specifies the background of the terminal window as well as the default background used for the scroll bar and the X11 pointer cursor. Under CDE, this flag defaults to the primary colorset select pixel or background pixel, see -bs. Without CDE, this flag defaults to *background/*Background with an ultimate fallback color of black. *background_color* describes the background color to use. Associated resource: background.

Item	Description
-bd *border_color*	Specifies the border color for all windows. The shell widget's border may not be visible when reparenting window managers such as **dtwm** and **mwm** are used. The default color is black. *border_color* describes the border color to use. Associated resource: borderColor.
-bg *background_color*	Identical to -background. *background_color* describes the background color to use. Associated resource: background.
-bordercolor *border_color*	Identical to -bd above. *border_color* describes the border color to use. Associated resource: borderColor.
-borderwidth *border_width*	Specifies the border width of the shell widget's window. This value may be overridden by reparenting window managers such as **dtwm** and **mwm**. The default is 0. *border_width* specifies the width of the window border in pixels. Associated resource: borderWidth.
-bs	Specifies that the terminal window should use the Motif select color instead of the background color for the terminal window's background color. This is the default behavior. Associated resource: backgroundIsSelect.
+bs	Specifies that the terminal window should not use the Motif select color instead of the background color for the terminal window's background color. Associated resource: backgroundIsSelect.
-bw *border_width*	Identical to -borderwidth. Associated resource: borderWidth.
-C	Specifies that output directed at **/dev/console** should be directed instead to the terminal window. It is provided as a way to prevent output that would normally be displayed on the ITE from overwriting the X server's display. It is not provided as a general mechanism to direct the output from an arbitrary system's **/dev/console** to an arbitrary X server.

Note: You must have ownership of and read/write access to **/dev/console** for this flag to work.

Item	Description
-display *display_name*	Specifies the X11 display server to be used by **dtterm**. This defaults to the value in the $DISPLAY environment variable. *display_name* specifies the X11 server to connect to.
-e *program_argument*...	Specifies an executable program to be invoked as a subprocess when **dtterm** is started. This flag must be the last flag on the command line. *program_argument* specifies the program and command line arguments to run.
-fb *fontset*	Specifies an XFontSet to be used when displaying bold terminal text. It should be specified as a Motif XmFontList. Only character or mono spaced fonts are supported. The behavior when using proportional fonts is undefined. A default bold font will be generated based on the XLFD name of the userFont. If that font is not available, bold text will be generated by overstriking (with a one pixel offset) the userFont. *fontset* specifies the bold terminal XFontSet to use. Associated resource: userFont.
-fg *foreground_color*	Specifies the foreground color of the terminal window as well as the default foreground color used for the scroll bar and for the X11 pointer cursor. Under CDE, this resource will default to the primary color set foreground pixel. Without CDE, this resource will default to *foreground or *Foreground with an ultimate fallback color of white. *foreground_color* specifies the foreground color to use. Associated resource: foreground.

Item	Description
-fn *fontset*	Specifies an XFontSet to be used when displaying terminal text. It should be specified as a Motif XmFontList. Only character or mono spaced fonts are supported. The behavior when using proportional fonts is undefined. This font will not be used to display non-terminal text (menu bar, popup menus, dialogs, etc.). The default is to use the XmNtextFontList value of the parent bulletin board (see XmBulletinBoard) in the same manner as the XmText widget. *fontset* specifies the terminal XFontSet to use. Associated resource: userFont.
-font *fontset*	Identical to -fn. *fontset* specifies the terminal XFontSet to use. Associated resource: userFont.
-foreground *foreground*	Identical to -fg. *foreground* specifies the foreground color to use. Associated resource: foreground.
-geometry *geometry_string*	Specifies the preferred size and position of the terminal window. The default size is 24 lines of 80 characters each. There is no default position. *geometry_string* specifies the terminal geometry to use. Associated resource: geometry.
-help	Displays a message summarizing the usage of **dtterm**.
-iconic	Specifies that the terminal emulator should initially be placed on the display iconified. Associated resource: iconic.
+iconic	Specifies that the terminal emulator should initially be placed on the display as a normal window. This is the default behavior. Associated resource: iconic.
-j	Specifies that jump scrolling should be used. Under jump scrolling, the screen may be scrolled more than one line at a time. This provides for faster screen updates when multiple lines of text are being sent to the terminal. The maximum number of lines that may be jump scrolled is limited to the number of lines in the terminal window. All lines are displayed. This is the default behavior. Associated resource: jumpScroll.
+j	Specifies that jump scrolling should not be used. For a description of jump scrolling, see -j. Associated resource: jumpScroll.
-kshMode	Specifies that **ksh** mode should be enabled. Under **ksh** mode, a key pressed with the extend modifier bit set will generate an escape character followed by the character generated by the un-extended keystroke. This flag is provided for use with emacs and the emacs command line editor mode of **ksh** or **ied**. It conflicts with **** the normal use of the meta key for generating extended single byte characters, and for generating multi-byte Asian characters. Associated resource: kshMode.
+kshMode	Specifies that the **ksh** mode should not be enabled. This is the default behavior. Associated resource: kshMode.
-l	Enables output logging. When logging is enabled, all output received from the subprocess is logged either to a file or to a command pipeline (as specified via the -If flag). Since the data is being logged directly from the subprocess, it includes all escape characters and carriage return/newline pairs sent by the terminal line discipline. Output may be enabled and disabled via escape sequences. Associated resource: logging.
+l	Disables output logging. For a description of output logging, see -l. This flag is the default. Associated resource: logging.

Item	Description
-lf *file_name*	Specifies the name of the file to which the output log described in the -l flag. If *file_name* begins with a pipe symbol (|), the rest of the string is assumed to be a command to be used as the endpoint of a pipe. The default filename is DttermLog*XXXXX* (where *XXXXX* is the process id of **dtterm**) and is created in the directory from which **dtterm** was started. If the last five characters are *XXXXX*, they are replaced by the process ID. *file_name* specifies the log file name to use. Associated resource: logFile.
-ls	Indicates that the shell that is started should be a login shell (i.e. the first character of argv[0] will be a dash, indicating to the shell that it should read the system's profile and the user's **$HOME/.profile** (for **ksh** and **sh**) or the system's **csh.login** and the user's **$HOME.login** (for **csh**). Associated resource: loginShell.
+ls	Specifies that a normal (non-login) shell should be started. This is the default behavior. Associated resource: loginShell.
-map	Indicates that **dtterm** should map (de-iconify) itself upon subprocess output if it is unmapped (iconified). An initial period of time during which **dtterm** will not map itself upon subprocess output may be specified via the mapOnOutputDelay resource. Associated resource: mapOnOutput.
+map	Specifies that there should be no special mapping behavior. This is the default behavior. Associated resource: mapOnOutput.
-mb	Indicates that **dtterm** should ring a margin bell when the user types near the right margin. The actual distance involved is specified by the -nb flag. Associated resource: marginBell.
+mb	Indicates that margin bell should not be rung when the user types near the right margin. This is the default. Associated resource: marginBell.
-ms *pointer_color*	Specifies the foreground color to use for the terminal window's (X11) pointer cursor. The default is to use the terminal window's foreground color. See foreground. *pointer_color* specifies the pointer foreground color to use. Associated resource: pointerColor.

Item	Description
-name *prog_name*	Specifies the X11 name of the **dtterm** window. *prog_name* the name to use.
-nb *number*	Specifies the number of characters from the right margin at which the margin bell will ring, if enabled. The default is 10. Associated resource: nMarginBell.
-r	Causes the **dtterm** window to be displayed with the foreground and background colors reversed. This is identical to the -rv and -reverse flags.
+r	Causes the **dtterm** window to be displayed with the normal foreground and background colors. This is the default, and is also identical to the +rv flag.
-reverse	Causes the **dtterm** window to be displayed with the foreground and background colors reversed. This is identical to the -r and -rv flag.
-rv	Causes the **dtterm** window to be displayed with the foreground and background colors reversed. This is identical to choosing Options | Global Options, and then changing the ``windowBackground'' options menu to ``Inverse.'' A **dtterm** window started with this flag has the`` Window Background'' options menu set to ``Inverse.'' See ``Global Options''.
+rv	Causes the **dtterm** window to be displayed with the normal foreground and background colors. This is the default.

Item	Description
-rw	Specifies that reverse-wraparound should be enabled. Associated resource: reverseWrap.
+rw	Indicates that reverse-wraparound should not be enabled. This is the default. Associated resource: reverseWrap.
-Sccn	Specifies that the terminal emulator should be run against a pre-opened pty or STREAMS device. This flag is provided for use where the pty or STREAMS device's worker name is of the form tty?? (i.e., exactly two characters following the tty). This flag is intended for use when **dtterm** is invoked programmatically from another application. *cc* specifies the last two characters of the pty or STREAMS device's worker name, where the worker name is of the form tty??. This value is ignored, but must be exactly two characters in length. *n* specifies the number of the file descriptor that corresponds to the pty or STREAMS device's already-opened controller side.
-Sc.n	This flag is identical to **-Sccn** above, but is provided for systems with a larger pty name space. *c* specifies the last component of the pty worker name. This values is ignored and may be empty. *n* specifies the number of the file descriptor that corresponds to the pty's already-opened controller side.
-sb	Indicates that a scrollbar should be displayed. This is the default. Associated resource: scrollBar.
+sb	Indicates that a scrollbar should not be displayed. Associated resource: scrollBar.
-sf	Indicates that Sun Function Key escape codes should be generated for function keys instead of standard VT220 escape sequences. Associated resource: sunFunctionKeys.
+sf	Indicates that the standard escape sequences should be generated for function keys instead of the Sun Function Key escape codes. This is the default behavior. Associated resource: sunFunctionKeys.
-sl *screens[s|l]*	Specifies the number of lines in the terminal buffer beyond the length of the window. The flag value consists of a number followed by an optional suffix. If no suffix is included, or the suffix is **l** (ell), the total length of the terminal buffer will be screens plus the length of the terminal window. If the suffix is **s** (ess), the total length of the terminal buffer will be (screens plus one) times the length of the terminal window. **dtterm** will try to maintain the same buffer-to-window ratio when the window is resized larger. The default is **4s**. *screens* specifies the number of screens or lines to save. Associated resource: saveLines.
-ti *term_id*	Supplies the name used to select the correct response to terminal ID queries. Valid values are vt100, vt101, vt102, and vt220. The default is vt220. *term_id* specifies the terminal ID to use.
-title *title_string*	Specifies the window title. If the -e flag is used, the default will be the last component of the program's path. If the -e flag is not used, the default will be the last component of the name used to run **dtterm** (i.e., argv[0]). *title_string* specifies the title to use. Associated resource: title.

Item	Description
-tm *term_modes*	Specifies a string containing terminal-setting keywords and the characters to which they may be bound. Allowable keywords include intr, quit, erase, kill, eof, eol, swtch, start, stop, brk, susp, dsusp, rprnt, flush, weras, and lnext. Keywords that do not apply to a specific architecture will be correctly parsed and ignored. Control characters may be specified as ^ followed by char (e.g. ^c or ^u), and ^? may be used to indicate delete. This is useful for overridding the default terminal settings without having to do an **stty** every time a terminal process is started. The default is NULL. *term_modes* specifies the terminal mode string. Associated resource: ttyModes.
-tn *term_name*	Specifies a name to set the $TERM environment variable to. The default is **vt220**. *term_name* specifies the terminal name to use. Associated resource: termName.
-usage	Prints a usage message on the screen.
-vb	Indicates that a visual bell is preferred over an audible one. Instead of ringing the terminal bell whenever a Control-G is received, the window will be flashed. Associated resource: visualBell.
+vb	Indicates that an audio bell is preferred over a visual one. This is the default behavior. Associated resource: visualBell.
-w *border_width*	Identical to -borderwidth. *border_width* specifies the width of the window border in pixels.
-xrm *resource_string*	Allows X11 Resource Manager-style resources to be specified on the command line. *resource_string* specifies an X11 resource string.

Resources

Item	Description
allowSendEvents	Specifies that the terminal emulator should allow synthetic events (generated and sent by another application). Enabling this resource opens up a possible security risk. The default is False.
appCursorDefault	If True, the cursor keys are initially in application mode. If False, they are initially in cursor mode. The default is False.
appKeypadDefault	If True, the keypad keys are initially in application mode. If False, they are initially in numeric mode. The default is False.
autoWrap	Specifies whether or not auto-wraparound is initially enabled. The default is True.
background	Specifies the background color of the terminal window as well as the default background color used for the scrollbar. Under CDE, this resource defaults to either the primary color set select pixel or the primary color set background pixe, see backgroundIsSelect. The default is the primary color set background pixel. Without CDE, this resource defaults to black.
backgroundIsSelect	When True, this resource specifies that the terminal window should use the Motif select color instead of the background color for the terminal window's background color. The default is False.
blinkRate	Specifies the number of milliseconds the cursor is in the on and off states while blinking. A value of 250 will blink the cursor two times per second. A value of 0 will turn blinking off. The default is 250.

Item	Description
borderColor	Defines the border color for the window. The window border may not be visible when reparenting window managers such as **dtwm** and **mwm** are used. The default is ``black''.
borderWidth	Specifies the border width of the shell widget's window. This value may be overridden by reparenting window managers such as **dtwm** and **mwm**. The default is 0.
c132	Specifies whether or not the DECCOLM escape sequence that switches to window with between 80 and 132 columns should be honored. The default is False.
charCursorStyle	Specifies the shape of the text cursor. A value of char_cursor_box specifies a cursor with the width and height of the base font's bounding box. A value of char_cursor_bar specifies a cursor with the width of the base font's bounding box, a height of two pixels, and drawn with it's top on the baseline. The default is char_cursor_box.
consoleMode	Specifies that output directed at **/dev/console** should be directed instead to the terminal window. It is provided as a way to prevent output that would normally be displayed on the ITE from overwriting the X server's display. It is not provided as a general mechanism to direct the output from an arbitrary system's **/dev/console** to an arbitrary X server. Note that you must have ownership of and read/write access to **/dev/console** for this flag to work. The default is False.
foreground	Specifies the foreground color of the terminal window as well as the default foreground color used for the scrollbar and the color used for the pointer cursor. Under CDE, this resource will default to the primary colorset foreground. Otherwise, it defaults to ``white''.
geometry	Specifies the preferred size and position of the terminal window. The default size is 24 lines of 80 characters each. There is no default position.
iconGeometry	Specifies the preferred position of the terminal emulator's icon. Window managers may ignore this value. There is no default.
iconic	If true, specifies that the terminal emulator should initially be placed on the display iconified. Window managers (including **dtwm** and **mwm** may ignore this value. The default is False.
iconicName	Specifies the name for the icon. If the -e flag is used, the default will be the last component of the program's path. If the -e flag is not used, the default will be the base name of the name used to run **dtterm** (i.e., argv[0]).
jumpScroll	Specifies that jump scrolling should be used. Under jump scrolling, the screen may be scrolled more than one line at a time. This provides for faster screen updates when multiple lines of text are being sent to the terminal. The maximum number of lines that may be jump scrolled is limited to the number of lines in the display. It is guaranteed that all lines will be displayed. The default is True.
kshMode	Specifies that **ksh** mode should be enabled. Under **ksh** mode, a key pressed with the extend modifier bit set will generate an escape character followed by the character generated by the un- extended keystroke. This flag is provided for use with emacs and emacs command line editor mode of **ksh** or **ied**. It conflicts with the normal use of the meta key for generating extended single byte characters and for generating multi-byte Asian characters. The default is False.

Item	Description
logFile	Specifies the name of the file to which the output log described below is written. If the filename begins with a pipe symbol (|), the rest of the string is assumed to be a command to be used as the endpoint of a pipe. The default filename is DttermLog*XXXXX* (where *XXXXX* is a unique character string) and is created in the directory from which the subprocess was started. If the last five characters are *XXXXX*, they are replaced by a unique character string.
logging	Enables output logging. When logging is enabled, all output received from the subprocess is logged either to a file or to a command pipeline (as specified via the logFile flag). Since the data is being logged directly from the subprocess, it includes all escape characters and carriage return/ newline pairs sent by the terminal line discipline. Output may be enabled and disabled via escape sequences. The default is False.
logInhibit	Specifies that device and file logging should be inhibited. The default is False.
loginShell	Specifies that the shell that is started should be a login shell (i.e. the first character of argv[0] will be a dash, indicating to the shell that it should read the system's profile and the user's **$HOME/.profile** (for **ksh** and **sh**) or the system's **csh.login** and the user's **$HOME/.login** (for **csh**). The default is False.

Item	Description
mapOnOutput	Indicates that the terminal emulator should map (de-iconify) itself upon subprocess output if it is unmapped (iconified). An initial period of time during which it will not map itself upon subprocess output may be specified via the mapOnOutputDelay resource. The default is False.
mapOnOutputDelay	Specifies the number of seconds after start-up that **dtterm** will not honor the mapOnOutput resource. This allows for initial output (e.g., shell prompts) to be sent to the terminal without auto mapping the window. The default is 0 (no delay)
marginBell	Specifies whether or not the bell should be run when the user types near the right margin. The default is False.
menuBar	Specifies that a pulldown menu should be displayed. The default is True.
menuPopup	Specifies that a popup menu should be enabled. The default is True.
nMarginBell	Specifies the number of characters from the right margin at which the margin bell should be rung, when enabled. The default is 10.
pointerBlank	Specifies that the pointer cursor should be put into blanking mode. In this mode, the cursor will turn on when the pointer is moved, and will be blanked either after a selectable number of seconds or after keyboard input has occurred. The delay is set via the pointerBlankDelay resource. The default is False.
pointerBlankDelay	Defines the number of seconds to wait before blanking the pointer cursor after the pointer has been moved. A value of 0 invokes pointer blanking only on keyboard input. The default is 2 seconds.
pointerColor	Specifies the foreground color to use for the terminal window's pointer (X11) cursor. The default is to use the terminal window's foreground color. See foreground.

Item	Description
pointerColorBackground	Specifies the background color to use for the terminal windows pointer (X11) cursor. The default is to use the terminal windows background color See background.
pointerShape	Specifies the X cursor font character to use as the pointer cursor. It should be specified as a string from the include file with the leading XC_ removed. The default is **xterm**.
reverseVideo	Specifies whether or not reverse video should be used. The default is False.
reverseWrap	Specifies whether or not reverse-wraparound should be enabled. The default is False.
saveLines	Specifies the number of lines in the terminal buffer beyond length of the window. The value consists of a number followed by an optional suffix. If no suffix is included, or the suffix is **l** (ell), the total length of the terminal buffer will be screens plus the length of the terminal window. If the suffix is **s** (ess), the total length of the terminal buffer will be (screens plus one) times the length of the terminal window. **dtterm** will try to maintain the same buffer-to-window ratio when the window is resized larger. The default is **4s**.
scrollBar	Specifies whether or not the scrollbar should be visible. The default is True.
sunFunctionKeys	Specifies whether or not Sun Function Key escape codes should be generated for function keys instead of standard VT220 escape sequences. The default is False.
termId	Supplies the name used to select the correct response to terminal ID queries. Valid values are vt100, vt101, vt102, and vt220. The default is vt220.
termName	Defines the name for the $TERM environment variable. The default is vt220.
title	Specifies the window title. If the -e flag is used, the default will be the last component of the program's path. If the -e flag is not used, the default will be the last component of the name used to run **dtterm** (i.e., argv[0]).
ttyModes	Specifies a string containing terminal-setting keywords and the characters to which they may be bound. Allowable keywords include: intr, quit, erase, kill, eof, eol, swtch, start, stop, brk, susp, dsusp, rprnt, flush, weras, and Inext. Keywords that do not apply to a specific architecture will be correctly parsed and ignored. Control characters may be specified as ^ followed by char (e.g. ^c or ^u), and ^? may be used to indicate delete. This is very useful for overriding the default terminal settings without having to do an **stty** every time a terminal process is started. The default is NULL.
userBoldFont	Specifies an XFontSet to be used when displaying bold terminal text. It should be specified as a Motif XmFontList. Only character or mono spaced fonts are supported. The behavior when using proportional fonts is undefined. A default bold font will be generated based on the XLFD name of the userFont. If that font is not available, bold text will be generated by overstriking (with a one pixel offset) the userFont.
userFont	Specifies an XFontSet to be used when displaying terminal text. It should be specified as a Motif XmFontList. Only character or mono spaced fonts are supported. The behavior when using proportional fonts is undefined. This font will not be used to display non-terminal text (menu bar, popup menu, dialog, etc.). The default is to use the XmNtextFontList value of the parent bulletin board (see XmBulletinBoard(3X)) in the same manner as the XmText widget.

Item	Description
visualBell	Specifies that a visual bell is preferred over an audible one. Instead of ringing the terminal bell whenever a CTRL-G is received, the windows will be flashed. The default is False.

Pointer Usage

Note: dtterm allows you to select regions of text. Selection is based on the model specified in the Inter-Client Communication Conventions Manual (ICCCM). **dtterm** supports primary selection only. You can copy or paste selected text using primary transfer. Input is treated as keyboard input, and is inserted at the cursor. The select/insert operations and their default assignments are described below.

Item	Description
select	The left button is used to select the text to be copied. Move the pointer to the beginning of the text to copy, press and hold the left button, move the cursor to the end of the text to copy, and release the button. Any currently selected text can be deselected by clicking the left button once without moving the mouse.
insert	The middle button pastes the text from the primary selection, treating it as keyboard input.

Actions

Item	Description
bell *([Percentage])*	This action rings the keyboard bell at the specified percentage above or below the base volume.
break ( )	This action send a break signal to the child process.
cancel ( )	This action sends a **CAN** (cancel) character to the child process.
do ( )	This action sends the escape sequence associated with the **Do** key to the child process.
edit-key *(string)*	This action sends the escape sequence associated with the corresponding edit key to the child process. The interpretation of these keys is application specific. Valid values for string are find, insert, next, prior, remove, and select.
extend-start ( )	Start the extension of the currently selected text. extend-end ( ) **Note:** Extends the current selection. The amount of text selected depends on the number of mouse clicks.
function-key-execute *(num [,type])*	This action sends the escape sequence associated with the corresponding function key *num* to the child process. Valid values for *num* are 1 through 35. If type is set to function (or not set at all), the escape sequence associated with function key *num* is sent to the child process. If *type* is set to **UDK**, then the string associated with user defined key *num* is sent to the child process.
grab-focus ( )	This action performs one of the following depending on the number of multiple mouse clicks. One click will deselect any selected text and set the selection anchor at the pointer position, two clicks will select a word, three clicks will select a line of text, and four clicks will select all text.
hard-reset ( )	This action will perform a hard reset on the terminal emulator.
help ( )	This action sends the escape sequence associated with the DEC VT220 Help key to the child process. The interpretation of this key is application specific.

Item	Description
keymap *(name)*	This action dynamically defines a new translation table whose resource name is name with the suffix Keymap (case is significant). The name "None" restores the original translation table.
keypad-key-execute *(string)*	This action sends the escape sequence associated with the corresponding keypad key to the child process. The interpretation of these keys are application specific. Valid values for *string* include: f1-f4, space, tab, enter, equal, multiply, add, separator, subtract, decimal, divide, and 0 - 9.
move-cursor *(direction)*	This action sends the escape sequence associated with the corresponding cursor motion to the child process. The interpretation of these keys are application specific. Valid values for *direction* include: up, down, backward, and forward.
redraw-display ( )	This action redraws the contents of the text window.
scroll *(count [,units])*	This action will scroll the display memory down if count is less than zero, or up if *count* is greater than zero. The number of lines scrolled is based on *count* and *units*. Valid values for *units* are page, halfpage, or line. The default for *units* is line.
select-adjust ( )	This action extends the selection. The amount of text selected depends on the number of mouse clicks: 1 click = char 2 clicks = word 3 clicks = line 4 clicks = buffer
select-all ( )	This action selects all text.
select-page ( )	This action selects all text on the screen.
self-insert ( )	This action sends the character associated with the key pressed to the child process.
soft-reset ( )	This action perform a soft reset of the terminal.
stop *(state)*	This action either toggles, starts, or stops the process of reading data from the child process. Valid values for *state* are toggle, on, and off.
string *(string)*	This action inserts the specified text *string* as if it had been typed. The *string* must be quoted if it contains whitespace or non-alphanumeric characters. The *string* is interpreted as a hex character constant if it begins with the characters 0x.
tab ( )	This action sends a tab to the child process.
visual-bell ( )	This action flashes the window quickly.
Virtual Bindings	The bindings for virtual keys are vendor specific. Virtual bindings do not apply when the **dtterm** widget has input focus. For information about bindings for virtual buttons and keys, see VirtualBindings.

Files

Item	Description
/usr/bin/diff	Contains the **diff** command.

du Command

Purpose

Summarizes disk usage.

Syntax

du [**-a** | **-s**] [**-k**] [**-m**] [**-g**][**-l**] [**-r**] [**-x**] [**-H** | **-L**][*File ...*]

Description

The **du** command displays the number of blocks used for files. If the *File* parameter specified is actually a directory, all files within the directory are reported on. If no *File* parameter is provided, the **du** command uses the files in the current directory.

If the *File* parameter is a directory, then the number of blocks reported is the sum of blocks allocated for the files in the directory and the blocks allocated for the directory itself.

If the object of the **du** command is a file or directory that exists inside a JFS2 snapshot, the **du** command gives information for the point-in-time object when the snapshot is created. This information does not include how much space is recovered if the snapshot itself is deleted.

Specifying the **-a** flag reports the number of blocks in individual files. Whether the **-a** flag is used or not, individual files specified by the *File* parameter are always listed.

Specifying the **-s** flag reports the total blocks for all specified files or all files in a directory.

The block count includes indirect blocks of each file. Block count is calculated in 512-byte units independent of the cluster size used by the system. Specifying the **-k** flag calculates the block count in 1024-byte units.

Notes:

1. Files with multiple links are counted and written for only one entry.
2. Block counts are based only on file size; therefore, deallocated blocks are not accounted for in the reported block counts.
3. If **du** cannot obtain the file attributes or cannot read directories, it reports an error and the exit status of the command is affected.

Flags

Item	Description
-a	For each file specified, displays the disk usage of the file. For each directory specified, displays the disk usage of each individual file within the directory, including all subdirectories. Contrast this flag with the **-s** flag.
-g	Calculates the block count in GB units rather than the default 512-byte units. The output values for the disk usage would be in floating point numbers as value of each unit in bytes is significantly high.
-H	If a symbolic link is specified on the command line, the **du** command shall count the size of the file or file hierarchy referenced by the link.
-k	Calculates the block count in 1024-byte units rather than the default 512-byte units.
-l	Allocates blocks evenly among the links for files with multiple links. By default, a file with two or more links is counted only once.

Item	Description
-L	If a symbolic link is specified on the command line or encountered during the traversal of a file hierarchy, the **du** command shall count the size of the file or file hierarchy referenced by the link.
-m	Calculates the block count in MB units rather than the default 512-byte units. The output values for the disk usage would be in floating point numbers as value of each unit in bytes is significantly high.
-r	Reports names of inaccessible files and directories. This is the default.
-s	For each file specified, displays the disk usage of the file. For each directory specified, displays the total disk usage of all files within the directory, including all subdirectories. Contrast this flag with the **-a** flag.
-x	When evaluating file sizes, evaluates only those files that reside on the same device as the file or directory specified by the *File* parameter. For example, you may specify a directory that contains files on several devices. In this case, the **-x** flag displays block sizes for all files that reside on the same device as the directory.

Notes:

1. If all or any two of the **-k**, **-m** and **-g** flags are specified, the last one specified takes effect. The output of the disk usage with the flags **-m** and **-g** would be rounded off to the nearest second decimal digit.
2. If both the mutually exclusive options **-H** and **-L** are specified simultaneously, the command does not report an error. The last specified option determines the behavior of the utility.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To summarize the disk usage of a directory tree and each of its subtrees, enter:

   ```
   du /home/fran
   ```

   This displays the number of disk blocks in the `/home/fran` directory and each of its subdirectories.
2. To summarize the disk usage of a directory tree and each of its subtrees in 1024-byte blocks, enter:

   ```
   du -k /home/fran
   ```

   This displays the number of 1024-byte disk blocks in the **/home/fran** directory and each of its subdirectories.
3. To summarize the disk usage of a directory tree and each of its subtrees in MB blocks, enter:

   ```
   du -m /home/fran
   ```

   This displays the number of MB disk blocks rounded off to nearest 2nd decimal digit in the **/home/fran** directory and each of its subdirectories.
4. To summarize the disk usage of a directory tree and each of its subtrees in GB blocks, enter:

   ```
   du -g /home/fran
   ```

This displays the number of GB disk blocks rounded off to nearest 2nd decimal digit in the **/home/fran** directory and each of its subdirectories.

5. To display the disk usage of each file, enter:

```
du  -a /home/fran
```

This displays the number of disk blocks contained in each file and subdirectory of the `/home/fran` directory. The number beside a directory is the disk usage of that directory tree. The number beside a regular file is the disk usage of that file alone.

6. To display only the total disk usage of a directory tree, enter:

```
du  -s /home/fran
```

The **-s** flag instructs the **du** command to display only the sum total disk usage of the `/home/fran` directory and the files it contains. By default, the **du** command displays an error message if it cannot read a file or directory.

7. To display the disk usage of the files and file hierarchies referenced by all the symbolic links in addition to the normal files found during traversal of a the `/home/fran` directory, type:

```
du -L /home/fran
```

8. To report the disk usage of the file or file hierarchy referenced by the symbolic link `mylink`, type:

```
du -H mylink
```

Files

Item	Description
/usr/bin/du	Contains the **du** command.

dump Command

Purpose

Dumps selected parts of an object file.

Syntax

dump { **-a** **-c** **-d** **-g** **-h** **-l** **-n** **-o** **-p-r** **-s** **-t** **-u** **-v** **-H** **-R** **-T**} [**-z***Name* [,*Number*] [**+z***Number*]] [**-t***Index* [**+t***Index*]] [**-X** {**32**|**64**|**32_64**|**d64**|**any**}] *File* ...

Note: Do not put a space between the **-z** *Name* flag and the ,*Number* parameter.

Description

The **dump** command dumps selected parts of the specified *File* parameter. The **dump** command accepts object files, archive object files, and executable files.

Flags

Item	Description
-a	Dumps the archive header of each member of each specified archive.
-c	Dumps the string table.
-d	Dumps the raw data for each section.
-g	Dumps the global symbols in the archive symbol table.

Item	Description
-h	Dumps section headers.
-l	Dumps line number information.
-n	Dumps all loader section information.
-o	Dumps each optional header.
-p	Suppresses header printing.
-r	Dumps relocation information.
-s	Dumps the raw data for each selection.
-t	Dumps symbol table entries.
-t*Index*	Dumps only the index symbol table entry specified with the *Index* parameter. Use the **-t** flag with the **+t** flag to specify a range of symbol table entries.
+t*Index*	Dumps the symbol entry in the range that ends with the *Index* parameter. The range starts at the first symbol table entry or at the entry specified by the **-t** flag.
-u	Underlines the name of the *File* parameter.
-v	Dumps the information in symbolic representation rather than numeric. Any flag except the **-o** flag and **-s** flag can be used with the **-v** flag.
-z*Name*[,*Number*]	Dumps line number entries for the *Name* parameter or a range of line number entries that starts at the specified number.
+z*Number*	Dumps all line numbers up to the *Number* parameter.
-H	Dumps the header of the loader section. The **-H** flag applies only to executable files.
-R	Dumps the relocation entries for the leader section. The **-R** flag applies only to executable files.
-T	Dumps the symbol table entries for the loader section. The **-T** flag applies only to executable files.
-X *mode*	Specifies the type of object file **dump** should examine. The *mode* must be one of the following: **32** Processes only 32-bit object files. **64** Processes only 64-bit object files. **32_64** Processes both 32-bit and 64-bit object files. **d64** Examines discontinued 64-bit XCOFF files (magic number = U803XTOCMAGIC). **any** Processes all of the supported object files. The default is to process 32-bit object files (ignore 64-bit objects). The *mode* can also be set with the **OBJECT_MODE** environment variable. For example, **OBJECT_MODE=64** causes **dump** to process any 64-bit objects and ignore 32-bit objects. The **-X** flag overrides the **OBJECT_MODE** variable.

Examples

1. To dump the string table of the `a.out` file, enter:

```
dump -c a.out
```

2. To dump the contents of an XCOFF data section to standard output, enter:

```
dump -d a.out
```

3. To dump the object file headers, enter:

```
dump -o a.out
```

4. To dump line number information for the `a.out` file, enter:

```
dump -l a.out
```

5. To dump relocation information for the `a.out` file, enter:

```
dump -r a.out
```

6. To dump the contents of the `a.out` object file text section, enter:

```
dump -s a.out
```

7. To dump symbol table information for the `a.out` object file, enter:

```
dump -t a.out
```

8. To print symbol table entries 20 to 31 without header information, enter:

```
dump -p -t20 +t30 a.out
```

9. To dump the object file headers from only 64-bit objects in lib.a, enter:

```
dump -X64 -o lib.a
```

dumpcheck Command

Purpose

Checks to see that the dump device and copy directory are able to receive the system dump. An error is logged by default if there will likely be insufficient resources to accommodate the dump.

Syntax

/usr/lib/ras/dumpcheck [[**-l**] [**-p**] [**-t** *TimeParameters*] [**-P**]] | [**-r**]

Description

The **/usr/lib/ras/dumpcheck** command is used to check the disk resources used by the system dump. The command logs an error if either the largest dump device is too small to receive the dump or there is insufficient space in the copy directory when the dump is to paging space.

dumpcheck is normally run by cron at 3:00 pm local time each day. This can be varied using the **-r** flag to remove it from root's **crontab** or **-t** *TimeParameters* to change the time at which dumpcheck is executed. It may also be configured from SMIT. **dumpcheck** is automatically added to root's **crontab** when the service aids are installed.

For maximum effectiveness, **dumpcheck** should be run when the system is most heavily loaded. At such times, the system dump is most likely to be at its maximum size. Also, even with **dumpcheck** watching

the dump size, it may still happen that the dump would not fit on the dump device or in the copy directory at the time it happens. This could occur if there is a peak in system load right at dump time.

The **dumpcheck** function is installed as part of the service aids file set, installed automatically.

Flags

Item	Description
-l	Logs any warnings to the error log. This is the default if no parameters are specified.
-p	Prints any warnings produced to stdout.
-P	Indicates that the changes are to be made permanently; that is, they apply to subsequent executions of the **dumpcheck** facility. The **-P** flag is unnecessary with the **-t** and **-r** flags. If the **-P** flag is specified, **dumpcheck** simply changes the **crontab** entry without performing any checks.
-r	Removes the **crontab** entry for this function, effectively unconfiguring it. This command is normally run by **cron**. The **-r** flag must be specified alone. It is not valid with any other flags.
`-t` *TimeParameters*	Changes the time when **dumpcheck** is executed. The *TimeParameters* flag must be enclosed within single or double quotes. It specifies the **crontab** time parameters, the first five parameters of a line in the **crontab** file. See the **crontab** command for the format of the time parameters. The **-t** flag is invalid with the **-r** flag. If the **-t** flag is specified, **dumpcheck** just changes the **crontab** entry without performing any checks.

Security

This command can only be executed by the root user.

Examples

1. To check dump resources and have the results printed to standard output rather than logged, type:

   ```
   /usr/lib/ras/dumpcheck -p
   ```

   To make this change permanently; that is, to have it made in the **crontab** entry, type:

   ```
   /usr/lib/ras/dumpcheck -p -P
   ```

2. To have **dumpcheck** run at 9:00 am and 3:00 pm Monday through Friday, type:

   ```
   /usr/lib/ras/dumpcheck -t "0 9,15 * * 1-5"
   ```

   To return to the default, type:

   ```
   /usr/lib/ras/dumpcheck -t "0 15 * * *"
   ```

   You may also use SMIT to configure the times when **dumpcheck** executes.

3. To discontinue running this feature, type:

   ```
   /usr/lib/ras/dumpcheck -r
   ```

   You may also use SMIT for this task.

dumpctrl Command

Purpose

Manages system dumps and live dumps.

Syntax

dumpctrl **-k**

dumpctrl **-R** [**l** | **s**] [**-P**]

dumpctrl **-s** [**-c** | **-C** *comp-path-list*] [**-l** | **-L** *comp-alias-list*] [**-t** | **-T** *type_subtype*] [**-r**] [**-u**]

dumpctrl **-q***c* [**-c** *comp-path-list*] [**-l** *comp-alias-list*] [**-t** *type_subtype*] [**-r**] [**-u**] [**-p** | **-P**]

dumpctrl **-q***l* [**-p** | **-P**]

dumpctrl **-q***s* [**-p** | **-P**]

dumpctrl [**-P**] [*global_attribute*]

dumpctrl [**-c** *comp-path-list*] [**l** *comp-alias-list*] [**-t** *type_subtype*] [**-r**] [**-u**] [**-n** | **-p** | **-P** | **-x**] [*per-component_attribute*]

Description

There are two types of dump components:

component
Refers to a component specified with the RAS infrastructure (one created with the `ras_register()` kernel service).

legacy component
Refers to a dump component specified with either the `dmp_add()` or the `dmp_ctl()` kernel service.

The **dumpctrl** command is used to obtain information about which components are registered for live dumps or system dumps, and to query and change dump characteristics.

Components are specified with the full path name, device logical alias, type or subtype. You can use multiple flags to specify multiple components or component lists.

Flags

At least one flag must be specified.

Item	Description
-c *comp-path-list*	Specifies components by path name. Wildcards are allowed. Use the **-c all** command to specify all of the components.
-k	Refreshes the list of dumps of the kernel. This flag is run every 5 minutes by default. This period can be changed by editing the **crontab** command for the root user and changing the entry for **/usr/sbin/dumpctrl -k**. For more information, see the **crontab** command. You must run the **dumpctrl -k** command after you add or remove dumps by hand. If the system is holding any dumps in the heap that it previously could not write to the file system, the system attempts to write those dumps and reclaim their storage space now.
-l *comp-alias-list*	Specifies components by alias. Wildcards are allowed.
-r	Dumps any subcomponents of the specified components.

Item	Description
-q *cmd*	Queries attributes for the live dump or system dump. • The **-qc** flag shows component-specific live dump attributes and system dump attributes. The **-qc** flag can be used with the **-p** or **-P** flag to query persistent per-component attributes. The **-qc** flag shows the attributes for all components if neither the **-c**, **-l**, nor **-t** flag is specified. In other words, **-c all** is the default. The following is a sample output for this command: • The **-ql** flag lists global live dump settings. The **-ql** flag can be used with the **-p** or **-P** flag to query persistent global live dump settings. • The **-qs** *type* flag shows global system dump attributes. The **-qs** can be used with the **-p** or **-P** flag to query global system dump attributes.
-r	Includes components below the specified components in the component hierarchy.
-**Rx**	Restores dump settings to their defaults. *x* can be `l` for live dump settings, or `s` for system dump settings. It resets only the global dump settings. Individual components cannot be specified. The **-P** flag and a new boot image are required to ensure all of these settings remain in effect across a restart.
-t *type_subtype*	Specifies a component by *type_subtype* names.
-s	Lists the path names and titles of all live dumps in the dump repository. If components are specified with the **-c**, **-l**, or **-t** flag, the list of dumps that are shown contains dumps only with the specified components. If components are specified with the **-C**, **-L**, or **-T** flag, the list of dumps that are shown contains dumps only with the specified failing components.
-C *comp-path-list*	Specifies components by path name. Wildcards are allowed. The reserved name `all` is also allowed to indicate all components. The **-C** flag is only valid with the **-s** flag.
-L *comp-alias-list*	Specifies components by alias. Wildcards are allowed. The **-L** flag is only valid with the **-s** flag.
-T *type_subtype*	Specifies a component by *type_subtype* names. The **-T** flag is valid only with the **-s** flag.
-u	Includes components above the specified components in the component hierarchy.

```
dumpctrl -qc -r -l vmm -l proc

-------------------------------------------------------------------------------
     Component name                       | Have | Live Dump | System Dump
                                          |Alias |  /level   |  /level
-------------------------------------------------------------------------------
vmm                                       | no   |  on/3     |  on/3
     .pft                                 | no   |  on/3     |  on/3
     ...
proc                                      | no   |  on/4     |  on/3
     ...
```

Persistence flags

Item	Description
-p	Changes apply only to newly created components, which are RAS infrastructure components that are created after the **dumpctrl** command runs.

Item	Description
-P	Makes the specified changes permanent. Any changes that are made remain in effect across a restart. If a new boot image is required, a message is produced to notify you about it. The **-P** flag applies to component attributes, the global enabling or disabling of live dump, the global live dump level, the enabling or disabling of legacy components, and the system dump device specifications.
-n	Changes apply to existing components. The **-n** flag is the default if neither **-p** nor **-P** is specified. To apply changes to both current and newly created components, use the **-n** and **-p** flags.
-x	Deletes this persistence specification. The **-x** flag deletes a permanent (**-P**) persistence specification. The specification must be specified in the same manner as it was originally specified with the **-P** flag.

Recursive-down customization (specified by the **-r** flag) take precedence over all other customization, regardless of the order in which they are specified relative to other non-recursive-down customization.

If you do not know the customization that is made but want to restore the default system setting, you can do one of the following actions:

- In the `/var/adm/ras/raspertune` file, delete the lines relevant to the customization and run the **bosboot** command to restart AIX.
- Read the `/var/adm/ras/raspertune` file to figure out the appropriate flags and parameters that are specified. Then, use the **-x** flag to delete the customization. Run the **bosboot** command and restart AIX.

For more information about how the various dump attributes interact with persistence, see the live dump and system dump attribute tables in "Attributes" on page 1126.

Attributes

The dump attributes can take the form `attribute=value`. For example,

```
dumpctrl dir=/usr/dumps freespc=20
```

This example sets dump directory to `/usr/dumps`, and the free space threshold to 20%.

Some shortcuts are provided, such as the **ldmpon** attribute, which is the same as `ldmpenable=yes`.

If components are given, unrecognized attributes are passed to callbacks of those components by using **RASCD_DMP_PASS_THROUGH**.

The following table lists live dump attributes.

Table 1. Live dump attributes and defaults

Attribute	Specification	Default value
ldmpenable	Specifies whether live dump is enabled. The possible values are yes and no. You can use the **ldmpon** attribute instead of `ldmpenable=yes`, and the **ldmpoff** attribute instead of `ldmpenable=no`.	yes For more information, see the following note "1" on page 1128.
dir	Specifies a live dump directory name.	`/var/adm/ras/livedump`
freespc	Specifies live dump free space threshold by using a decimal value from 0 to 99.	25 (means 25%)

Table 1. Live dump attributes and defaults (continued)		
Attribute	**Specification**	**Default value**
ldmplevel	Specifies the live dump level by using a decimal value from 0 to 9. You can specify the **ldmpminimal**, **ldmpnormal**, or **ldmpdetail** attribute instead of `ldmplevel=1, 3, 7`	3 (normal) For more information, see the following note "1" on page 1128.
heapsz	Specifies live dump heap size by using a decimal value in megabytes.	0 For more information, see the following note "2" on page 1128.
duptype	Specifies duplicate dump suppression type. The following are the possible values: • all • pre • post • none	all
maxfreeze	Specifies the maximum recommended system freeze interval by using a decimal number in milliseconds.	100 ms

The following table lists system dump attributes.

Table 2. System dump attributes and defaults		
Attribute	**Specification**	**Default value**
sdmpenable	Specifies whether system dump is enabled. The possible values are yes and no. You can also specify the **sdmpon** or **sdmpoff** instead of `sdmpenable=yes` or `sdmpenable=no`.	yes For more information, see the following note "3" on page 1128.
legacyenable	Specifies whether dump legacy components are enabled. The possible values are yes and no. You can also specify the **legacyon** or **legacyoff** instead of `legacyenable=yes` or `legacyenable=no`.	yes
sdmplevel	Specifies the system dump level by using a decimal value from 0 to 9. You can specify the **sdmpminimal**, **sdmpnormal**, or **sdmpdetail** attribute instead of `sdmplevel=1, 3, 7`	3 (normal) For more information, see the following note "4" on page 1128.
copydir	Specifies a copy directory path name.	`/var/adm/ras`

Table 2. System dump attributes and defaults (continued)		
Attribute	**Specification**	**Default value**
forcecopy	Specifies whether the **forcecopy** attribute is enabled. The possible values are yes and no. If a dump must be copied from paging space at boot time, and there is not enough space in the copy directory, you are prompted to copy the dump to removable media if the **forcecopy** value is yes. If the value is no, the dump is not copied and the system boots normally, although the dump might be lost.	yes
keyseq	Specifies whether the key sequences always cause a dump. The possible values are yes and no.	no
primary	Specifies the primary dump device path name.	/dev/hd6 or /dev/lg_dumplv
secondary	Specifies the secondary dump device path name.	/dev/sysdumpnull

Notes:

1. The **ldmpenable** and **ldmplevel** attributes can be specified with or without components. If specified without components, the attributes apply to the corresponding global attributes.
2. The **heapsz** attribute (heap size) can be set to 0, meaning that, at dump initialization time, the system calculates the live dump heap size that is based on the amount of real memory, which is the minimum of 64 MB and 1/64 the size of real memory.
3. Individual components must be specified when the **sdmpenable** attribute is given. If no components are given, the **sdmpenable** attribute cannot be specified because the system dump cannot be disabled.
4. The **sdmplevel** attribute can be specified with or without components. If specified without components, it applies to the system default level. The components with **sdmplevel** that are greater than the global **sdmplevel** value are not included in a system dump.

The following table lists live dump attributes and their persistence.

Table 3. Live dump attributes and persistence		
Attribute	**Description**	**Persistence**
ldmpenable	Live dump enabled	Controlled by persistence flags, new boot image is required with the **-P** flag.
dir	Live dump directory	Takes effect immediately and upon system restart.
freespc	Live dump free space threshold	Takes effect immediately and upon system restart.
ldmplevel	Live dump level	Controlled by persistence flags, new boot image is required with the **-P** flag.
heapsz	Live dump heap size	Takes effect immediately and upon system restart.

Table 3. Live dump attributes and persistence (continued)		
Attribute	**Description**	**Persistence**
duptype	Duplicate dump suppression type	Takes effect immediately and upon system restart.
maxfreeze	Maximum recommended system freeze interval	Takes effect immediately and upon system restart.

Note: Persistence affects the attributes only when they apply to RAS infrastructure components. Persistence also controls the global live dump level and global enabled or disabled status.

The following table lists system dump attributes and their persistence.

Table 4. System dump attributes and persistence		
Attribute	**Description**	**Persistence**
sdmpenable	System dump enabled	Controlled by persistence flags, new boot image is required with the **-P** flag.
legacyenable	Dump legacy components	Takes effect immediately, and upon system restart with the **-P** flag. No new boot image is required with the **-P** flag.
sdmplevel	System dump level	Controlled by persistence flags, new boot image is required with the **-P** flag.
copydir	Copy directory	Takes effect immediately and upon system restart.
forcecopy	Brings up the boot time menu if it cannot copy	Takes effect immediately and upon system restart.
keyseq	Key sequences always cause a dump	Takes effect immediately and upon system restart.
primary	Primary dump device	Takes effect immediately, and upon system restart with the **-P** flag. No new boot image is required with the **-P** flag.
secondary	Secondary dump device	Takes effect immediately, and upon system restart with the **-P** flag. No new boot image is required with the **-P** flag.

Note: Persistence affects the attributes when they apply to components.

The **copydir, forcecopy, keyseq, primary**, and **secondary** attributes behave like their **sysdumpdev** command counterparts that are specified with the **-d**, **-D**, **-k**, **-K**, **-p**, and **-s** flags.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.

Item	Description
non-zero	An error occurred. This command fails under the following conditions: • One or more parameters are invalid. • One or more attributes are invalid. • A component cannot be specified. • At least one component must be specified. • The persistent specification cannot be found. (It can occur with the **-x** flag.)

Security

Only the root user can use this command.

dumpfs Command

Purpose

Dumps file system information.

Syntax

dumpfs { *FileSystem* | *Device* }

Description

The **dumpfs** command prints out the superblock, i-node map, and disk map information for the file system or special device specified. This listing is used to find out file system information. Primarily, the **dumpfs** command is for debugging purposes.

The **dumpfs** command can also run against a JFS2 snapshot. The **dumpfs** command prints out the superblock, snapshot map, and block map xtree copy for the specified snapshot.

Note: The `dumpfs` command will not work on UDF, NFS, or JFS diskettes.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Examples

To print the information for **/dev/hd4**, enter:

```
dumpfs /dev/hd4
```

e

The following AIX commands begin with the letter e.

echo Command

Purpose

Writes character strings to standard output.

Syntax

echo [*String ...*]

Description

The **echo** command writes character strings to standard output. *String*s are separated by spaces, and a new-line character follows the last *String* parameter specified. If no *String* parameter is specified, a blank line (new-line character) is displayed.

Normally you could distinguish between a flag and a string that begins with a hyphen by using a -- (double hyphen). Since no flags are supported with the **echo** command, a -- (double hyphen) is treated literally.

The **echo** command recognizes the following escape conventions:

Item	Description
\a	Displays an alert character.
\b	Displays a backspace character.
\c	Suppresses the new-line character that otherwise follows the final argument in the output. All characters following the **\c** sequence are ignored.
\f	Displays a form-feed character.
\n	Displays a new-line character.
\r	Displays a carriage return character.
\t	Displays a tab character.
\v	Displays a vertical tab character.
****	Displays a backslash character.
\0*Number*	Displays an 8-bit character whose ASCII value is a 0-, 1-, 2-, or 3-digit octal number.

Note: The **bsh**, **ksh**, and **csh** commands each contain a built-in **echo** subcommand. The **echo** command and the **bsh** and **ksh echo** subcommands work the same way. The **csh echo** subcommand does not work the same way as the **echo** command.

The \ (backslash) is a quote character in the shell. This means that unless the \ is used with an escape character or enclosed in quotes, for example "\" or '\', the shell removes the backslashes when the command is expanded.

After shell expansion, the **echo** command writes the output based on the escape sequences in the input. Refer to the Backslash Reduction table for an example comparison of how backslashes in a command are first reduced by the shell and then by the **echo** command:

Backslash Reduction

Command Entered	After Shell Expansion	After echo Command Processing
echo hi\\\\there	echo hi\\there	hi\there
echo 'hi\\\\there'	echo 'hi\\\\there'	hi\\there
echo "hi\\\\there'	echo "hi\\there"	hi\there

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To write a message to standard output, enter:

   ```
   echo Please insert diskette . . .
   ```

2. To display a message containing special characters, enter:

   ```
   echo "\n\n\nI'm at lunch.\nI'll be back at 1:00."
   ```

   This skips three lines and displays the message:

   ```
   I'm at lunch.
   I'll be back at 1:00.
   ```

   Note: You must put the message in quotation marks if it contains escape sequences. Otherwise, the shell interprets the \ (backslash) as a metacharacter and treats the \ differently.

3. To use the **echo** command with pattern-matching characters, enter:

   ```
   echo The back-up files are: *.bak
   ```

   This usage displays the message `The back-up files are:` followed by the file names in the current directory ending with `.bak`.

4. To add a single line of text to a file, enter:

   ```
   echo Remember to set the shell search path to $PATH. >>notes
   ```

   This usage adds the message to the end of the file notes after the shell substitutes the value of the **PATH** shell variable.

5. To write a message to the standard error output, enter:

   ```
   echo Error: file already exists. >&2
   ```

   This command redirects the error message to standard error. If the >&2 is omitted, the message is written to standard output.

File

Item	Description
/usr/bin/echo	Contains the **echo** command.

ed or red Command

Purpose

Line editor for text files.

Syntax

ed [**-p** *String*] [**-s** | **-**] [*File*]

red [**-p***String*] [**-s** | **-**] [*File*]

Description

The **ed** command starts the ed editor line-editing program. The ed editor works on only one file at a time by copying it into a temporary edit buffer and making changes to that copy. The ed editor is part of a family of editors that also includes the edit editor, ex editor, and vi editor. The ed editor makes the changes you specify in a buffer. It does not alter the file itself until you use the write (**w**) subcommand.

You can specify the name of the file you want to edit when you start the ed editor with the **ed** command, or you can use the **e** subcommand. When the **ed** command reads a new file into the buffer, the contents of that file replace the buffer's previous contents.

The **red** command is a restricted version of the **ed** command, for use with the restricted shell (**rsh**). With the **red** command, you edit only files that reside in the current directory or in the **/tmp** directory; you cannot use the **!** subcommand.

An ed editor subcommand consists of zero, one, or two addresses, followed by a single-character subcommand, followed by optional parameters to that subcommand. The addresses specify one or more lines in the buffer. Because every subcommand has default addresses, it is frequently unnecessary to specify addresses.

The ed editor allows editing only the current line unless you address another line in the buffer. You can move and copy only complete lines of data. The ed editor is useful for editing large files or for editing within a shell program.

The ed editor operates in one of two modes:

Item	Description
command mode	In command mode, the ed editor recognizes and runs subcommands. When you start the ed editor, it is in command mode. Type a **.** (period) and press Enter to confirm that you are in command mode.
text input mode	In text input mode, the ed editor allows you to enter text into the file buffer but does not recognize subcommands. You enter text input mode by using the **a**subcommand, **c** subcommand, or **i** subcommand. You exit text input mode and return to the command mode by typing a **.** (period) alone at the beginning of a line. To place a **.** (period) into the buffer while in text input mode, enter a character followed by the **.** (period). Then, exit text input mode and use the **s** subcommand to remove the character.

The following list provides the maximum limits of the **ed** editor.

- 64 characters per file name
- 256 characters per global subcommand list
- 128,000 character buffer size

  Note: The buffer contains the original file as well as editing information.

The maximum number of lines depends on the amount of memory available. The maximum file size depends on the amount of physical data storage (disk or tape drive) available or on the maximum number of lines permitted in user memory.

Flags

Item	Description
-p *String*	Sets the editor prompt to the *String* parameter. The default for *String* is a null value (no prompt).
-s	Suppresses character counts that the editor displays with the **e** subcommand, **r** subcommand, and **w** subcommand. This flag also suppresses diagnostic messages for the **e** subcommand and the **q** subcommand, and suppresses the ! (exclamation point) prompt after an **!** subcommand.
-	Provides the same functions as the **-s** flag.

Pattern Matching

The ed editor supports a limited form of special pattern-matching characters that you can use as regular expressions (REs) to construct pattern strings. You can use these patterns in addresses to specify lines and in some subcommands to specify portions of a line.

Regular Expressions

The following REs match a single character or a collating element as follows:

Item	Description
Character	Matches itself and can be any ordinary character (other than one of the special pattern-matching symbols).
.	Matches any single character except the new-line character.
[*String*]	Matches any one character in the string. Certain pattern-matching characters have special meanings within brackets as follows: **^** Matches any character except the characters in the *String* parameter and the new-line character if the first character of the *String* parameter is a ^ (circumflex). This condition is true only if the ^ is the first character in the string, [^*String*]. **-** Indicates a range of consecutive ASCII characters according to the current collating sequence. For example, [a-f] can be equivalent to [abcdef] or [aAbBcCdDeEfF] or [abcdef] and could even include accented a and e characters. A collating sequence can define equivalence classes for characters. The minus sign loses its significance if it occurs as the first character in the string, [-*String*]; if it immediately follows an initial circumflex, [^-*String*]; or if it appears as the last character in the string, [*String*-]. **]** Functions as a part of the string rather than as the string terminator, when the] (right bracket) is the first character in the string, []*String*], or when it immediately follows an initial circumflex, [^]*String*].

Forming Patterns

The following rules describe how to form patterns from REs:

- An RE that consists of a single, ordinary character matches that same character in a string.

- An RE followed by an * (asterisk) matches zero or more occurrences of the character that the RE matches. For example, the following pattern:

```
ab*cd
```

matches each of the following strings:

```
acd
abcd
abbcd
abbbcd
```

but not the following string:

```
abd
```

If a choice exists, the longest matching leftmost string is chosen. For example, given the following string:

```
122333444
```

the pattern .* matches 122333444, the pattern .*3 matches 122333, and the pattern .*2 matches 122.

- An RE followed by:

Item	Description
\{*m*\}	Matches *exactly m* occurrences of the character matched by the RE.
\{*m*,\}	Matches *at least m* occurrences of the character matched by the RE.
\{*m*,*n*\}	Matches *any number* of occurrences of the character matched by the RE from *m* to *n* inclusive.

The numbers *m* and *n* must be integers from 0 to 255, inclusive. Whenever a choice exists, this pattern matches as many occurrences as possible.

- You can combine REs into patterns that match strings containing that same sequence of characters. For example, the pattern AB*CD matches the string AB*CD, and the pattern [A-Za-z]*[0-9]* matches any string that contains any combination of alphabetic characters (including none), followed by any combination of numerals (including none).
- The character sequence \(*Pattern*\) marks a subpattern that matches the same string the sequence would match if it were not enclosed.
- The characters *Number* match the same string of characters that a subpattern matched earlier in the pattern (see the preceding rule). The pattern of the *Number* parameter represents a digit. The pattern *Number* matches the string matched by the occurrence of the subpattern specified by the *Number* parameter, counting from left to right.

  For example, the following pattern:

```
\(A\)\(B\)C\2\1
```

matches the string ABCBA. You can nest subpatterns.

Restricting What Patterns Match

You can restrict a pattern to match only the first segment of a line, the final segment, or the entire line. The null pattern, // (two slashes), duplicates the previous pattern.

Matching the First Segment of a Line

The ^*Pattern* parameter matches only a string that begins in the first character position on a line.

Matching the Last Segment of a Line

The *Pattern*$ parameter matches only a string that ends with the last character (not including the new-line character) on a line.

Matching the Entire Line

The ^*Pattern*$ parameter restricts the pattern to match an entire line.

Addressing Lines

The ed editor uses three types of addresses: line number addresses, addresses relative to the current line, and pattern addresses. The current line (usually the last line affected by a subcommand) is the point of reference in the buffer.

You can use line addressing to do the following:

- Designate a new current line
- Display the addressed line or lines
- Cause a command to act on a certain line or lines

Subcommands that do not accept addresses regard the presence of an address as an error. Subcommands that accept addresses can use either given or default addresses. When given more addresses than it accepts, a command uses the last (rightmost) ones.

In most cases, commas (,) separate addresses (for example 2 , 8). Semicolons (;) also can separate addresses. A semicolon between addresses causes the ed editor to set the current line to the first address and then calculate the second address (for example, to set the starting line for a search). In a pair of addresses, the first address must be numerically smaller than the second.

You can use line numbers and symbolic addresses to perform the following tasks:

- Addressing the current line
- Addressing a line by number
- Addressing the line before the first line
- Addressing the last line
- Addressing a line above an addressed line
- Addressing a line below an addressed line
- Addressing the first line through the last line
- Addressing the current line through the last line
- Addressing a group of lines
- Addressing the next line that contains a specified pattern
- Addressing the previous line that contains a specified pattern
- Addressing a marked line

Addressing the Current Line

A . (period) addresses the current line. The . (period) is the default for most ed editor subcommands and does not need to be specified.

Addressing a Line by Number

To address a specified line of the buffer, type:

```
Number
```

where the *Number* parameter represents a line number. For example:

```
2253
```

addresses line number 2253 as the current line.

Addressing the Line before the First Line

To address the line before the first line of the buffer, type:

```
0
```

Addressing the Last Line

To address the last line of the buffer, type:

```
$
```

Addressing a Line above an Addressed Line

To specify an address that is a specified number of lines above the current line, type:

```
-Number
```

where the *Number* parameter is the specified number of lines above the current line that you want to address. For example:

```
-5
```

addresses the line five lines above the current line as the current line.

You also can specify only a - to address the line immediately above the current line. The minus sign has a cumulative effect. For example, the address - - (two minus signs) addresses the line two lines above the current line.

Addressing a Line below an Addressed Line

To specify an address that is a specified number of lines below the current line, type:

```
+Number
```

where the *Number* parameter is the specified number of lines below the current line that you want to address. The + (plus sign) is optional. For example:

```
+11
```

addresses the line 11 lines below the current line as the current line.

You also can specify only a + to address the line immediately below the current line. The + has a cumulative effect. For example, the address + + (two plus signs) addresses the line two lines below the current line.

Addressing the First Line through the Last Line

To address the first line through the last line, type:

```
,
```

The , (comma) represents the address pair 1,$ (first line through last line). The first line becomes the current line.

Addressing the Current Line through the Last Line

To address the current line through the last line, type:

```
;
```

The ; (semicolon) represents the address pair .,$ (current line through last line).

Addressing a Group of Lines

To address a group of lines, type:

```
FirstAddress,LastAddress
```

where the *FirstAddress* parameter is the line number (or symbolic address) of the first line in the group you want to address, and the *LastAddress* parameter is the line number (or symbolic address) of the last line in the group. The first line in the group becomes the current line. For example:

```
3421,4456
```

addresses the lines 3421 through 4456. Line 3421 becomes the current line.

Addressing the Next Line That Contains a Specified Pattern

To address the next line that contains a matching string, type:

```
/Pattern/
```

where the *Pattern* parameter is a character string or regular expression. The search begins with the line after the current line and stops when it finds a match for the pattern. If necessary, the search moves to the end of the buffer, wraps around to the beginning of the buffer, and continues until it either finds a match or returns to the current line. For example:

```
/Austin, Texas/
```

addresses the next line that contains Austin, Texas as the current line.

Addressing the Previous Line That Contains a Specified Pattern

To address the previous line that contains a match for the pattern, type:

```
?Pattern?
```

where the *Pattern* parameter is a character string or regular expression. The ?*Pattern*? construction, like /*Pattern*/, can search the entire buffer, but it searches in the opposite direction. For example:

```
?Austin, Texas?
```

addresses the previous line that contains Austin, Texas as the current line.

Addressing a Marked Line

To address a marked line with the **k** subcommand, type:

```
'x
```

where the *x* parameter is a lowercase letter *a* to *z*. For example:

```
'c
```

addresses the line marked as c with the **k** subcommand.

Subcommands

Use the ed editor subcommands to perform the following actions:

- Editing a file
- Manipulating files
- Performing miscellaneous functions
  - Changing the prompt string
  - Entering system commands
  - Exiting the ed editor
  - Requesting help

In most cases, you can enter only one ed editor subcommand on a line. However, you can add the **l** (list) and **p** (print) subcommands to any subcommand except the **e** (edit), **E** (Edit), **f** (file), **q** (quit), **Q** (Quit), **r** (read), **w** (write), and **!** (operating system commands) subcommands.

The **e**, **f**, **r**, and **w** subcommands accept file names as parameters. The ed editor stores the last file name used with a subcommand as a default file name. The next **e**, **E**, **f**, **r**, or **w** subcommand given without a file name uses the default file name.

The ed editor responds to an error condition with one of two messages: ? (question mark) or `?File`. When the ed editor receives an Interrupt signal (the Ctrl-C key sequence), it displays a ? and returns to command mode. When the ed editor reads a file, it discards ASCII null characters and all characters after the last new-line character.

Editing a File

You can use the ed editor subcommands to perform the following tasks:

- Adding text
- Changing text
- Copying text
- Deleting text
- Displaying text
- Joining and splitting lines
- Making global changes
- Marking text
- Moving text
- Saving text
- Searching text
- Substituting text
- Undoing text changes

Note: In the following descriptions of ed editor subcommands, default addresses are shown in parentheses. Do not type the parentheses. The address . (period) refers to the current line. A . (period) in the first position of an otherwise empty line is the signal to return to command mode.

Adding Text

Item	Description
(.)**a** [**l**] [**n**] [**p**] *Text***.**	The **a** (append) subcommand adds text to the buffer *after* the addressed line. The **a** subcommand sets the current line to the last inserted line, or, if no lines were inserted, to the addressed line. A 0 address adds text to the beginning of the buffer. Type the **l** (list), **n** (number), or **p** (print) optional subcommand if you want to display the added text. Type your text, pressing the Enter key at the end of each line. If you do not press Enter at the end of each line, the ed editor automatically moves your cursor to the next line after you fill a line with characters. The ed editor treats everything you type before you press Enter as one line, regardless of how many lines it takes up on the screen. Type a . (period) at the start of a new line, after you have typed all of your text.

Item	Description
(.)**i** [**l**] [**n**] [**p**]*Text***.**	The **i** (insert) subcommand inserts text *before* the addressed line and sets the current line to the last inserted line. If no lines are inserted, the **i** subcommand sets the current line to the addressed line. You cannot use a 0 address for this subcommand. Type the **l** (list), **n** (number), or **p** (print) optional subcommand if you want to display the inserted text. Type your text, pressing the Enter key at the end of each line. If you do not press Enter at the end of each line, the ed editor automatically moves your cursor to the next line after you fill a line with characters. The ed editor treats everything you type before you press Enter as one line, regardless of how many lines it takes up on the screen. Type a . (period) at the start of a new line, after you have typed all of your text. **Note:** The **i** subcommand differs from the **a** subcommand only in the placement of the text.

You can use different ed editor subcommands to add text in different locations. Use the preceding format to perform the following editing tasks:

- Adding text after the current line
- Adding text before the current line
- Adding text after an addressed line
- Adding text before an addressed line
- Adding text after lines that contain a search pattern
- Adding text before lines that contain a search pattern
- Adding text after lines that do not contain a search pattern
- Adding text before lines that do not contain a search pattern

To Add Text after the Current Line

1. Type the following subcommand:

   ```
   a[l][n][p]
   ```

   where **l**, **n**, and **p** are optional subcommands that display the added text.
2. Type the text, and press Enter.
3. Type a **.** (period), and press Enter again to return to command mode.

To Add Text before the Current Line

1. Type the following subcommand:

   ```
   i[l][n][p]
   ```

   where **l**, **n**, and **p** are optional subcommands that display the added text.
2. Type the text, and press Enter.
3. Type a **.** (period), and press Enter again to return to command mode.

To Add Text after an Addressed Line

1. Type the following subcommand:

   ```
   Addressa[l][n][p]
   ```

where the *Address* parameter is the line number of the line that the inserted text should follow. The **l**, **n**, and **p** optional subcommands display the added text.

2. Type the text, and press Enter.
3. Type a **.** (period), and press Enter again to return to command mode.

To Add Text before an Addressed Line

1. Type the following subcommand:

```
Addressi[l][n][p]
```

where the *Address* parameter is the line number of the line that the inserted text should precede. The **l**, **n**, and **p** optional subcommands display the added text.

2. Type the text, and press Enter.
3. Type a **.** (period), and press Enter again to return to command mode.

To Add Text after Lines That Contain a Search Pattern

1. Type the following subcommand:

```
[Address]g/Pattern/a[l][n][p]
```

where *Address* is an optional parameter that specifies the range of lines to search for the pattern specified in the *Pattern* parameter. The *Pattern* parameter is a character string or regular expression. If you omit the *Address* parameter, the ed editor searches the entire file for lines that contain the pattern. The **l**, **n**, and **p** optional subcommands display the added text.

2. Type a backslash:

```
\
```

3. Type the text. To start new lines within the added text, type a backslash:

```
\
```

and press Enter. The text you type is added after every line that contains the pattern specified in the command.

4. To return to command mode, press Enter.

To Add Text before Lines That Contain a Search Pattern

1. Type the following subcommand:

```
[Address]g/Pattern/i[l][n][p]
```

where *Address* is an optional parameter that specifies the range of lines to search for the pattern specified in the *Pattern* parameter. The *Pattern* parameter is a character string or regular expression. If you omit the *Address* parameter, the ed editor searches the entire file for lines that contain the pattern. The **l**, **n**, and **p** optional subcommands display the added text.

2. Type a backslash:

```
\
```

3. Type the text. To start new lines within the added text, type a backslash:

```
\
```

and press Enter. The text you type is added before every line that contains the pattern specified in the command.

4. To return to command mode, press Enter.

To Add Text after Lines That Do Not Contain a Search Pattern

1. Type the following subcommand:

```
[Address]g/Pattern/a[l][n][p]
```

   where *Address* is an optional parameter that specifies the range of lines to search for lines that do not contain the pattern specified in the *Pattern* parameter. The *Pattern* parameter is a character string or regular expression. If you omit the *Address*, the ed editor searches the entire file for lines that do not contain the pattern. The **l**, **n**, and **p** optional subcommands display the added text.
2. Type a backslash:

```
\
```

3. Type the text. To start new lines within the added text, type a backslash:

```
\
```

   and press Enter. The text you type is added after every line that does not contain the pattern specified in the command.
4. To return to command mode, press Enter.

To Add Text before Lines That Do Not Contain a Search Pattern

1. Type the following subcommand:

```
[Address]g/Pattern/i[l][n][p]
```

   where *Address* is an optional parameter that specifies the range of lines to search for lines that do not contain the pattern specified in the *Pattern* parameter. The *Pattern* parameter is a character string or regular expression. If you omit the *Address* parameter, the ed editor searches the entire file for lines that do not contain the pattern. The **l**, **n**, and **p** optional subcommands display the added text.
2. Type a backslash:

```
\
```

3. Type the text. To start new lines within the added text, type a backslash:

```
\
```

   and press Enter. The text you type is added before every line that does not contain the pattern specified in the command.
4. To return to command mode, press Enter.

Changing Text

Item	Description
(.,.)**c** [**l**] [**n**] [**p**]*Text***.**	The **c** (change) subcommand deletes the addressed lines you want to replace and then replaces them with the new lines you enter. The **c** subcommand sets the current line to the last new line of input, or, if no input existed, to the first line that was not deleted. Type the **l** (list), **n** (number), or **p** (print) optional subcommand if you want to display the inserted text. Type the new text, and press Enter at the end of each line. When you have entered all of the new text, type a . (period) on a line by itself.

You can change text in several different ways with the ed editor. Use the preceding format to perform the following editing tasks:

- Changing the text of the current line

- Changing the text of a line or group of lines
- Changing text of lines that contain a specified pattern
- Changing text of lines that do not contain a specified pattern

To Change the Text of the Current LIne

1. Type the following subcommand:

   ```
   c[l][n][p]
   ```

   where **l**, **n**, and **p** are optional subcommands that display the changed text.
2. Type the text, and press Enter.
3. Type a **.** (period), and press Enter again to return to command mode.

To Change the Text of a Line or Group of Lines

1. Type the following subcommand:

   ```
   Addressc[l][n][p]
   ```

   where the *Address* parameter is the address of the line or group of lines to change. The **l**, **n**, and **p** optional subcommands display the changed text.
2. Type the text, and press Enter.
3. Type a **.** (period), and press Enter again to return to command mode.

To Change the Text of Lines That Contain a Specified Pattern

1. Type the following subcommand:

   ```
   Addressg/Pattern/c[l][n][p]
   ```

   where the *Address* parameter is the address of the group of lines that you want to search for the pattern specified with the *Pattern* parameter. The **l**, **n**, and **p** optional subcommands display the changed text.
2. Type a backslash:

   ```
   \
   ```
3. Type the new text. To start new lines within the new text, type a backslash:

   ```
   \
   ```

   and press Enter.
4. To return to command mode, press Enter again, type a . (period), and press Enter again.

To Change the Text of Lines That Do Not Contain a Specified Pattern

1. Type the following subcommand:

   ```
   Addressv/Pattern/c[l][n][p]
   ```

   where the *Address* parameter is the address of the group of lines that you want to search for the pattern specified with the *Pattern* parameter. The **l**, **n**, and **p** optional subcommands display the changed text.
2. Type a backslash:

   ```
   \
   ```
3. Type the new text. To start new lines within the new text, type a backslash:

   ```
   \
   ```

and press Enter.

4. To return to command mode, press Enter again, type a . (period), and press Enter again.

Copying Text

Item	Description
(.,.)**t***Address* [**p**] [**l**] [**n**]	The **t** (transfer) subcommand inserts a copy of the addressed lines after the line specified by the *Address* parameter. The **t** subcommand accepts the 0 address to insert lines at the beginning of the buffer. The **t** subcommand sets the current line to the last line copied. Type the **l** (list), **n** (number), or **p** (print) optional subcommand if you want to display the transferred text.

Copying a line or a set of lines leaves the specified lines in their original location and puts a copy in the new location. You can select the lines to copy by specifying an address or pattern. Use the preceding format to perform the following editing tasks:

- Copying the current line
- Copying lines specified by address
- Copying lines that contain a specified pattern
- Copying lines that do not contain a specified pattern

To Copy the Current Line

1. Type the following subcommand:

   ```
   tAddress[l][n][p]
   ```

   where the *Address* parameter is the line number or symbolic address of the line you want a copy of the current line to follow. The **l**, **n**, and **p** optional subcommands display the copied line.
2. Type the text, and press Enter.
3. Type a **.** (period), and press Enter again to return to command mode.

To Copy Lines Specified by Address

1. Type the following subcommand:

   ```
   LineNumbertDestinationAddress[l][n][p]
   ```

   where the *LineNumber* parameter is the address of the lines you want to copy, and the *DestinationAddress* parameter is the line you want the copy to follow. The **l**, **n**, and **p** optional subcommands display the copied line.
2. Type the text, and press Enter.
3. Type a **.** (period), and press Enter again to return to command mode.

To Copy Lines That Contain a Specified Pattern

Type the following subcommand:

```
[Address]g/Pattern/t[DestinationAddress][l][n][p]
```

where *Address* is an optional parameter that specifies the range of lines to search for lines that contain the specified pattern, the *Pattern* parameter is the text you are searching for, and the *DestinationAddress* is an optional parameter that identifies the line you want the copied text to follow. The **l**, **n**, and **p** optional subcommands display the copied line.

If you omit the *Address* parameter, the ed editor searches the entire file for lines that contain the pattern. If you omit the *DestinationAddress* parameter, the copied text is placed after the current line.

To Copy Lines That Do Not Contain a Specified Pattern

Type the following subcommand:

```
[Address]v/Pattern/t[DestinationAddress][l][n][p]
```

where *Address* is an optional parameter that specifies the range of lines to search for lines that do not contain the specified pattern, the *Pattern* parameter is the text, and the *DestinationAddress* is an optional parameter that identifies the line you want the copied text to follow. The **l**, **n**, and **p** optional subcommands display the copied line.

If you omit the *Address* parameter, the ed editor searches the entire file for lines that do not contain the pattern. If you omit the *DestinationAddress* parameter, the copied text is placed after the current line.

Deleting Text

Item	Description
(.,.)**d** [**l**] [**n**] [**p**]	The **d** (delete) subcommand removes the addressed lines from the buffer. The line after the last line deleted becomes the current line. If the deleted lines were originally at the end of the buffer, the new last line becomes the current line. Type the **l** (list), **n** (number), or **p** (print) optional subcommand if you want to display the deletion.

The ed editor provides several ways to delete text. Use the preceding format to perform the following editing tasks:

- Deleting the current line
- Deleting a line or group of lines
- Deleting a line or group of lines that contain a specified pattern
- Deleting a line or group of lines that does not contain a specified pattern
- Deleting text from the current line
- Deleting text within selected lines
- Deleting text from addressed lines
- Deleting text from lines that contain a specified pattern
- Deleting a pattern from lines that contain a different specified pattern
- Deleting a pattern from lines that do not contain a different specified pattern

To Delete the Current Line

Type the following subcommand:

```
d[l][n][p]
```

where **l**, **n**, and **p** are optional subcommands that display the deleted line.

To Delete a Line or Group of Lines

Type the following subcommand:

```
Addressd[l][n][p]
```

where the *Address* parameter is the line number or symbolic address of the lines you want to delete, and **l**, **n**, and **p** are optional subcommands that display the deleted line or lines.

To Delete a Line or Group of Lines That Contain a Specified Pattern

Type the following subcommand:

```
[Address]g/Pattern/d[l][n][p]
```

where *Address* is an optional parameter that specifies the line number or symbolic address of the lines you want to search, and the *Pattern* parameter is a character string or regular expression that represents the text you want to find. If you omit the *Address* parameter, the ed editor searches the entire file for lines that contain the specified pattern. The **l**, **n**, and **p** optional subcommands display the deleted line or lines.

To Delete a Line or Group of Lines That Does Not Contain a Specified Pattern

Type the following subcommand:

```
[Address]v/Pattern/d[l][n][p]
```

where *Address* is an optional parameter that specifies the line number or symbolic address of the lines you want to search, and the *Pattern* parameter is a character string or regular expression that represents the text you want to find. If you omit the *Address* parameter, the ed editor searches the entire file for lines that do not contain the specified pattern. The **l**, **n**, and **p** optional subcommands display the deleted line or lines.

To Delete Text from the Current Line

1. Type the following subcommand:

   ```
   s/Pattern
   ```

   where the *Pattern* parameter is a character string or regular expression that represents the text you want to delete.
2. To delete the *first instance* of the pattern from the line, type:

   ```
   //
   ```

   OR

   To delete *every instance* of the pattern from the line, type:

   ```
   //g
   ```
3. If you want to display the deletion, type one of the following optional subcommands:

   l

   n

   p
4. Press Enter.

To Delete Text within Selected Lines

1. Type the address of a group of lines to select (or skip this step to select all lines).
2. To select the lines indicated by the *Pattern* parameter in step 4, type:

   ```
   g
   ```

   OR

   To select the lines *not* indicated by the *Pattern* parameter in step 4, type:

   ```
   v
   ```
3. To enter the text you want to search, type the following subcommand:

   ```
   /Pattern/s
   ```

where the *Pattern* parameter is the text you want to search.

4. Type one of the following commands to make the desired deletion:

   To delete the first instance of the *Pattern* parameter within each selected line, type:

   ```
   ///
   ```

   To delete every instance of the *Pattern* parameter within each selected line, type:

   ```
   ///g
   ```

   To delete the first specified number of occurrences of the *Pattern* parameter on each selected line (where the *Number* parameter is an integer), type:

   ```
   ///Number
   ```

   To delete the first character string indicated by the *OtherPattern* parameter within each line selected by the *Pattern* parameter (where the *OtherPattern* parameter is the pattern you want to search), type:

   ```
   /OtherPattern//
   ```

   To delete every instance of the *OtherPattern* parameter within each line selected by the *Pattern* parameter, type:

   ```
   /OtherPattern//g
   ```

   To delete the first specified number of occurrences of the *OtherPattern* parameter on each line selected by the *Pattern* parameter (where the *Number* parameter is an integer), type:

   ```
   /OtherPattern//Number
   ```

5. If you want to display the deletion, type one of the following optional subcommands:

   l

   n

   p

6. Press Enter.

For example, to delete all instances of a pattern from *a range of lines*, type:

```
38,$g/tmp/s/gn
```

The previous example searches all the lines from line 38 to the last line (`38,$`) for the `tmp` character string and deletes every instance (`/g`) of that character string within those lines. It then displays the lines that had text deleted from them and their line numbers (`n`).

To delete all instances of a pattern from *all lines* that contain that pattern, type:

```
g/rem/s///gl
```

The previous example searches the entire file (address parameter is omitted) for all lines that contain (`g`) the `rem` character string. It deletes all instances (`///g`) of the `rem` character string from each of those lines and then displays the lines that had text deleted from them, including the nonprinting characters in those lines (`l`).

To Delete Text from Addressed Lines

1. Type the following subcommand:

   ```
   Addresss/Pattern
   ```

Note: The *Address* parameter is followed by the **s** subcommand, where the *Address* parameter is the line number, range of line numbers, or symbolic address of the lines from which you want to delete the pattern, and the *Pattern* parameter is a character string or regular expression that represents the text you want to delete.

2. To delete the *first instance* of the pattern from each line, type:

```
//
```

   OR

   To delete *every instance* of the pattern from each line, type:

```
//g
```

3. If you want to display the deletion, type one of the following optional subcommands:

   l

   n

   p

4. Press Enter.

To Delete Text from Lines That Contain a Specified Pattern

1. Type the following subcommand:

```
[Address]g/Pattern/s
```

   where *Address* is an optional parameter that specifies the line number, range of line numbers, or symbolic address of the lines that contains a specified pattern, and the *Pattern* parameter is a character string or regular expression that represents the text you want to find and delete. If you omit the *Address* parameter, the ed editor searches all lines in the file for the pattern.

2. To delete the *first instance* of the pattern from each line that contains it, type:

```
///
```

   OR

   To delete *every instance* of the pattern from each line that contains it, type:

```
///g
```

3. If you want to display the deletion, type one of the following optional subcommands:

   l

   n

   p

4. Press Enter.

To Delete a Pattern from Lines That Contain a Different Specified Pattern

1. Type the following subcommand:

```
[Address]g/SearchPattern/s
```

   where *Address* is an optional parameter that specifies the line number, range of line numbers, or symbolic address of the lines that contains a specified pattern, and the *SearchPattern* parameter is a character string or regular expression that represents text that is in the lines you want to change. If you omit the *Address* parameter, the ed editor searches all lines in the file for the specified pattern.

2. To specify the text you want to delete, type:

```
/DeletePattern/
```

3. To delete the *first instance* of the pattern from each line, type:

```
/
```

OR

To delete *every instance* of the pattern from each line, type:

```
/g
```

Note: The entire subcommand string looks like this:

```
[Address]g/SearchPattern/s/DeletePattern//[g]
```

4. If you want to display the deletion, type one of the following optional subcommands:

l

n

p

5. Press Enter.

For example, to delete the first instance of a pattern from lines that contain a different specified pattern, type:

```
1,.g/rem/s/tmp//l
```

The previous example searches from the first line to the current line (`1,.`) for all lines that contain (g) the `rem` character string. It deletes the first instance of the `tmp` character string from each of those lines (/), then displays the lines that had text deleted from them, including the nonprinting characters in those lines (l).

To Delete a Pattern from Lines That Do Not Contain a Different Specified Pattern

1. Type the following subcommand:

```
[Address]v/SearchPattern/s
```

where *Address* is an optional parameter that specifies the line number, range of line numbers, or symbolic address of the lines that contains a specified pattern, and the *SearchPattern* parameter is a character string or regular expression that represents text that is not in the lines you want to find and change. If you omit the *Address* parameter, the ed editor searches all lines in the file for the specified pattern.

2. To specify the text you want to delete, type:

```
/DeletePattern/
```

3. To delete the *first instance* of the pattern, type:

```
/
```

OR

To delete *every instance* of the pattern from each line, type:

```
/g
```

Note: The entire subcommand string looks like this:

```
[Address]v/SearchPattern/s/DeletePattern//[g]
```

4. If you want to display the deletion, type one of the following optional subcommands:

   l

   n

   p

5. Press Enter.

For example, to delete the first instance of a pattern from lines that do not contain a specified pattern, type:

```
1,.v/rem/s/tmp//l
```

The previous example searches from the first line to the current line (`1,.`) for all lines that do not contain (v) the `rem` character string. It deletes the first instance of the `tmp` character string from each of those lines (/), then displays the lines that had text deleted from them, including the nonprinting characters in those lines (`l`).

Displaying Text

Item	Description
(.,.)**l**	The **l** (list) subcommand writes the addressed lines to standard output in a visually unambiguous form and writes the characters `\\\`, `\\a`, `\\b`, `\\f`, `\\r`, `\\t`, and `\\v` in the corresponding escape sequence. The **l**subcommand writes nonprintable characters as one 3-digit octal number, with a preceding \ (backslash) for each byte in the character (most significant byte first). The **l** subcommand wraps long lines, and you can indicate the wrap point by writing the \ (backslash)/new-line character sequence. Wrapping occurs at the 72nd column position. The $ (dollar sign) marks the end of each line. You can append the **l** subcommand to any ed editor subcommand except the **e**, **E**, **f**, **q**, **Q**, **r**, **w**, or **!** subcommand. The current line number is set to the address of the last line written.
(.,.)**n**	The **n** (number) subcommand displays the addressed lines, each preceded by its line number and a tab character (displayed as blank spaces); **n** sets the current line to the last line displayed. You can append the **n** subcommand to any ed editor subcommand except **e**, **f**, **r**, or **w**. For example, the **dn** subcommand deletes the current line and displays the new current line and line number.
(.,.)**p**	The **p** (print) subcommand displays the addressed lines and sets the current line to the last line displayed. You can append the **p** subcommand to any ed editor subcommand except **e**, **f**, **r**, or **w**. For example, the **dp** subcommand deletes the current line and displays the new current line.
(.)**=**	Without an address, the **=** (equal sign) subcommand displays the current line number. When preceded by the $ address, the **=** subcommand displays the number of the last line in the buffer. The **=** subcommand does not change the current line and cannot be appended to a **g** subcommand or **v** subcommand.

When you search for lines that contain or do not contain a specified pattern, you can select a range of line numbers to search. You can select and display one line or a group of lines in an ed editor file several different ways. Use the preceding format to perform the following editing tasks:

- Displaying an addressed line or group of lines
- Displaying an addressed line or group of lines and their nonprinting characters
- Displaying an addressed line or group of lines and their line numbers

- Displaying lines that contain a search pattern
- Displaying lines that contain a search pattern, including their nonprinting characters
- Displaying lines that contain a search pattern, including their line numbers
- Displaying lines that do not contain a search pattern
- Displaying lines that do not contain a search pattern, including their nonprinting characters
- Displaying lines that do not contain a search pattern, including their line numbers

To Display an Addressed Line or Group of Lines

Type the following subcommand:

```
Addressp
```

where the *Address* parameter is the line number or symbolic address of the lines you want to display.

The line or lines addressed are displayed on the screen. If the group of lines is too long to fit on the screen, the ed editor displays as many as will fit, beginning with the first line addressed.

To Display an Addressed Line or Group of Lines and Their Nonprinting Characters

Type the following subcommand:

```
Addressl
```

where the *Address* parameter is the line number or symbolic address of the lines you want to display.

The line or lines addressed and their nonprinting characters are displayed on the screen. If the group of lines is too long to fit on the screen, the ed editor displays as many as will fit, beginning with the first line addressed.

To Display an Addressed Line or Group of Lines and Their Line Numbers

Type the following subcommand:

```
Addressn
```

where the *Address* parameter is the line number or symbolic address of the lines you want to display.

The line or lines addressed are displayed on the screen. The line number for each line is displayed beside the line. If the group of lines is too long to fit on the screen, the ed editor displays as many as will fit, beginning with the first line addressed.

To Display Lines That Contain a Search Pattern

Type the following subcommand:

```
Addressg/Pattern/p
```

where the *Address* parameter is the range of lines and the *Pattern* parameter is the character string or regular expression that you want to search.

The line or lines that contain the specified pattern are displayed on the screen. If the group of lines is too long to fit on the screen, the ed editor displays as many as will fit, beginning with the first line addressed.

To Display Lines That Contain a Search Pattern, Including Their Nonprinting Characters

Type the following subcommand:

```
[Address]g/Pattern/l
```

where *Address* is an optional parameter that specifies the range of lines and the *Pattern* parameter is the character string or regular expression that you want to search. If you omit the *Address* parameter, the ed editor searches the entire file.

The line or lines that contain the specified pattern are displayed on the screen. Nonprinting characters show up in the display. If the group of lines is too long to fit on the screen, the ed editor displays as many as will fit, beginning with the first line addressed.

To Display Lines That Contain a Search Pattern, Including Their Line Numbers

Type the following subcommand:

```
[Address]g/Pattern/n
```

where *Address* is an optional parameter that specifies the range of lines and the *Pattern* parameter is the character string or regular expression that you want to search. If you omit the *Address* parameter, the ed editor searches the entire file.

The line or lines that contain the specified pattern are displayed on the screen. The line number for each line is displayed beside the line. If the group of lines is too long to fit on the screen, the ed editor displays as many as will fit, beginning with the first line addressed.

To Display Lines That Do Not Contain a Search Pattern

Type the following subcommand:

```
[Address]v/Pattern/p
```

where *Address* is an optional parameter that specifies the range of lines and the *Pattern* parameter is the character string or regular expression that you want to search. If you omit the *Address* parameter, the ed editor searches the entire file.

The line or lines that do not contain the specified pattern are displayed on the screen. If the group of lines is too long to fit on the screen, the ed editor displays as many as will fit, beginning with the first line addressed.

To Display Lines That Do Not Contain a Search Pattern, Including Their Nonprinting Characters

Type the following subcommand:

```
[Address]v/Pattern/l
```

where *Address* is an optional parameter that specifies the range of lines and the *Pattern* parameter is the character string or regular expression that you want to search. If you omit the *Address* parameter, the ed editor searches the entire file.

The line or lines that do not contain the specified pattern are displayed on the screen, including the nonprinting characters. If the group of lines is too long to fit on the screen, the ed editor displays as many as will fit, beginning with the first line addressed.

To Display Lines That Do Not Contain a Search Pattern, Including Their Line Numbers

Type the following subcommand:

```
[Address]v/Pattern/n
```

where *Address* is an optional parameter that specifies the range of lines and the *Pattern* parameter is the character string or regular expression that you want to search. If you omit the *Address* parameter, the ed editor searches the entire file.

The line or lines that do not contain the specified pattern are displayed on the screen, along with their line numbers. If the group of lines is too long to fit on the screen, the ed editor displays as many as will fit, beginning with the first line addressed.

Joining and Splitting Lines

Item	Description
(.,.+1)**j** [**l**] [**n**] [**p**]	The **j** (join) subcommand joins contiguous lines by removing the intervening new-line characters. If given only one address, the **j** subcommand does nothing. Type the **l** (list), **n** (number), or **p** (print) subcommand if you want to display the joined lines. These subcommands are optional.

The ed editor provides several ways to join or split a line. Use the preceding format to perform the following editing tasks:

- Joining the current and next lines
- Joining addressed lines
- Splitting the current line
- Splitting an addressed line

To Join the Current and Next Lines

Type the following subcommand:

```
j[l][n][p]
```

where **l**, **n**, and **p** are optional subcommands that display the joined lines.

To Join Addressed Lines

Type the following subcommand:

```
Addressj[l][n][p]
```

where the *Address* parameter is a set of contiguous lines that will form one line, and **l**, **n**, and **p** are optional subcommands that display the joined lines.

To Split the Current Line

1. To split the current line after a specified pattern, type the following subcommand:

   ```
   s/Pattern/Pattern\
   ```

   where the *Pattern* parameter is the character string that you want to split the line after.

   Note: Make sure that both strings represented by the *Pattern* parameter are exactly alike.
2. Press Enter.
3. Type the following backslash:

   ```
   /
   ```
4. To display the split line, type one of the following optional subcommands:

   l

   n

   p
5. Press Enter.

To Split an Addressed Line

1. To split an addressed line after a specified pattern, type the following subcommand:

   ```
   Addresss/Pattern/Pattern\
   ```

where the *Address* parameter is the address of the line to split, and the *Pattern* parameter is the character string to split the line after.

Note: Make sure that both strings represented by the *Pattern* parameter are exactly alike.

2. Press Enter.
3. Type the following backslash:

```
/
```

4. To display the split line, type one of the following optional subcommands:

   l

   n

   p

5. Press Enter.

Making Global Changes

Item	Description
(1,$)**g/***Pattern***/***SubcommandList* [**l**] [**n**] [**p**]	The **g** (global) subcommand first marks every line that matches the *Pattern* parameter. The pattern can be a fixed character string or a regular expression. Then, for each marked line, this subcommand sets the current line to the marked line and runs the *SubcommandList* parameter. Enter a single subcommand or the first subcommand of a list of subcommands on the same line with the **g** subcommand; enter subsequent subcommands on separate lines. Except for the last line, each of the lines should end with a \ (backslash). The *SubcommandList* parameter can include the **a**, **i**, and **c** subcommands and their input. If the last command in the *SubcommandList* parameter would usually be the . (period) that ends input mode, the . (period) is optional. If no *SubcommandList* parameter exists, the current line is displayed. The *SubcommandList* parameter cannot include the **g** , **G**, **v**, or **V** subcommand. Type the **l** (list), **n** (number), or **p** (print) subcommand if you want to display the changes. These subcommands are optional. **Note:** The **g** subcommand is similar to the **v** subcommand, which runs the *SubcommandList* parameter for every line that does not contain a match for the pattern.
(1,$)**G/***Pattern***/** [**l**] [**n**] [**p**]	The interactive **G** (Global) subcommand marks every line that matches the *Pattern* parameter, displays the first marked line, sets the current line to that line, and then waits for a subcommand. A pattern can be a fixed character string or a regular expression. The **G** subcommand does not accept the **a**, **i**, **c**, **g**, **G**, **v**, and **V** subcommands. After the subcommand finishes, the **G** subcommand displays the next marked line, and so on. The **G** subcommand takes a new-line character as a null subcommand. A :& (colon ampersand) causes the **G** subcommand to run the previous subcommand again. You can stop the **G** subcommand by pressing Ctrl+C. Type the **l** (list), **n** (number), or **p** (print) subcommand if you want to display the changes. These subcommands are optional.

Item	Description
(1,$)**v/***Pattern***/***SubcommandList* [**l**] [**n**] [**p**]	The **v** subcommand runs the subcommands in the *SubcommandList* parameter for each line that does not contain a match for the *Pattern* parameter. A pattern can be a fixed character string or a regular expression. Type the **l** (list), **n** (number), or **p** (print) subcommand if you want to display the changes. These subcommands are optional. The **v** subcommand does not accept the **a**, **i**, **c**, **g**, **G**, and **V** subcommands. **Note:** The **v** subcommand complements the **g** subcommand, which runs the *SubcommandList* parameter for every line that contains a match for the pattern.
(1,$)**V/***Pattern*/ [**l**] [**n**] [**p**]	The **V** subcommand marks every line that does not match the *Pattern* parameter, displays the first marked line, sets the current line to that line, and then waits for a subcommand. A pattern can be a fixed character string or a regular expression. Type the **l** (list), **n** (number), or **p** (print) subcommand if you want to display the changes. These subcommands are optional. The **V** subcommand does not accept the **a**, **i**, **c**, **g**, **G**, and **v** subcommands. **Note:** The **V** subcommand complements the **G** subcommand, which marks the lines that match the pattern.

Marking Text

Item	Description
(.)**k***x* [**l**] [**n**] [**p**]	The **k** (mark) subcommand marks the addressed line with the name specified by the *x* parameter, which must be a lowercase ASCII letter. The address '*x* (single quotation mark before the marking character) then addresses this line. The **k** subcommand does not change the current line. Type the **l** (list), **n** (number), or **p** (print) subcommand if you want to display the marked text. These subcommands are optional.

To Mark the Current Line

Type the following subcommand:

```
kLetter[l][n][p]
```

where the *Letter* parameter is the letter *a* through *z* for a mark, and **l**, **n**, and **p** are optional subcommands that display the marked text.

To Mark an Addressed Line

Type the following subcommand:

```
AddresskLetter[l][n][p]
```

where the *Address* parameter is the line number or symbolic address of the line you want to mark, and the *Letter* parameter is the letter *a* through *z* for a mark. The **l**, **n**, and **p** optional subcommands display the marked text.

Moving Text

Item	Description
(.,.)**m***A* [**l**] [**n**] [**p**]	The **m** (move) subcommand repositions the addressed line or lines. The first moved line follows the line addressed by the *A* parameter. A parameter of 0 moves the addressed line or lines to the beginning of the file. The address specified by the *A* parameter cannot be one of the lines to be moved. The **m** subcommand sets the current line to the last moved line. Type the **l** (list), **n** (number), or **p** (print) subcommands if you want to display the deletion. These subcommands are optional.

Moving a line or a set of lines deletes the specified lines from their original location and places them in a new location. You can select which lines to move by address or pattern. Use the preceding format to perform the following editing tasks:

- Moving the current line
- Moving lines specified by address
- Moving lines that contain a specified pattern
- Moving lines that do not contain a specified pattern

To Move the Current Line

Type the following subcommand:

```
mAddress[l][n][p]
```

where the *Address* parameter is the line number or symbolic address of the line you want the current line to follow, and **l**, **n**, and **p** are optional subcommands that display the moved line.

To Move Lines Specified by Address

Type the following subcommand:

```
LineNumbermDestinationAddress[l][n][p]
```

where the *LineNumber* parameter is the address of the lines you want to move, and the *DestinationAddress* parameter is the line you want the moved lines to follow. The **l**, **n**, and **p** optional subcommands display the moved lines.

To Move Lines That Contain a Specified Pattern

Type the following subcommand:

```
[Address]g/Pattern/m[DestinationAddress][l][n][p]
```

where *Address* is an optional parameter that specifies the range of lines to search for lines that contain the specified pattern, the *Pattern* parameter is the text you are searching for, and *DestinationAddress* is an optional parameter that represents the line you want the moved lines to follow. The **l**, **n**, and **p** optional subcommands display the moved lines.

If you omit the *Address* parameter, the ed editor searches the entire file for lines that contain the pattern. If you omit the *DestinationAddress* parameter, the moved text is placed after the current line.

To Move Lines That Do Not Contain a Specified Pattern

Type the following subcommand:

```
[Address]v/Pattern/m[DestinationAddress][l][n][p]
```

where *Address* is an optional parameter that specifies the range of lines to search for lines that do not contain the specified pattern, the *Pattern* parameter is the text, and *DestinationAddress* is an

optional parameter that represents the line you want the moved text to follow. The **l**, **n**, and **p** optional subcommands display the moved lines.

If you omit the *Address* parameter, the ed editor searches the entire file for lines that do not contain the pattern. If you omit the *DestinationAddress* parameter, the moved text is placed after the current line.

Saving Text

Item	Description
(1,$)**w** *File*	The **w** (write) subcommand copies the addressed lines from the buffer to the file specified by the *File* parameter. If the file does not exist, the **w** subcommand creates it with permission code 666 (read and write permission for everyone), unless the **umask** setting specifies another file creation mode. The **w** subcommand does not change the default file name (unless the *File* parameter is the first file name used since you started the ed editor). If you do not provide a file name, the **w** subcommand uses the default file name. The **w** subcommand does not change the current line. If the ed editor successfully writes the file from the buffer, it displays the number of characters written. If you specify the **!** *Command* subcommand instead of a file name, the **w** subcommand reads the output of the operating system command specified by the *Command* parameter. The **w** subcommand does not save the name of the operating system command you specified as a default file name. **Note:** Because 0 is not a legal address for the **w** subcommand, you cannot create an empty file with the **ed** command.

You can save changes to a file in several ways. Use the preceding format to perform the following actions:

- Saving a file to the current file
- Saving part of a file to the current file
- Saving a file to a different file
- Saving part of a file to a different file

To Save a File to the Current File

Type the following subcommand:

```
w
```

The current file is saved under its current name, and the ed editor displays the number of characters written.

To Save Part of a File to the Current File

Type the following subcommand:

```
Addressw
```

where the *Address* parameter specifies the line or group of lines to write. The ed editor displays the number of characters written.

To Save a File to a Different File

Type the following subcommand:

```
w File
```

where the *File* parameter is the name of the file to write to.

The current file is saved to the file specified by the *File* parameter. The ed editor displays the number of characters written.

To Save Part of a File to a Different File

Type the following subcommand:

```
Addressw File
```

where the *Address* parameter specifies the line or group of lines to write and the *File* parameter specifies the file to write to.

The specified lines are saved to the file specified by the *File* parameter. The ed editor displays the number of characters written.

Searching Text

You can search forward or backward from the current line for a pattern of text. The pattern can be a character string or a regular expression made up of literal characters and the special characters ^ (circumflex), $ (dollar sign), . (period), [(left bracket),] (right bracket), * (asterisk), \ (backslash), % (percent sign), and the & key.

You can use the ed editor to perform the following text searches:

- Searching forward
- Searching backward
- Repeating a search in the same direction
- Repeating a search in the opposite direction

To Search Forward

Type the following subcommand:

```
/Pattern
```

where the *Pattern* parameter is a character string or regular expression that specifies the text to search for.

The cursor moves to the first character of the text specified by the pattern.

To Search Backward

Type the following subcommand:

```
?Pattern
```

where the *Pattern* parameter is a character string or regular expression that specifies the text to search for.

The cursor moves to the first character of the text specified by the pattern.

To Repeat a Search in the Same Direction

Type the following subcommand:

```
/
```

The cursor moves to the first character of the closest instance of the text specified by the pattern in the last search command.

To Repeat a Search in the Opposite Direction

Type the following subcommand:

```
?
```

The cursor moves to the first character of the closest instance of the text specified by the pattern in the last search command.

Substituting Text

Item	Description
(.,.)**s/***Pattern***/***Replacement***/** [**l**] [**n**] [**p**] (.,.)**s/***Pattern***/** *Replacement***/ng** [**l**] [**n**] [**p**]	The **s** (substitute) subcommand searches each addressed line for a string that matches the *Pattern* parameter and replaces the string with the specified *Replacement* parameter. A pattern can be a fixed character string or a regular expression. Without the global subcommand (**g**), the **s** subcommand replaces only the first matching string on each addressed line. With the **g** subcommand, the **s** subcommand replaces every occurrence of the matching string on each addressed line. If the **s** subcommand does not find a match for the pattern, it returns the error message ? (question mark). Type the **l** (list), **n** (number), or **p** (print) subcommand to display the substituted text. These subcommands are optional. **Note:** Any character except a space or a new-line character can separate (delimit) the *Pattern* and *Replacement* parameters. The **s** subcommand sets the current line to the last line changed. If the *Number* parameter (an integer) is specified, then the first number that matches strings in each addressed line is replaced. An & (ampersand) character used in the *Replacement* parameter has the same value as the *Pattern* parameter. For example, the subcommand **s/are/&n't/** has the same effect as the subcommand **s/are/aren't/** and replaces **are** with **aren't** on the current line. A \& (backslash, ampersand) removes the special meaning of the & character in the *Replacement* parameter. A subpattern is part of a pattern enclosed by the strings \((backslash, left parenthesis) and \) (backslash, right parenthesis); the pattern works as if the enclosing characters were not present. In the *Replacement* parameter, *Number* refers to strings that match subpatterns. For example, the **s/\(t\)\(h\) \(e\)/t\1\2ose)** subcommand replaces **the** with **those** if a match for the pattern **the** exists on the current line. Whether subpatterns are nested or in a series, *Number* refers to the occurrence specified by the *Number* parameter, counting from the left of the delimiting characters, \) (backslash, right parenthesis). The % (percent sign), when used alone as the *Replacement* parameter, causes the **s** subcommand to repeat the previous *Replacement* parameter. The % does not have this special meaning if it is part of a longer *Replacement* parameter or if it is preceded by a \ (backslash). You can split lines by substituting new-line characters into them. In the *Replacement* parameter. Pressing the \+Enter key sequence quotes the new-line character (not displayed) and moves the cursor to the next line for the remainder of the string. New-line characters cannot be substituted as part of a **g** subcommand or **v** subcommand list.

The ed editor provides several ways to substitute text. Use the preceding format to perform the following editing tasks:

- Substituting text within the current line
- Substituting text within an addressed line or group of lines

- Substituting a specified pattern within lines that contain that pattern
- Substituting a pattern within lines that contain a different pattern
- Substituting a pattern within lines that do not contain a different pattern

To Substitute Text within the Current Line

1. Type the following subcommand:

   ```
   s/OldString/NewString
   ```

   where the *OldString* parameter is the existing text and the *NewString* parameter is the text you want to substitute for it.
2. Type one of the following actions:

   To substitute the *NewString* parameter for the first instance of the *OldString* parameter within the current line, type:

   ```
   /
   ```

   To substitute the *NewString* parameter for every instance of the *OldPattern* parameter within the current line, type:

   ```
   /g
   ```
3. To display the changed text, type one of the following optional subcommands:

   l

   n

   p
4. Press Enter.

To Substitute Text within an Addressed Line or Group of Lines

1. Type the following subcommand:

   ```
   Addresss/OldPattern/NewString
   ```

   where the *Address* parameter is the address of the line or group of lines where you want to substitute text, the *OldPattern* parameter is the existing text, and the *NewString* parameter is the text you want to substitute.
2. Type one of the following actions:

   To substitute the *NewString* parameter for the first instance of the *OldPattern* parameter within each line, type:

   ```
   /NewString/
   ```

   To substitute the *NewString* parameter for every instance of the *OldPattern* parameter within each line, type:

   ```
   /NewString/g
   ```

   To substitute the *NewString* parameter for the first instance of the *NumberOldPattern* parameter on each address line, type:

   ```
   /NewString/Number
   ```
3. To display the changed text, type one of the following optional subcommands:

   l

n

p

4. Press Enter.

To Substitute a Specified Pattern within Lines That Contain That Pattern

1. Type the following subcommand:

```
Addressg/Pattern/s//NewString
```

where the *Address* parameter is the address of the group of lines that you want to search for the pattern specified with the *Pattern* parameter, and the *NewString* parameter is the text you want to substitute for the *Pattern* parameter.

2. Type one of the following actions:

To substitute the *NewString* parameter for the first instance of the *Pattern* parameter within each line, type:

```
/
```

To substitute the *NewString* parameter for every instance of the *Pattern* parameter within each line, type:

```
/g
```

3. To display the changed text, type one of the following optional subcommands:

l

n

p

4. Press Enter.

To Substitute a Pattern within Lines That Contain a Different Pattern

1. Type the following subcommand:

```
Addressg/Pattern/s/OldString/NewString
```

where the *Address* parameter is the address of the group of lines that you want to search for the pattern specified with the *Pattern* parameter, the *OldString* parameter is the text you want to replace, and the *NewString* parameter is the text you want to substitute in place of the *OldString* parameter.

2. Type one of the following actions:

To substitute the *NewString* parameter for the first instance of the *OldString* parameter within each line that contains the *Pattern* parameter, type:

```
/
```

To substitute the *NewString* parameter for every instance of the *OldString* parameter within each line that contains the *Pattern* parameter, type:

```
/g
```

3. To display the changed text, type one of the following optional subcommands:

l

n

p

4. Press Enter.

To Substitute a Pattern within Lines That Do Not Contain a Different Pattern

1. Type the following subcommand:

```
Addressv/Pattern/s/OldString/NewString
```

   where the *Address* parameter is the address of the group of lines that you want to search for the pattern specified with the *Pattern* parameter, the *OldString* parameter is the text you want to replace, and the *NewString* parameter is the text you want to substitute in place of the *OldString* parameter.
2. Type one of the following actions:

   To substitute the *NewString* parameter for the first instance of the *OldString* parameter within each line that does not contain the *Pattern* parameter, type:

```
/
```

   To substitute the *NewString* parameter for every instance of the *OldString* parameter within each line that does not contain the *Pattern* parameter, type:

```
/g
```

3. To display the changed text, type one of the following optional subcommands:

   l

   n

   p
4. Press Enter.

Undoing Text Changes

Item	Description
u [**l**] [**n**] [**p**]	The **u** (undo) subcommand restores the buffer to the state it was in before it was last modified by an ed editor subcommand. The **u** subcommand cannot undo the **e**, **f**, and **w** subcommands. Type the **l** (list), **n** (number), or **p** (print) subcommand if you want to display the changes. These subcommands are optional.

To Undo Text Changes

Type the following subcommand:

```
u[l][n][p]
```

where **l**, **n**, and **p** are optional subcommands that display the changes. All add, change, move, copy, or delete editing functions performed to the text after the last save are undone.

Manipulating Files

You can use ed editor subcommands to manipulate files to perform the following tasks:

- Adding another file to the current file
- Changing the default file name
- Editing additional files

Adding Another File to the Current File

Item	Description
($)**r** *File*	The **r** (read) subcommand reads a file into the buffer after the addressed line. The **r** subcommand does not delete the previous contents of the buffer. When entered without the *File* parameter, the **r** subcommand reads the default file, if any, into the buffer. The **r** subcommand does not change the default file name. A 0 address causes the **r** subcommand to read a file in at the beginning of the buffer. After it reads a file successfully, the **r** subcommand displays the number of characters read into the buffer and sets the current line to the last line read. If ! (exclamation point) replaces the *File* parameter in an **r** subcommand, the rest of the line is taken as an operating system shell command whose output is to be read. The **r** subcommand does not store the names of operating system commands as default file names.

To Insert a File after the Current Line

Type the following subcommand:

```
r File
```

where the *File* parameter is the name of the file to be inserted.

The ed editor reads the file specified by the *File* parameter into the current file after the current line and displays the number of characters read into the current file.

To Insert a File after a Line Specified by Address

Type the following subcommand:

```
Addressr File
```

where the *Address* parameter specifies the line that you want the inserted file to follow, and the *File* parameter is the name of the file to be inserted.

The ed editor reads the file specified by the *File* parameter into the current file after the specified line and displays the number of characters read into the current file.

Changing the Default File Name

Item	Description
f [*File*]	The **f** (file name) subcommand changes the default file name (the stored name of the last file used) to the name specified by the *File* parameter. If a *File* parameter is not specified, the **f** subcommand displays the default file name. (The **e** subcommand stores the default file name.)

To Display the Name of a File

Type the following subcommand:

```
f
```

The ed editor displays the name of the file in the edit buffer.

To Name a File

Type the following subcommand:

```
f File
```

where the *File* parameter is the new name for the file in the edit buffer.

The file in the edit buffer is renamed.

Editing Additional Files

Item	Description
e *File*	The **e** (edit) subcommand first deletes any contents from the buffer, sets the current line to the last line of the buffer, and displays the number of characters read into the buffer. If the buffer has been changed since its contents were saved (with the **w** subcommand), the ed editor displays a ? (question mark) before it clears the buffer. The **e** subcommand stores the *File* parameter as the default file name to be used, if necessary, by subsequent **e**, **r**, or **w** subcommands. (To change the name of the default file name, use the **f** subcommand.) When an ! (exclamation point) replaces the *File* parameter, the **e** subcommand takes the rest of the line as an operating system shell command and reads the command output. The **e** subcommand does not store the name of the shell command as a default file name.
E *File*	The **E** (Edit) subcommand works like the **e** subcommand with one exception; the **E** subcommand does not check for changes made to the buffer after the last **w** subcommand. Any changes you made before re-editing the file are lost.

You can use the **e** or **E** subcommands to perform the following tasks:

- Re-editing the current file without saving it
- Re-editing the current file after saving it
- Editing a file after the current file Is saved
- Editing a file without saving the current file

To Re-Edit the Current File without Saving It

Type the following subcommand:

```
E
```

The ed editor displays the number of characters in the file. Any changes you made before re-editing the file are lost.

To Re-Edit the Current File after Saving It

Type the following subcommand:

```
e
```

The ed editor displays the number of characters in the file.

To Edit a File after the Current File Is Saved

Type the following subcommand:

```
e File
```

where the *File* parameter is the name of a new or existing file that you want to edit.

For an existing file, the ed editor displays the number of characters in the file. For a new file, the ed editor displays a ? (question mark) and the name of the file.

To Edit a File without Saving the Current File

Type the following subcommand:

```
E File
```

where the *File* parameter is the name of a new or existing file that you want to edit.

For an existing file, the editor displays the number of characters in the file. For a new file, the ed editor displays a ? (question mark) and the name of the file.

Miscellaneous Functions of the ed Editor Subcommands

You can use ed editor subcommands to perform the following tasks:

- Changing the prompt string
- Entering system commands
- Exiting the ed editor
- Requesting help

Changing the Prompt String

Item	Description
P	The **P** (Prompt) subcommand turns on or off the ed editor prompt string, which is represented by an * (asterisk). Initially, the **P** subcommand is turned off.

To Start or Stop Displaying the Prompt String

Type the following subcommand:

```
P
```

The ed editor prompt, an * (asterisk), is displayed or not displayed, depending on its previous setting.

Entering System Commands

Item	Description
! *Command*	The **!** subcommand allows you to run operating system commands without leaving the ed editor. Anything that follows the **!** subcommand on an ed editor subcommand line is interpreted as an operating system command. Within the text of that command string, the ed editor replaces the unescaped % (percent sign) with the current file name, if one exists. You can repeat the previous operating system command by entering an ! (exclamation point) after the **!** ed editor subcommand. If the operating system command interpreter (the **sh** command) expands the command string, the ed editor echoes the expanded line. The **!** subcommand does not change the current line.

You can use the **!** subcommand to perform the following actions:

- Running one operating system command
- Repeating an operating system command
- Running several operating system commands

To Run One Operating System Command

Type the following subcommand:

```
!Command
```

where the *Command* parameter specifies an operating system command usually entered at the prompt.

The command runs and displays its output. After the command completes, the editor displays an **!** (exclamation point).

To Repeat an Operating System Command

Type the following subcommand:

```
!
```

The previously run operating system command runs and displays its output. After the command completes, the editor displays an **!** (exclamation point).

To Run Several Operating System Commands

1. Type the following subcommand to display an operating system prompt:

   ```
   !sh
   ```

2. Type an operating system command.
3. Press Enter to run the command and display its output.
4. Repeat steps 2 and 3 to run more operating system commands.
5. Press Ctrl+D to return to command mode. The editor displays an **!** (exclamation point).

Exiting the ed Editor

Item	Description
q	The **q** (quit) subcommand exits the ed editor after checking whether the buffer has been saved to a file after the last changes were entered. If the buffer has not been saved to a file, the **q** subcommand displays the ? (question mark) message. Enter the **q** subcommand again to exit the ed editor anyway. The changes to the current file are lost.
Q	The **Q** (Quit) subcommand exits the ed editor without checking whether any changes were made since the buffer was saved to a file. Any changes made to the buffer since the last save are lost.

To Quit after Checking for Edits

1. Type the following subcommand:

   ```
   q
   ```

2. If the ed editor displays a ?, type one of the following subcommands:

   To save changes before quitting, type:

   ```
   w
   ```

   then press Enter.

   To quit without saving changes, type:

   ```
   q
   ```

3. Press Enter.

To Quit and Discard Edits

1. Type the following subcommand:

   ```
   Q
   ```

2. Press Enter. Any changes made to the buffer since the last save are lost.

Requesting Help

Item	Description
h	The **h** (help) subcommand provides a brief help message for the most recent ? diagnostic or error message displayed.
H	The **H** (Help) subcommand causes the ed editor to display help messages for all subsequent ? diagnostic messages. The **H** subcommand also explains the previous ? if one existed. The **H** subcommand alternately turns this mode on and off; it is initially off.

To Start or Stop Displaying Help Messages

Type the following subcommand:

```
H
```

The help messages are displayed or not displayed for ? responses from the ed editor, depending on the previous setting.

To Display the Last Help Message

Type the following subcommand:

```
h
```

A help message is displayed for the last ? response from the ed editor.

Character Class Support in the ed Editor

In standard *Patterns* expression, a range expression matches the set of all characters that fall between two characters in the collation sequence of the current locale. The syntax of the range expression is as follows:

```
[character-character]
```

The first character must be lower than or equal to the second character in the collation sequence. For example, [a-c] matches any of the characters a, b, or c in the En_US locale.

The range expression is commonly used to match a character class. For example, [0-9] is used to mean all digits, and [a-z A-Z] is used to mean all letters. This form may produce unexpected results when ranges are interpreted according to the collating sequence in the current locale.

Instead of the preceding form, use a character class expression within [ ] (brackets) to match characters. The system interprets this type of expression according to the character class definition in the current locale. However, you cannot use character class expressions in range expressions.

The syntax of a character class expression is as follows:

```
[:CharacterClass:]
```

That is, a left bracket, a colon, the name of the character class, another colon, and then a right bracket.

The following character classes are supported in all locales:

Item	Description
[:upper:]	Uppercase letters
[:lower:]	Lowercase letters
[:alpha:]	Uppercase and lowercase letters
[:digit:]	Digits
[:alnum:]	Alphanumeric characters
[:xdigit:]	Hexadecimal digits
[:punct:]	Punctuation character (neither a control character nor alphanumeric)
[:space:]	Space, tab, carriage return, new-line, vertical tab, or form feed character
[:print:]	Printable characters, including space
[:graph:]	Printable characters, not including space
[:cntrl:]	Control characters
[:blank:]	Space and tab characters

The brackets are part of the character class definition. To match any uppercase ASCII letter or ASCII digit, use the following regular expression:

```
[[:upper:] [:digit:]]
```

Do not use the expression `[A-Z0-9]`.

A locale may support additional character classes.

The newline character is part of the `[:space:]` character class but will not be matched by this character class. The newline character may only be matched by the special search characters $ (dollar sign) and ^ (caret).

Exit Status

The **ed** and **red** commands return the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

edit Command

Purpose

Provides a simple line editor for the new user.

Syntax

edit [**-r**] [*File* ...]

Description

The **edit** command starts a line editor designed for beginning users, a simplified version of the ex editor. The edit editor belongs to a family of editors that includes the ed editor, ex editor, and vi editor. Knowing about the edit editor can help you learn the more advanced features of the other editors. To edit the contents of a file, enter:

```
edit  File
```

When the file specified by the *File* parameter names an existing file, the **edit** command copies it to a buffer and displays the number of lines and characters in it. It then displays a **:** (colon) prompt to show that it is ready to read subcommands from standard input.

If the file specified in the *File* parameter does not already exist, the **edit** command indicates this information and creates the new file. You can specify more than one file name for the *File* parameter, in which case the **edit** command copies the first file into its buffer and stores the remaining file names in an argument list for later use. The edit editor does not make changes to the edited file until you use the **w** subcommand to write the changes.

The edit editor operates in one of the following two modes:

Item	Description
command mode	Recognizes and runs the edit editor subcommands. When you start the edit editor, it is in command mode. To enter command mode at other times, enter only a **.** (period) at the beginning of a line.

Item	Description
text input mode	Allows you to enter text into the edit editor buffer. Enter text input mode by using the **append (a)** subcommand, **change (c)** subcommand, or **insert (i)** subcommand. To end text input mode, enter only a **.** (period) at the beginning of a line.

Flags

Item	Description
-r	Recovers the file being edited after an editor or system malfunction.

Addressing Lines in a File

The edit editor uses the following three types of addresses:

- Line number addresses
- Relative position addresses
- Pattern addresses

Line Number Addresses

Line number addresses specify a line within a file by its line number or symbolic name. This method is the simplest way to address a line or lines.

To address the first line by its symbolic name, enter:

```
.
```

To address the last line by its symbolic name, enter:

```
$
```

You also can specify a range of lines by separating the line numbers or symbolic addresses with a comma or a semicolon. The second address must refer to a line that follows the first addressed line in the range.

For example:

```
1,5
```

addresses the lines 1 through 5.

```
.,$
```

addresses the first through the last lines.

Relative Position Addresses

The edit editor can address a line by its relative position to the current line. An address that begins with the -*Number* or +*Number* parameter addresses a line the specified number of lines before or after the current line, respectively.

For example:

```
+8
```

addresses 8 lines after the current line.

You can also address a line relative to the first or last line by using the symbolic names in combination with the -*Number* or +*Number* addresses.

For example:

```
.+3
```

addresses 3 lines after the first line, and:

```
$-10
```

addresses 10 lines before the last line.

Pattern Addresses

You can specify an address line by searching the buffer for a particular pattern. The edit editor searches forward or backward and stops at the first line that contains the match for the *Pattern* parameter. If necessary, the search wraps past the end or beginning of the buffer until it finds a match or returns to the current line.

To search forward, enter:

```
/Pattern/
```

To search backward, enter:

```
?Pattern?
```

You also can specify a range of lines by separating the *Pattern* parameters with a comma or a semicolon. The second address must refer to a line that follows the first addressed line in the range.

For example:

```
Pattern,Pattern
```

The following characters have special meanings when used as part of the *Pattern* parameter:

Item	Description
^	Matches the beginning of a line when used as the first character of the *Pattern* parameter.
$	Matches the end of a line when used as the last character of the *Pattern* parameter.

Using edit Editor Subcommands

The edit editor subcommands affect the current line, which is represented by a **.** (period). When you start the edit editor, the current line is the last line in the buffer. As the buffer is edited, the current line changes to the last line affected by a subcommand. To work with different parts of a file, you must know how to find the current line and how to address different lines in a file.

You can use the edit editor subcommands to perform the following tasks:

- Adding text
- Changing the name of the current file
- Changing text
- Deleting text
- Displaying the current file name and status
- Displaying text and finding the current line
- Editing additional files
- Ending and exiting the edit editor
- Making global changes
- Moving or copying text
- Saving a file after a system crash

- Saving text
- Substituting text
- Undoing a change

Adding Text

In the following subcommands, the *Address* parameter is optional. If you specify an address, do not type the brackets. You can use the full subcommand or its abbreviation, which is shown in parentheses.

Item	Description
[*Address*]**append** (**a**) *Text* **.**	Appends the text you type after the current line if you do not specify an *Address* parameter. You may need to find the current line or specify an address if you are not in the correct position in the buffer. If you specify an address, the **a** subcommand appends text after the specified line. If you specify a 0 address, the **a** subcommand places the text at the beginning of the buffer. Type the text, pressing the Enter key at the end of each line. When you have entered all the text, type a **.** (period) alone at the start of a line to end text input mode and return to command mode. You can use the **1,$p** subcommand to display the entire contents of the buffer. **Note:** The **a** subcommand differs from the **i** subcommand in the placement of text.
[*Address*]**insert** (**i**)*Text***.**	Inserts text before the current line if you do not specify an *Address* parameter. You may need to find the current line or specify an address if you are not in the correct position in the buffer. If you specify an address, the **i** subcommand inserts text before the specified line. You cannot specify a 0 address. Type your text, pressing the Enter key at the end of each line. When you have entered all your text, type a **.** (period) alone at the start of a line to end text input mode and return to command mode. You can use the **1,$p** subcommand to display the entire contents of the buffer. **Note:** The **i** subcommand differs from the **a** subcommand in the placement of text.

Changing the Name of the Current File

Item	Description
file *File*	Changes the name of the current file to the name specified by the *File* parameter. The edit editor does not consider this file to be edited.

Changing Text

In the following subcommand, the *Address* parameters are optional. If you specify an address, do not type the brackets. You can use the full subcommand or its abbreviation, which is shown in parentheses.

Item	Description
[*Address1*,*Address2*]**change** (**c**). *Text*	Replaces the current line with the text you type if you do not specify the *Address* parameters. You may need to find the current line or specify an address if you are not in the correct position in the buffer. If you specify an address, the **c** subcommand replaces the addressed line or lines. You can specify a range of lines by separating the addresses with a comma. Type your text, pressing the Enter key at the end of each line. When you have entered all your text, type a **.** (period) alone at the start of a line to end text input mode and return to command mode. You can use the **1,$p** subcommand to display the entire contents of the buffer. The last input line becomes the current line.

Deleting Text

In the following subcommand, the *Address* and *Buffer* parameters are optional. If you specify an address or buffer, do not type the brackets. You can use the full subcommand or its abbreviation, which is shown in parentheses.

Item	Description
[*Address1*,*Address2*] **delete** [*Buffer*] (**d**)	Deletes the current line if you do not specify the *Address* parameters. You may need to find the current line or specify an address if you are not in the correct position in the buffer. If you specify an address, the **d** subcommand deletes the addressed line or lines. You can specify a range of lines by separating the addresses with a comma. The line following the last deleted line becomes the current line. If you specify a buffer by giving a lowercase letter from a to z, the edit editor saves the addressed lines in that buffer. If you specify an uppercase letter, the ed editor appends the lines to that buffer. You can use the **pu** subcommand to put the deleted lines back into the buffer.

Displaying the Current File Name and Status

In the following subcommand, you can use the full subcommand or its abbreviation, which is shown in parentheses.

Item	Description
file (**f**)	Displays the current file name along with the following related information: • Whether the file was modified since the last **w** subcommand • Current line number • Number of lines in the buffer • Percentage of the buffer indicating the current line location

Displaying Text and Finding the Current Line

In the following subcommands, the *Address* parameters are optional. If you specify an address, do not type the brackets. You can use either the full subcommand or its abbreviation, which is shown in parentheses.

Item	Description
[*Address1*,*Address2*]**number** (**nu**)	Displays the addressed line or lines preceded by its buffer line number. If you do not specify the *Address* parameters, the **nu** subcommand displays the current line and number. If you specify an address, the **nu** subcommand displays the addressed line or lines. You can specify a range of lines by separating the addresses with a comma. The last line displayed becomes the current line.
[*Address1*,*Address2*]**print** (**p**)	Displays the addressed line or lines. If you do not specify the *Address* parameters, the **p** subcommand displays the current line. If you specify an address, the **p** subcommand displays the addressed line or lines. You can specify a range of lines by separating the addresses with a comma. The last line displayed becomes the current line.
[*Address*]**=**	Displays the line number of the addressed line. If you do not specify an *Address* parameter, the **=** subcommand displays the line number of the current line.
[*Address*]**z**	Displays a screen of text beginning with the addressed line. If an *Address* parameter is not specified, the **z** subcommand displays a screen of text beginning with the current line.
[*Address*]**z-**	Displays a screen of text with the addressed line at the bottom. If an *Address* parameter is not specified, the **z-** subcommand displays a screen of text with the current line at the bottom.
[*Address*]**z.**	Displays a screen of text with the addressed line in the middle. If an *Address* parameter is not specified, the **z.** subcommand displays a screen of text with the current line in the middle.

Editing Additional Files

In the following subcommand, you can use the full subcommand or its abbreviation, which is shown in parentheses.

Item	Description
edit *File* (**e**)	Begins an editing session on a new file specified by the *File* parameter. The editor first checks to see if the buffer was edited since the last **write (w)** subcommand. If the file was edited since the last **w** subcommand, the edit editor issues a warning and cancels the **e** subcommand. Otherwise, the edit editor deletes the contents of the editor buffer, makes the named file the current file, and displays the new file name. After insuring that this file can be edited, the edit editor reads the file into its buffer. If the edit editor reads the file without error, it displays the number of lines and characters that it read. The last line read becomes the new current line.
next (**n**)	Copies the next file named in the command line argument list to the buffer for editing.

Ending and Exiting the edit Editor

In the following subcommands, you can use the full subcommand or its abbreviation, which is shown in parentheses.

Item	Description
quit (**q**)	Ends the editing session after using the **write (w)** subcommand. If you have modified the buffer and have not written the changes, the edit editor displays a warning message and does not end the editing session.
quit! (**q!**)	Ends the editing session, discarding any changes made to the buffer since the last **w** subcommand.

Making Global Changes

In the following subcommand, the *Address* parameters are optional. If you specify an address, do not type the brackets. You can use the full subcommand or its abbreviation, which is shown in parentheses.

Item	Description
[*Address1*,*Address2*]**global/** *Pattern***/***SubcommandList* (**g**)	Marks each of the addressed lines that match the *Pattern* parameter. The edit editor then performs the list of subcommands specified in the *SubcommandList* parameter on each marked line. If you do not specify the *Address* parameters, the **g** subcommand works on the current line. You may need to find the current line or specify an address if you are not in the correct position in the buffer. If you specify an address, the **g** subcommand works on the addressed line or lines. You can specify a range of lines by separating the addresses with a comma. A single subcommand or the first subcommand in a subcommand list appears on same line as the **g** subcommand. The remaining subcommands must appear on separate lines, where each line (except the last) ends with a **** (backslash). The default subcommand is the **print (p)** subcommand. The subcommand list can include the **append (a)** subcommand, **insert (i)** subcommand, and **change (c)** subcommand, and their associated input. In this case, if the ending period is on the last line of the command list, you can omit it. **Note:** The **undo (u)** subcommand and the **g** subcommand cannot appear in the subcommand list.

Moving or Copying Text

In the following subcommands, the *Address1* and *Address2* parameters are optional. If you specify an address, do not type the brackets. You must specify the *Address3* parameter. You can use either the full subcommand or its abbreviation, which is shown in parentheses.

Item	Description
[*Address1,Address2*]**move** *Address3* (**m**)	Moves the current line after the line specified by the *Address3* parameter if you do not specify an address or an address range. You may need to find the current line or specify an address if you are not in the correct position in the buffer. If you specify an address, the **m** subcommand moves the addressed line or lines. You can specify a range of addresses by separating the addresses with a comma. The first of the moved lines becomes the current line.
[*Address1*,*Address2*]**yank** [*Buffer*] (**ya**)	Copies the specified line or lines into the *Buffer*, an optional parameter specified by a single alpha character a to z. You can use the **pu** subcommand to put these lines into another file.

Item	Description
[*Address*]**put** [*Buffer*] (**pu**)	Retrieves the contents of the specified *Buffer* parameter and places it after the current line if you do not specify an address. You may need to find the current line or specify an address if you are not in the correct position in the buffer. If you specify an address, the **pu** subcommand retrieves the contents of the specified buffer and places it after the addressed line. If you do not specify a *Buffer* parameter, the **pu** subcommand restores the last deleted or copied text. You can use the **pu** subcommand with the **delete (d)** subcommand to move lines within a file or with the **yank (ya)** subcommand to duplicate lines between files. You cannot use the **pu** and **ya** subcommands inside a macro.

Saving a File after a System Malfunction

Item	Description
preserve	Saves the current editor buffer as though the system had just malfunctioned. Use this subcommand when a **write (w)** subcommand has resulted in an error and you do not know how to save your work. Use the **recover** subcommand to recover the file.
recover *File*	Recovers the file specified by the *File* parameter from the system save area. Use this subcommand after a system crash or after a **preserve** subcommand.

Saving Text

In the following subcommand, the *Address* parameters are optional. If you specify an address, do not type the brackets. You can use the full subcommand or its abbreviation, which is shown in parentheses.

Item	Description
[*Address1*,*Address2*]**write** [*File*] (**w**)	Writes the entire contents of the buffer to the file specified by the *File* parameter if you do not specify an address. If you specify an address, the **w** subcommand writes the addressed line or lines to the file specified. You can specify a range of lines by separating the addresses with a comma. The edit editor displays the number of lines and characters that it writes. If you do not specify a file, the edit editor uses the current file name. If a *File* parameter does not exist, the editor creates one.

Substituting Text

In the following subcommand, the *Address* parameters are optional. If you specify an address, do not type the brackets. You can use either the full subcommand or its abbreviation, which is shown in parentheses.

Item	Description
[*Address1*,*Address2*] **substitute/***Pattern***/** *Replacement***/** (**s**)	Replaces the first instance of the specified *Pattern* parameter on each addressed line. You can replace every instance of the *Pattern* parameter by adding the **global (g)** subcommand to the end of the **s** subcommand.
[*Address1*,*Address2*] **substitute/***Pattern***/** *Replacement*/**g**	If you do not specify an address, the **s** subcommand works on the current line. You may need to find the current line or specify an address if you are not in the correct position in the buffer. If you specify an address, the **s** subcommand works on the addressed line or lines. You can specify a range of lines by separating the addresses with a comma.

Undoing a Change

In the following subcommand, you can use the full subcommand or its abbreviation, which is shown in parentheses.

Item	Description
undo (**u**)	Reverses the changes made in the buffer by the last buffer editing subcommand. You cannot undo a **write (w)** subcommand or an **edit (e)** subcommand. **Note:** The **global** subcommands are considered a single subcommand to a **u** subcommand.

edquota Command

Purpose

Edits user and group quotas.

Syntax

To Edit User Quotas

edquota [**-u**] [**-p** *Proto-UserName*] *UserName* ...

To Edit Group Quotas

edquota [**-g** [**-p** *Proto-GroupName*] *GroupName* ...]

To Edit Change User or Group Grace Period

edquota -t [**-u** | **-g**]

Description

The **edquota** command creates and edits quotas for JFS file systems.

The **edquota** command creates a temporary file that contains current disk quotas of each user and group. It determines the list of file systems with established quotas from the **/etc/filesystems** file. The **edquota** command also invokes the vi editor (or the editor specified by the **EDITOR** environment variable) on the temporary file so that quotas can be added and modified.

Note: If you specify an editor in the **EDITOR** environment variable, you must specify the full path name of the editor.

Quotas are maintained separately for each file system. When you create or edit a quota for a user or a group, the quota applies to a specific file system. A quota must be set in each file system where you want to use quotas.

By default, or when used with the **-u** flag, the **edquota** command edits the quotas of one or more users specified by the *UserName* parameter on the command line. When used with the **-g** flag, the **edquota**

command edits the quotas of one or more groups specified by the *GroupName* parameter. The **-p** flag identifies a prototypical user (*UserName*) or a prototypical group *(Proto-GroupName*) and duplicates these quotas for a specified user or group.

A user can exceed established soft limits for a default grace period of 1 week. Upon expiration of the grace period, the soft limit is enforced as a hard limit. The grace period can be specified in days, hours, minutes, or seconds. A value of 0 indicates that the default grace period is imposed; a value of 1 second indicates that no grace period is granted. The **-t** flag changes the grace period.

Fields displayed in the temporary file are:

Item	Description
Blocks in use	The current number of 1KB file system blocks used by this user or group.
Inodes in use	The current number of files used by this user or group.
Block soft limit	The number of 1KB blocks the user or group will be allowed to use during normal operations.
Block hard limit	The total amount of 1KB blocks the user or group will be allowed to use, including temporary storage during a quota grace period.
Inode soft limit	The number of files the user or group will be allowed to create during normal operations.
Inode hard limit	The total number of files the user or group will be allowed to create, including temporary files created during a quota grace period.

Note: A hard limit with a value of 1 indicates that no allocations are permitted. A soft limit with a value of 1, in conjunction with a hard limit with a value of 0, indicates that allocations are permitted only on a temporary basis.

When the editor is exited, the **edquota** command reads the temporary file and modifies the binary quota files to reflect any changes.

Hard or soft limits can only be specified in whole 1 KB block amounts.

Flags

Item	Description
-g	Edits the quotas of one or more specified groups.
-p	When invoked with the **-u** flag, duplicates the quotas established for a prototypical user for each specified user. When invoked with the **-g** flag, the **-p** flag duplicates the quotas established for a prototypical group for each listed group.
-t	Changes the grace period during which quotas can be exceeded before a soft limit is imposed as a hard limit. The default value of the grace period is 1 week. When invoked with the **-u** flag, the grace period is set for all file systems with user quotas specified in the **/etc/filesystems** file. When invoked with the **-g** flag, the grace period is set for all file systems with group quotas specified in the **/etc/filesystems** file. **Note:** After changing a grace period using the **edquota** command, the new grace period value will not go into effect until the **quota.user** and **quota.group** files are refreshed by running the **quotaoff** command followed by the **quotaon** command. Users who have already reached their old grace period must reduce their file system usage to a level below their soft limits in order to use the new grace period. In the future, when these users exceed their soft limits, the new grace period will be in effect.
-u	Edits the quotas of one or more users.

Note: If the user or group names contains all numbers then it will be treated as a user or group ID. Quotas will then be edited for the ID rather than the name.

Security

Item	Description
Access Control:	Only the root user can execute this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To create quotas for user `sharl`, using the quotas established for user `davec` as a prototype, enter:

```
edquota -u -p davec sharl
```

Files

Item	Description
quota.user	Specifies user quotas.
quota.group	Specifies group quotas.
/etc/filesystems	Contains file system names and locations.

efsenable Command

Purpose

Activates Encrypted File System (EFS) capability on a system.

Syntax

efsenable **-a** [**-v**] [**-k** <*algo*>] [**-f** <*cipher*>] [**-m** <*mode*>] [**-u** <yes|no>] [**-e** <*algo*>] [**-d** *Basedn*]

efsenable **-q**

Description

The **efsenable** command activates the EFS capability on a system. It creates the EFS administration keystore, the user keystore and the security group keystore. Keystore is a key repository that contains EFS security information. The access key to the EFS administration keystore is stored in the newly created active user's keystore and in the security group keystore. The **efsenable** command creates the **/var/efs** directory. The **/etc/security/user** and **/etc/security/group** files are updated with new EFS attributes. The **efsenable** command also updates the **Config_Rules** ODM database.

Note: The Crypto Library (CLiC) package **clic.rte** must be installed on the system for this command to succeed. This EFS command also requires that Role Based Access Control (RBAC) is enabled on the system, which is the default setting.

Note: The Crypto Library (CLiC) fileset **clic.rte.lib** needs to be minimally at 4.6 for AIX releases of **efsenable** 6.1 TL3 and later.

Flags

Item	Description
-a	Activates the EFS capability on a system.
-d *Basedn*	Sets up the base distinguished names (DN) ou=UsrKeystore, ou=GrpKeystore, ou=EfsCookies and ou=AdmKeystore on the LDAP server to facilitate for the keystore entries to be created along with the local directory structure for the keystore. The *Basedn* passed as argument along with this flag will be used as the *Basedn* for the keystore base distinguished names.
-v	Verbose mode.
-k *algo*	Default algorithm for keys. The *algo* flag can be one of the following values: • RSA_1024 (by default) • RSA_2048 • RSA_4096
-f *cipher*	Default cipher for files. The *cipher* flag can be one of the following values: • AES_128_CBC (by default) • AES_192_CBC • AES_256_CBC • AES_128_ECB • AES_192_ECB • AES_256_ECB
-m *mode*	Default mode for keystores. The *mode* flag can be one of the following values: • admin (by default) • guard
-u [yes|no]	Specifies if the user can change the mode. Default value is "yes".
-e *algo*	Algorithm for the EFS administration key. The possible *algo* values are the same as those of the **-k** flag.
-q	Displays the list of available algorithms.

Exit status

Item	Description
0	The command executed successfully.
1	An error occurred during the execution of the command.
2	A syntax error occurred on the command line.

Security

Item	Description
Access Control:	Only the root user or a user with the **aix.security.efs** authorization and being a member of the security group can run this command.

Examples

1. To display the available algorithms, enter:

```
efsenable -q
```

2. To activate an EFS with default parameters, enter:

```
efsenable –a
```

3. To activate an EFS with a non-default algorithm for keys, and cipher for files, enter:

```
efsenable –a –k RSA_4096 –f AES_256_CBC –e RSA_4096
```

4. To activate an EFS with base DN created on LDAP server along with the local directory structure, type the following command:

```
efsenable –a –d cn=aixdata
```

Files

Item	Description
/etc/security/user	Contains the updates of EFS attributes.
/etc/security/group	Contains the updates of EFS attributes.
/var/efs/users/	Contains the directory for user keystores.
/var/efs/groups/	Contains the directory for group keystores.
/var/efs/efs_admin/	Contains the directory for EFS administration keystore.
/var/efs/efsenabled	Instructs that the EFS is enabled on the system.

efskeymgr Command

Purpose

Manages user and group repositories for the Encrypted File System (EFS) keys (or keystores).

Syntax

efskeymgr -?

efskeymgr -q

efskeymgr -V

efskeymgr [-L *load_module*]**-C** <*group*>

efskeymgr -P < *Open-SSH Public Key file* >

Note: The public key file is located in the **~/.ssh/ directory** directory.

efskeymgr [-L *load_module*] [**-d**] [**-k** <*ks*>] [**-g**] [**-p** <*pw*>] **-v**

efskeymgr [-L *load_module*] [**-d**] [**-k** <*ks*>] **-m**

efskeymgr [-L *load_module*] [**-d**] [**-k** <*ks*>] [**-g**] [**-p** <*pw*>] **-o** <*cmd*>

efskeymgr [-L *load_module*] [**-d**] [**-c** <*cmd*>]

efskeymgr [-L *load_module*] [**-d**] [**-k** <*ks*>] [**-g**] [**-p** <*pw*>] **-n**

efskeymgr [-L *load_module*] [**-d**] [**-k** <*ks*>] [**-g**] [**-p** <*pw*>] **-r** <*mode*>

efskeymgr [-L *load_module*] [**-d**] [**-k** <*ks*>] [**-g**] [**-p** <*pw*>] **-s** <*ks2*>

efskeymgr [-L *load_module*] [**-d**] [**-k** <*ks*>] [**-g**] [**-p** <*pw*>] **-S** <*ks2*>

efskeymgr[-L *load_module*] [**-d**] [**-k** <*ks*>] [**-g**] [**-p** <*pw*>] **-R** <*algo*>

efskeymgr [-L *load_module*] [**-d**] [**-k** <*ks*>] [**-g**] [**-p** <*pw*>] **-D** <*fp*>

efskeymgr [-L *load_module*] [**-d**] [**-k** <*ks*>] [**-g**] [**-p** <*pw*>] **-e** <*file*>

Description

The **efskeymgr** command is dedicated to all key management operations needed by an EFS. Once an EFS is enabled on the system with the **efsenable** command, the keystores (public and private key repositories) are created in the **/var/efs** directory.

The initial password of a user keystore is the user login password. Group keystores and admin keystores are not protected by a password but by an access key. Access keys are stored inside all user keystores that belong to this group.

When you open a keystore (at login or explicitly with the **efskeymgr** command), the private keys contained in this keystore are pushed to the kernel and associated with the process. If access keys are found in the keystore, the corresponding keystores are also opened and the keys are automatically pushed into their kernel.

Keystores support two administration modes: admin mode and guard mode.

admin mode
: When a keystore is set to this mode, an EFS administrator with the **aix.security.efs** RBAC authorization and the access key to admin keystore can open the keystore for management including password reset, key regeneration, access key addition or removal, and so on.

guard mode
: When a keystore is set to this mode, the EFS administrator cannot get access to the keystore. In this mode, if the password to keystore is lost, there is no possible recovery of the private key.

When the keystore password is the same as the login password, the keystore is automatically opened at the login time and the keys are available in the session. The keystore password is kept in sync with the login password when the **passwd** command is used and the old password is provided. If at some point the keystore password is not in sync with the login password, you can change the keystore password using the **efskeymgr** command. When the passwords are not synchronized, the keys are no longer automatically associated with the session when you log in.

The following command grants or removes the EFS credentials only for the execution of the *cmd* command. When the *cmd* command returns, the previous process credentials are restored.

```
efskeymgr –o <cmd> and efskeymgr –c <cmd>
```

When a private key is regenerated in a keystore, a new private key is created and the old key is marked "deprecated".

Note: The new key is not pushed into the kernel. You must open your keystore again, either with the **efskeymgr** command or by closing and opening your session, for the new key to be available for file operations.

The deprecated key can still be used to decrypt files, but is no longer used to encrypt files. The deprecated key can be removed from the keystore, but in this case all files that were encrypted with the old key will no longer be accessible.

Note: This EFS command requires that Role Based Access Control (RBAC) is enabled on the system, which is the default setting.

Delayed operations

In some cases, the keystore cannot be modified directly by a command or an action. When this occurs, a special file is created in the keystore directory, and will be parsed next time the keystore is opened. This special file is called a cookie. For keystores in admin mode, the cookies are parsed automatically when the keystore is opened (at login or when the **efskeymgr** command is run). For keystores in guard mode, the cookies are never automatically parsed. The user must give its approval for each modification of its

keystore. When you open a session, a message is displayed if one or more operations are pending on your EFS keystore:

- Your private key must be regenerated.
- You are granted access to **group**/*group1* keystore.

You must run the **efskeymgr -v** command to process pending operations.

The following actions are possible:

- Private key regeneration. This results in a new private key being generated, and the old one being marked "deprecated".
- New access key. When you accept this cookie, you obtain access to a new keystore (for example, keystore of a group to which you are added).
- Remove access key. When you accept this cookie (for example, when the access key is removed from a group), you loose your access to a keystore.

Note: When you run the **efskeymgr** command with any flag that opens your keystore, for example, the **-v** flag, you are prompted what you want to do with each cookie. The choices are as follows:

- Accept the cookie: your keystore is modified according to the cookie, then the cookie is destroyed.
- Postpone the cookie: your keystore is not modified and the cookie is not removed. You will be prompted next time for action.
- Delete the cookie: your keystore is not modified and the cookie is removed. You must use the **efskeymgr** command to do the action again.

Flags

Item	Description
General flags:	
-d	Verbose mode.
-g	Does not process pending operations when opening the keystore.
-k *ks*	The operation is targeted to the *ks* keystore instead of the active user's keystore. The *ks* value can be as follows: **user/<login>** User <login> keystore. **group/<grpname>** Group <grpname> keystore. **admin/** EFS administration keystore.
-L *load_module*	Specifies the loadable module to use for keystore operations.
-p *pw*	Password to use to open the keystore. It is not advised to use this flag as it can be seen by other users using the **ps** command, for example.
-P *filename*	Push the public key cookies for all the keys present in the OpenSSH file located in the **~/.ssh/authorized_keys** directory.

Item	Description
Flags for commands (no access to the keystore files):	
-?	Displays the command help and exits.
-q	Displays a list of supported algorithms for the key regeneration.
-V	Displays the keys associated with the active process credentials in the kernel.
Flags for commands (read-only access to keystores):	
-c <*cmd*>	Removes all keys from the kernel, then runs the *cmd* command. The keys are restored when the *cmd* command terminates.
-m	Lists all pending operations on the keystore.
-o <*cmd*>	Opens the keystore and pushes the keys, then runs the *cmd* command. The keys are discarded when the *cmd* command terminates.
-v	Displays the content of the keystore file.
Flags for commands (read/write access to keystores):	
-C <*group*>	Creates the keystore of the *group* group.
-D <*fp*>	Removes a deprecated private key from the keystore. The *fp* value is the key fingerprint.
-e <*file*>	Exports a keystore to a file. The file is PKCS#12 encoded and contains the public and private keys from the keystore. This file can be used in openssh, for example.
-n	For user keystores, prompts for a new password for the keystore. For group keystores, generates a new access key and sends to group members. For admin keystores, generates a new access key. The key must then be sent to the EFS administrators with the **efskeymgr** command.
-R <*algo*>	Regenerates the keystore private key. See the **-q** flag for the valid values for the *algo* parameter.
-r <*mode*>	Changes the keystore administration mode. The *mode* value can be as follows: **admin** The EFS administrator can administer the keystore. Pending operations are applied automatically. **guard** The EFS administrator cannot manage the keystore. The user is prompted for any pending operation.
-S <*ks2*>	Removes the *ks2* access key from the keystore. On subsequent opening of keystore, the *ks2* private key is no longer pushed automatically.
-s <*ks2*>	Sends the keystore access key to the *ks2* keystore. On subsequent opening of the *ks2* key, the keystore private key is loaded automatically.

Exit status

Item	Description
0	The command ran successfully.
1	An error occurred during the execution of the command.
2	A syntax error occurred on the command line.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To view your keystore content, enter:

```
efskeymgr –v
```

2. To view the keys associated with the active shell, enter:

```
efskeymgr -V
```

3. To regenerate the private key from your keystore, enter:

```
efskeymgr –R RSA_1024
```

4. To delete a deprecated key, enter:

```
efskeymgr –D dbb62547:d6925088:45357fd3:54cddbba:27b255a9
```

5. To send the access key of the group "students" to the user "joe", enter:

```
efskeymgr –k group/students –s user/joe
```

6. To push the Open-SSH Client users Open-SSH Public key cookies in the target keystore, where the **~/.ssh/authorized_keys** file contains the installed public keys, enter:

```
efskeymgr  -P  ~/.ssh/authorized_keys
```

7. To create Group keystore directly on LDAP, if configured:

```
efskeymgr -L LDAP -C staff
```

Files

Item	Description
/var/efs	Contains all keystores.
/etc/security/user	Contains the EFS attributes for the creation and management of users keystore.
/etc/security/group	Contains the EFS attributes for the creation of groups keystore.

efskstoldif Command

Purpose

Prints certain EFS users or groups keystore that are defined locally to **stdout** in ldif format.

Syntax

efskstoldif -d *baseDN* [**-u** | **-g**] {ALL | *Name* [*Name*] ...}

Description

The **efskstoldif** command reads data from locally defined EFS users or groups keystore files and prints the result to **stdout** in ldif format. If redirected to a file, the result can be added to a LDAP server with the **ldapadd** command with the **-b** flag or the **ldif2db** command.

The **efskstoldif** command reads the **/etc/security/ldap/sectoldif.cfg** file to determine what to name the user, group and cookie sub-trees that the data will be exported to. The **efskstoldif** command only exports data to the USERKEYSTORE, GROUPKEYSTORE, EFSCOOKIES and ADMINKEYSTORE types defined in the file. The names specified in the file will be used to create sub-trees under the base distinguished name (DN) specified with the **–d** flag. For more information, see the /etc/security/ldap/sectoldif.cfg file in AIX Version 6.1 TL 4 for reference.

The LDIF output generation does not look the **efs_keystore_access** nor the **efs_adminks_access** attribute of the users/groups. Whatever will be its value either “file” or “ldap” the LDIF format will be generated. For whatever users or groups keystore the ldif format is generated, if any cookies exist for those keystore then even for them the ldif generation takes place.

Note: If there are any cookies present on files, even the LDIF generation happens for them too. System Administrator has to take care of the consistency of the keystore entries on LDAP and files if required.

Flags

Item	Description
-d *baseDN*	Specifies the base distinguished names (DN) under which to place the EFS Keystore data.
-g **ALL***Names* . ..	Directs the command to generate the output for the groups specified in the succeeding arguments. **ALL** Specifies that all the groups must be considered. **Name** Specifies the single group name or list of group names separated by blanks.
-u **ALL***Names* . ..	Directs the command to generate the output for the users specified in the succeeding arguments. **ALL** Specifies that all the users must be considered. **Name** Specifies the single user name or list of user names separated by blanks.

Exit status

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: This command should grant execute (x) access only to the root user.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/etc/security/user	Contains the EFS attributes for the creation and management of users keystore.
/etc/security/group	Contains the EFS attributes for the creation and management of users keystore.
/var/efs	Contains all keystores.

Examples

1. To export all the users and groups keystore content to ldif format with the base DN of cn=aixdata, type the following command:

   ```
   efskstoldif -d cn=aixdata
   ```

2. To export all the users keystore content to ldif format with the base DN of cn=aixdata, type the following command:

   ```
   efskstoldif -d cn=aixdata -u ALL
   ```

3. To export all the groups keystore content to ldif format with the base DN of cn=aixdata, type the following command:

   ```
   efskstoldif -d cn=aixdata -g ALL
   ```

4. To export only selected users keystore content to ldif format with the base DN of cn=aixdata, type the following command:

   ```
   efskstoldif -d cn=aixdata -u davis smith
   ```

5. To export only selected groups keystore content to ldif format with the base DN of cn=aixdata, type the following command:

   ```
   efskstoldif -d cn=aixdata -g finance managers
   ```

efsmgr Command

Purpose

Manages the files encryption and decryption for the Encrypted File System (EFS).

Syntax

efsmgr -?

efsmgr -q [**-v**]

efsmgr -C <*cipher*> [**-v**]

efsmgr [**-c** <*file*>] **-e** <*file*> [**-v**]

efsmgr [**-c** <*cipher*>] [**-s**] **-E** <*dir*> [**-v**]

efsmgr [**-c** <*cipher*>] **-t** <*file*> [**-v**]

efsmgr [**-c** <*cipher*>] [**-s**] **-T** <*dir*> [**-v**]

efsmgr -d <*file*> [**-v**]

efsmgr [**-s**] **-D** <*dir*> [**-v**]

efsmgr -l <*file*> [**-v**]

efsmgr [**-s**] **-L** <dir> [**-v**]

efsmgr -a <*file*> [**-u** <*user*> | **-g** <*group*>] [**-v**]

efsmgr -r <*file*> [**-u** <*user*> | **-g** <*group*>] [**-v**]

Description

The **efsmgr** command is dedicated to the files encryption management inside EFS. Encrypted files can only be created on the EFS-enabled JFS2 file systems. For more information about enabling EFS on your system, see the **mkfs**, **chfs**, **crfs**, and **efsenable** commands.

There are two ways to create encrypted files: either explicitly by using the following command, or implicitly when inheritance is set on the file system or the directory where the file is being created.

```
efsmgr -e <file>
```

When inheritance is set on a directory, all new files created in this directory are encrypted by default. The cipher used to encrypt files is the inherited cipher. New directories also inherit the same cipher. If inheritance is disabled on a subdirectory, the new files created in this subdirectory will not be encrypted.

When inheritance is set on a file system, all new files created in this file system are encrypted using the inherited cipher. If inheritance is set both on a directory and a file system with different ciphers, new files created in this directory will be encrypted using the cipher inherited from the directory.

Setting or removing inheritance on a directory or a file system has no effect on the existing files. The **efsmgr** command must be used explicitly to encrypt or decrypt files.

The file owner's private key must be loaded into the process before the encrypted file can be created. The access to the encrypted file can be granted to any user or group with a keystore, which is a key repository that contains EFS security information. For more information about managing user and group repositories, see the **efskeymgr** command.

When an encrypted file is being opened, the Discretionary Access Control (DAC) and the Access Control List (ACL) are checked for the file access permission. If the access is granted, the keys loaded into the kernel for the process are searched for a private key matching one of the file's protection keys. If a matching key is found, the file content can be read, otherwise the access is denied.

Note: This EFS command requires that Role Based Access Control (RBAC) is enabled on the system, which is the default setting.

Flags

Item	Description
-c <*cipher*>	Uses this cipher instead of the inherited or the default cipher. See the **-q** command for the valid *cipher* values.
-g <*group*>	This group must be added or removed from the EFS access list. The *group* value can be either the gid or the group name.

Item	Description
-s	The operation is targeted to a file system rather than a directory. In this case, the *dir* parameter must be the mount point of a file system with EFS support.
-u <*user*>	This user must be added or removed from the EFS access list. The *user* value can be either the uid or the login name.
-v	Verbose mode.
-?	Displays the command help and exits.
-a <*file*>	Adds access to the specified file to a list of users and groups specified with the **-u** and **-g** flags.
-C <*cipher*>	Changes the default cipher for your user to the *cipher* value.
-D <*dir*>	Removes the inheritance on the directory. To apply the command on the whole file system, you must add the **-s** flag.
-d <file>	Decrypts the specified file.
-E <*dir*>	Sets the inheritance on the *dir* directory. To apply the command on the whole file system, you must add the **-s** flag.
-e <file>	Encrypts the specified file.
-L <*dir*>	Displays the inherited cipher on the specified directory.
-l <*file*>	Lists the encryption information of the specified file: cipher, and keys that can decrypt the file.
-q	Displays a list of supported ciphers.
-r <*file*>	Revokes access to the specified file to a list of users and groups specified with the **-u** and **-g** flags.
-T <*dir*>	Changes the inherited cipher on the specified directory. To apply the command on the complete file system, you must add the **-s** flag.
-t <*file*>	Refreshes the encryption keys of the specified file. This can also be used to change the file cipher.

Exit status

Item	Description
0	The command executed successfully.
1	An error occurred during the execution of the command.
2	A syntax error occurred on the command line.

Examples

1. To encrypt the **database.txt** file using a strong cipher, enter:

```
efsmgr –e database.txt –c AES_256_CBC
```

2. To display the list of keys that can open the file, enter:

```
efsmgr –l database.txt
```

3. To add access to user joe and to group maintainers to the file, enter:

```
efsmgr –a database.txt –u joe –g maintainers
```

4. To set the inheritance on the file system of the home directory, enter:

```
efsmgr -c AES_128_CBC -s -E /home
```

Files

Item	Description
/etc/security/user	Contains the default cipher attributes for the user.

egrep Command

Purpose

Searches a file for a pattern.

Syntax

egrep [**-h**] [**-i**] [**-p**[*Separator*]] [**-s**] [**-u**] [**-v**] [**-w**] [**-x**] [**-y**] [[**-b**] [**-n**] | [**-c** | **-l** | **-q**]] { { **-e***Pattern* | **-f***StringFile* } ... | *Pattern* } [*File ...*]

Description

The **egrep** command searches an input file (standard input by default) for lines matching a pattern specified by the *Pattern* parameter. These patterns are full regular expressions as in the **ed** command (except for the \ (backslash) and \\ (double backslash)). The following rules also apply to the **egrep** command:

- A regular expression followed by a + (plus sign) matches one or more occurrences of the regular expression.
- A regular expression followed by a ? (question mark) matches zero or one occurrence of the regular expression.
- Multiple regular expressions separated by a | (vertical bar) or by a new-line character match strings that are matched by any of the regular expressions.
- A regular expression may be enclosed in ( ) (parentheses) for grouping.

The new-line character will not be matched by the regular expressions.

The order of precedence for operators is [,], *, ?, +, concatenation, | and the new-line character.

> **Note:** The **egrep** command is the same as the **grep** command with the **-E** flag, except that error and usage messages are different and the **-s** flag functions differently.

The **egrep** command displays the file containing the matched line if you specify more than one *File* parameter. Characters with special meaning to the shell ($, *, [, |, ^, (,), \ ) must be in quotation marks when they appear in the *Pattern* parameter. When the *Pattern* parameter is not a simple string, you usually must enclose the entire pattern in single quotation marks. In an expression such as `[a-z]`, the minus means through according to the current collating sequence. A collating sequence may define equivalence classes for use in character ranges. It uses a fast, deterministic algorithm that sometimes needs exponential space.

Notes:

1. Lines are limited to 2048 bytes.
2. Paragraphs (under the **-p** flag) are currently limited to a length of 5000 characters.
3. Do not run the **grep** command on a special file because it produces unpredictable results.
4. Input lines should not contain the NULL character.
5. Input files should end with the newline character.

6. Although some flags can be specified simultaneously, some flags override others. For example, if you specify **-l** and **-n** together, only file names are written to standard output.

Flags

Item	Description
-b	Precedes each line by the block number on which it was found. Use this flag to help find disk block numbers by context. The **-b** flag cannot be used with input from stdin or pipes.
-c	Displays only a count of matching lines.
-e *Pattern*	Specifies a *Pattern*. This works like a simple *Pattern* but is useful when the *Pattern* begins with a - (minus sign).
-f *StringFile*	Specifies a file that contains strings.
-h	Suppresses file names when multiple files are being processed.
-i	Ignores the case of letters when making comparisons.
-l	Lists just the names of files (once) with matching lines. Each file name is separated by a new-line character. If standard input is searched, a path name of "`(StandardInput)`" is returned.
-n	Precedes each line with its relative line number in the file.
-p[*Separator*]	Displays the entire paragraph containing matched lines. Paragraphs are delimited by paragraph separators, as specified by the *Separator* parameter, which are patterns in the same form as the search pattern. Lines containing the paragraph separators are used only as separators; they are never included in the output. The default paragraph separator is a blank line.
-q	Suppresses all output to standard output, regardless of matching lines. Exits with a 0 status if an input line is selected.
-s	Displays only error messages. This is useful for checking status.
-u	Causes output to be unbuffered.
-v	Displays all lines except those that match the specified pattern.
-w	Does a word search.
-x	Displays lines that match the specified pattern exactly with no additional characters.
-y	Ignores the case of letters when making comparisons.

Exit Status

This command returns the following exit values:

Item	Description
0	A match was found.
1	No match was found.
>1	A syntax error was found or a file was inaccessible (even if matches were found).

Examples

To use an extended pattern that contains some of the pattern-matching characters +, ?, |, (, and), enter:

```
egrep "\(([A-z]+|[0-9]+)\)" my.txt
```

This displays lines that contain letters in parentheses or digits in parentheses, but not parenthesized letter-digit combinations. It matches `(y)` and `(783902)`, but not `(alpha19c)`.

> **Note:** When using the **egrep** command, \ ((backslash followed by open parenthesis) or \ ((backslash followed by close parenthesis) match parentheses in the text, but ((open parenthesis) and) (closed parenthesis) are special characters that group parts of the pattern. The reverse is true when using the **grep** command.

Files

Item	Description
/usr/bin/egrep	Contains the hard link to the **egrep** command.
/bin/egrep	Specifies the symbolic link to the **egrep** command.

eimadmin Command

Purpose

Manages Enterprise Identity Mapping (EIM) domains.

Syntax

`eimadmin -a | -p | -l | -m | -e -D | -R | -I | -A | -C [-s` *switch*`] [-v` *verboseLevel*`] [-c` *accessType*`] [-f` *accessUserType*`] [-g` *registryParent*`] [-i` *identifier*`] [-j` *otherIdentifier*`] [-k` *URI*`] [-n` *description*`] [-o` *information*`] [-q` *accessUser*`] [-r` *registryName*`] [-t` *associationType*`] [-u` *registryUser*`] [-x` *registryAlias*`] [-y` *registryType*`] [-z` *registryAliasType*`] [-d` *domainDN*`] [-h` *ldapHost*`] [-b` *bindDN*`] [-w` *bindPassword*`] [-K` *keyFile* `[ -P` *keyFilePassword*`] [-N` *certificateLabel*`]] [-S` *connectType*`]`

Description

The `eimadmin` command is an AIX System Services Shell tool. An administrator can use it to define an EIM domain and prime the domain with registries, identifiers, and associations between identifiers and registry users. An administrator can also use `eimadmin` to give users (and other administrators) access to an EIM domain, or list or remove the EIM entities.

Administrators can use the `eimadmin` command in two ways:

- By including information with command-line options on an `eimadmin` command
- By including information in an input file that an `eimadmin` command references

You can create the file manually or by exporting records from a database. The administrator directs utility processing by specifying a combination of command-line options.

The `eimadmin` command can perform the following actions:

- Add an object (`-a`)
- Purge an object (`-d`)
- List objects (`-l`)
- Modify attributes associated with objects (`-m`)
- Erase attributes (`-e`)

on the following objects:

- Domains (`-D`)

- Registries (`-R`)
- Identifiers (`-I`)
- Associations (`-A`)
- Access authorities (`-C`)

Note:

1. Each `eimadmin` command must include one action and one object type. Depending on the object and the action you are performing on it, EIM might require additional parameters.
2. Some options are for multivalue attributes, which you can specify more than once. Other options are for single-value attributes, which you can specify only once. (If you repeat an option that is for a single-value attribute, `eimadmin` processes only the first value it encounters in the command.) Apart from these stipulations, the order in which you specify parameters is not important.
3. You can code the parameters of the `eimadmin` command in several ways:
   - Concatenate an action and an object, omitting the embedded hyphen: `-aD`
   - Include both hyphens, and separate the two options with a space: `-a -D`

   In other words, the following example is *not* valid because it includes both hyphens and there is no space before `-D`: `-a-D`

Flags

The `eimadmin` command takes the following action flags.

Item	Description
-a	Adds an object. (Creates an object definition and its attributes.)
-e	Erases an attribute. (Clears a single-value attribute or removes a multivalue attribute.)
-l	Lists an object. (Retrieves an object definition and its attributes.)
-m	Modifies an attribute. (Alters an attribute of an existing object, either by changing a single-value attribute or adding a multivalue attribute.)
-p	Purges an object. (Removes an object definition and its attributes.)

The `eimadmin` command takes the following object flags.

Item	Description
-A	An association. This is a relationship between an identifier in the EIM domain and a user ID.
-C	An access authority. This is an EIM-defined LDAP access control group.
-D	A domain. This is a collection of identifiers, user registries, and associations between identifiers and user IDs, stored within an LDAP directory.
-I	An identifier. This is the name of a person or entity participating in an EIM domain.
-R	A registry. This is the name of a user registry. Associations are defined between identifiers and user IDs in the user registry.

The `eimadmin` command takes the following processing control flags.

Item	Description
-s *switch*	The *switch* specifies a value that affects the way the `eimadmin` command functions operate. You can specify the following value: **RMDEPS** Removes dependents when removing a domain or system registry. This makes it easier to remove a domain by first removing all identifiers and registries defined for the domain. It also makes it easier to remove a system registry by first removing all application registries defined for the registry.  **Attention: Attention:** The `eimadmin` command does not warn you that dependents exist before removing them, so use this switch carefully.
-v *verboseLevel*	The *verboseLevel* parameter is an integer from 1 to 10 that controls the amount of trace detail that the `eimadmin` command displays. (It is for diagnosing problems in the `eimadmin` utility.) The default value of 0 indicates no trace information. You can specify an integer value from 1 to 10, from the least to greatest amount of trace information. The utility checks the value and displays trace information defined for the level and all lower levels. The following levels trigger specific information: • 3–indicates EIM API call parameters and return values • 6–indicates option values and input file labels • 9–indicates utility routine entry and exit statements

The `eimadmin` command takes the required and optional attribute flags listed in the following table. The flag options are single-valued unless otherwise indicated. If you specify an option more than once, the utility processes only the first occurrence.

Note:

1. You can specify these attributes as command options or as fields in input files. If you are specifying command options, you must enclose values with imbedded blanks within quotation marks (") or ('). Quotation marks are optional for single-word values. Specifying a multiword value without quotation marks in effect truncates the command line options; values after the first word are truncated.
2. The following special characters are not allowed in *registryName*, *registryParent*, or *identifier*:

```
, = + < > # ; \ *
```

Item	Description
-c *accessType*	Specifies the scope of access authority the user has over the EIM domain. *accessType* must be one of the following values: **ADMIN** Specifies administrative access. **REGISTRY** Specifies registry access. If you specify REGISTRY, you must also specify a registry value (`-r`). The registry value can be a specific registry name or it can be an asterisk (*) to indicate access to all registries. **IDENTIFIER** Specifies identifier access. **MAPPING** Specifies mapping operations access.

Item	Description
-f *accessUserType*	Specifies the type for the access user name. *accessUserType* must be one of the following types: **DN** The *accessUser* is a distinguished name. **KERBEROS** The *accessUser* is a Kerberos identity.
-g *registryParent*	Specifies the name of a system registry. An application registry is a subset of a system registry. If you are adding an application registry, you must use the -r option and the -g option. The -r value is the application registry you are defining. The -g option is the preexisting system registry.
-i *identifier*	Specifies a unique identifier name. For example: John Day.
-j *otherIdentifier*	Specifies a nonunique identifier name. For example: John. **Note:** You can specify this option multiple times to assign multiple nonunique identifiers.
-k *URI*	Specifies the Universal Resource Identifier (URI) for the registry (if one exists).
-n *description*	Specifies any text (that you provide) to associate with the domain, registry, identifier, or association. **Note:** You can define a user description only for target associations.
-o *information*	Specifies additional information to associate with an identifier or association. **Note:** You can define user information only for target associations. You can specify this option multiple times to assign multiple pieces of information.
-q *accessUser*	Specifies the user distinguished name (DN) or the Kerberos identity with EIM access, depending on the *accessUserType* specified.
-r *registryName*	Specifies the name of a registry. When you add a new registry, eimadmin treats the registry as a system registry unless you also specify the -g option. If you specify the -g option, eimadmin treats the registry as an application registry.
-t *associationType*	Specifies the relationship between an identifier and a registry. *associationType* must be one of the following: **ADMIN** Indicates associating a user ID with an identifier for administrative purposes. **SOURCE** Indicates that the user ID is the source (or from) of a lookup operation. **TARGET** Indicates that the user ID is the target (or to) of a lookup operation. **Note:** You can specify this option multiple times to define multiple relationships.
-u *registryUser*	Specifies the user ID of the user defined in the registry.
-x *registryAlias*	Specifies another name for a registry. You must specify this option multiple times to assign multiple aliases.

Item	Description
-y *registryType*	Specifies the type of registry. Predefined types that `eimadmin` recognizes include the following: • RACF® • OS/400® • KERBEROS (for case ignore) • KERBEROSX (for case exact) • AIX • NDS • LDAP • PD (Policy Director) • WIN2K You can also create your own types by concatenating a unique OID with one of the following two normalization methods: • caseIgnore • caseExact
-z *registryAliasType*	Specifies the type for a registry alias. You can invent your own value or use one of the following suggested values: • DNSHostName • KerberosRealm • IssuerDN • RootDN • TCPIPAddress • LdapDnsHostName **Note:** For a set of command line options or single input data record, the `eimadmin` command recognizes only the first specification of *registryAliasType*. However, the `eimadmin` command does recognize multiple registry aliases and associates all of them with the single *registryAliasType*.

The `eimadmin` command takes the following connection type flags.

Item	Description
-b *bindDN*	Specifies the distinguished name to use for the simple bind to LDAP.
-d *domainDN*	Specifies the full distinguished name (DN) of the EIM domain. *domainDN* begins with `'ibm-eimDomainName='` and consists of the following elements: ***domainName*** The name of the EIM domain you are creating. For example, `MyDomain`. **parent distinguished name** The distinguished name for the entry immediately above the given entry in the directory information tree hierarchy, such as `o=ibm,c=us`. For example: `ibm-eimDomainName=MyDomain,o=ibm,c=us`

Item	Description
-h *ldapHost*	Specifies the URL and port for the LDAP server controlling the EIM data. The format is: `ldap://some.ldap.host:389` `ldaps://secure.ldap.host:636`
-K *keyFile*	Specifies the name of the SSL key database file, including the full path name. If the file cannot be found, it is assumed to be the name of a RACF key ring that contains authentication certificates. This value is required for SSL communications with a secure LDAP host (prefixed `ldaps://`). For example: `/u/eimuser/ldap.kdb`
-N *certificateLabel*	Specifies which certificate to use from the key database file or RACF key ring. If this option is not specified, the certificate marked as the default in the file or ring is used.
-P *keyFilePassword*	Specifies the password required to access the encrypted information in the key database file. Alternatively, you can specify an SSL password stash file for this option by prefixing the stash file name with `file://`. For example: `secret  or file:///u/eimuser/ldapclient.sth` **Note:** The `eimadmin` command prompts for a key file password if you specify the name of a key database file for the `-K` option but not the `-P` option on the command line.
-S *connectType*	Specifies the method of authentication to the LDAP server. *connectType* must be one of the following values: • `SIMPLE` (bind DN and password) • `CRAM-MD5` (bind DN and protected password) • `EXTERNAL` (digital certificate) • `GSSAPI` (Kerberos) If not specified, *connectType* defaults to `SIMPLE`. For connect type `GSSAPI`, the default Kerberos credential is used. This credential must be established using a service such as kinit prior to running `eimadmin`. For KINIT and related information, refer to the AIX Authentication Service Administration.
-w *bindPassword*	Specifies the password associated with the bind DN.

The connection information needed by the utility includes the EIM domain (`-d`) and its controlling server (`-h`), the identity (`-b`,`-w`; or `-K`,`-P`,`-N`) with which to authenticate (bind) to the server, and the authentication method (`-S`).

For object types other than domain (`-D`), specifying the domain, server and bind identity is optional. If these are not specified, the information is retrieved from a RACF profile.

Note: If any of the connect information is specified, the full set of values required for the connect type must also be specified. Omitting one or more values (but not all) results in an error. The following table shows the required and optional values for each connect and host type when specified with the `eimadmin` command.

Connection Type/Host Type	Required Values	Optional Values
SIMPLE or CRAM-MD5/secure (`ldaps://`)	-d, -h, -b, -w, -K, -P	-N
SIMPLE or CRAM-MD5/nonsecure (`ldap://`)	-d, -h, -b, -w	

Connection Type/Host Type	Required Values	Optional Values
EXTERNAL/secure (`ldaps://`)	-d, -h, -K, -P, -S	-N
EXTERNAL/nonsecure (`ldap://`)	unsupported	unsupported
GSSAPI/secure (`ldaps://`)	-d, -h, -K, -P, -S	-N
GSSAPI/nonsecure (`ldap://`)	-d, -h, -S	

Note:

1. There are two exceptions to the preceding table:
   - The domain option (`-d`) is not required for domain functions if the value is specified through an input file.
   - An SSL key database file password or stash file (-P) is not required when -K specifies a RACF key ring.
2. The `eimadmin` command prompts for the simple bind password if it is required and `-w` is not specified on the command line, and prompts for the SSL key database file password if it is required and `-P` is not specified on the command line.

The following table summarizes required and optional flags for each object type and action pair. You can specify the value for most options in an input file instead of specifying it on the command line.

Object Type (Action)	Flags	Comments
`D` (a)	• Required: d, h • Optional: n	Add a domain.
`D` (p)	• Required: d, h • Optional: s	Remove a domain. If the domain is not empty, include `-s RMDEPS`.
`D` (l)	• Required: d, h • Optional:	List domains. Specify `-d*` to list all domains.
`D` (m)	• Required: d, h • Optional: n	Modify or add a domain attribute.
`D` (e)	• Required: d, h • Optional: n	Remove or clear a domain attribute.
`R` (a)	• Required: `r`, y • Optional: g, k, n, x, z	Add a registry. The value specified for `-r` is assumed to be a new system registry unless `-g` is also specified, in which case the `-r` value indicates a new application registry.
`R` (p)	• Required: `r` • Optional: s	Remove a registry.
`R` (l)	• Required: `r` • Optional: y	List registries. Return all registry entries in the domain that match the specified `-r` value search filter, which might contain the wild card `*`.
`R` (m)	• Required: `r` • Optional: k, n, x, z	Modify or add a registry attribute, including a registry alias.

Object Type (Action)	Flags	Comments
R (e)	• Required: r • Optional: k, n, x, z	Remove or clear a registry attribute, including a registry alias.
I (a)	• Required: i • Optional: j, n, o	Add an identifier.
I (p)	• Required: i • Optional:	Remove an identifier.
I (l)	• Required: i • Optional:	List an identifier by unique identifier name. Return all identifier entries in the domain that matches the specified -i value search filter, which might contain the wild card *.
I (l)	• Required: j • Optional:	List an identifier by nonunique identifier name. Return all identifier entries in the domain that have a nonunique identifier matching the specified -j value search filter, which might contain the wild card *.
I (m)	• Required: i • Optional: j, n, o	Modify or add an identifier attribute.
I (e)	• Required: i • Optional: j, n, o	Remove or clear an identifier attribute.
A (a)	• Required: i, r, u, t • Optional: n, o	Add an association. You can repeat the -t option to add multiple associations types. The -n and -o flags are relevant only to TARGET associations.
A (p)	• Required: i, r, u, t • Optional:	Remove an association. You can repeat the -t option to remove multiple associations types.
A (l)	• Required: i • Optional: t	List associations. Return all associations in the domain for specified -i unique identifier. Specify a -t value to limit the entries returned to the given association type.
A (m)	• Required: r, u • Optional: n, o	Modify or add an association attribute. The -n and -o flags are relevant only to TARGET associations.
A (e)	• Required: r, u • Optional: n, o	Remove or clear an association attribute. The -n and -o flags are relevant only to TARGET associations.
C (a)	• Required: c, q, f • Optional: r	Add access. For access type REGISTRY, provide a specific -r registry value, or a wild card * indicating access to all registries in the domain.
C (p)	• Required: c, q, f • Optional: r	Remove access. For access type REGISTRY, provide a specific -r registry value, or a wild card * indicating access to all registries in the domain.
C (l)	• Required: c • Optional: r	List access by type. For access type REGISTRY, provide a specific -r registry value, or a wild card * indicating access to all registries in the domain.

Object Type (Action)	Flags	Comments
C (l)	• Required: q, f • Optional:	List access by user.

Exit Status

The eimadmin command returns one of the following exit codes upon completion:

Item	Description
0	Successful.
4	One or more errors encountered but, if you specified an input file, all records were processed.
8	A severe error occurred that caused processing to stop before reaching the end of an input file, if specified.

Examples

1. To list a single domain, type:

   ```
   eimadmin -lD -h ldap://my.server -b "cn=EIM admin,o=MyCompany,c=US" -d "ibm-eimDomainName=My Employees,o=My Company,c=US"
   ```

   This returns something similar to the following output:

   ```
   domain name: My Employees
   domain DN: ibm-eimDomainName=My Employees,o=My Company,c=US
   description: employees in my company
   ```

2. To list a single registry, type:

   ```
   eimadmin -lR -r MyRegistry
   ```

   This returns something similar to the following output:

   ```
   registry: MyRegistry
   registry kind: APPLICATION
   registry parent: MySystemRegistry
   registry type: RACF
   description: my racf registry
   URI: ldap://some.big.host:389/profileType=User,cn=RACFA,o=My Company,c=US
   registry alias: TCPGROUP
   registry alias type: DNSHostName
   ```

3. To list identifiers, type:

   ```
   eimadmin -lI -i "J.C.Smith"
   ```

   This returns something similar to the following output:

   ```
   unique identifier: J.C.Smith
   other identifier: J.C.Smith
   other identifier: Joseph
   other identifier: Joe
   description: 004321
   information: D01
   information: 1990-04-11
   ```

4. To list target associations, type:

   ```
   eimadmin -lA -i "J.C.Smith" -t target
   ```

   This returns something similar to the following output:

```
unique identifier: J.C.Smith
registry: MyRegistry
registry type: RACF
association: target
registry user: SMITH
description: TSO
information: 1989-08-01
information: ADMIN1
```

5. To list accesses, type:

```
eimadmin -lC -c admin
```

This returns something similar to the following output:

```
access user: cn=JoeUser,o=My Company,c=us
access user: cn=admin1,o=My Company,c=us
access user: cn=admin2,o=My Company,c=us
```

Location

/usr/bin/eimadmin

Security

The LDAP administrator has the authority to use the `eimadmin` command and access to all the functions it provides. EIM administrators can use the command as long as the following conditions are true:

- They have a bind distinguished name and password defined at the LDAP server containing the EIM domain
- Their bind distinguished name has one of the EIM authorities:
  - EIM administrator
  - EIM registries administrator
  - EIM registry X administrator
  - EIM identifiers administrator

Standard Error

The `eimadmin` command issues a message to prompt for a password or to indicate an error. Do not expect to receive a message for successful completion unless you use an input file. When processing records in an input file, `eimadmin` issues an informational message as the process starts and stops, in addition to a progress message every 50 records.

Note: The `eimadmin` command returns one or more data lines for list (-l) requests unless it finds no matching EIM entries, or the bind identity is not authorized to access that data.

elogevent Command

Purpose

Logs event information generated by the event response resource manager (ERRM) to a specified log file.

Syntax

elogevent [-h] *log_file*

Description

The `elogevent` captures event information that is posted by the event response resource manager (ERRM) in environment variables the ERRM generates when an event occurs. This script can be used as an action that is run by an event response resource. It can also be used as a template to create other user-defined actions. This script always return messages in English.

Event information that is returned about the ERRM environment variables includes the following:

Local Time
Time when the event or rearm event is observed. The actual environment variable supplied by ERRM is ERRM_TIME. This value is localized and converted to readable form before being displayed.

This script uses the `alog` command to write event information to and read event information from the specified *log_file*.

Flags

-h
Writes the script's usage statement to standard output.

Parameters

log_file
Specifies the name of the file where event information is logged. An absolute path for the *log_file* parameter should be specified.

The *log_file* is treated as a circular log and has a fixed size of 64KB. When *log_file* is full, new entries are written over the oldest existing entries.

If *log_file* already exists, event information is appended to it. If *log_file* does not exist, it is created so that event information can be written to it.

Exit Status

0
The script has run successfully.

1
A required *log_file* is not specified.

2
The *log_file* path is not valid.

Restrictions

- This script must be run on the node where the ERRM is running.
- The user who runs this script must have write permission for the *log_file* where the event information is logged.

Standard Output

When the `-h` flag is specified, the script's usage statement is written to standard output.

Examples

1. To log information, specify the log file as `/tmp/event.log`. ERRM runs this command:

```
/opt/rsct/bin/elogevent/tmp/event.log
```

The `/tmp/event.log` file does not need to exist when the command is run.

2. To see the contents of the /tmp/event.log file, run this command:

```
alog -f /tmp/event.log -o
```

The following sample output shows a warning event for the /var file system (a file system resource):

```
===============================================================================
Event reported at Mon Mar 27 16:38:03 2007

Condition Name:                         /var space used
Severity:                               Warning
Event Type:                             Event
Expression:                             PercentTotUsed>90

Resource Name:                          /var
Resource Class Name:                    IBM.FileSystem
Data Type:                              CT_UINT32
Data Value:                             91
```

Location

/opt/rsct/bin/elogevent

emgr Command

Purpose

Starts the interim fix manager, which installs, removes, lists, and checks system interim fixes.

Syntax

emgr **-l** [**-L** *Label* | **-n** *interimfixNumber* | **-u** *VUID*] [**-v**{**1**|**2**|**3**}] [**-X**] [-a *path*]

emgr **-e** *interimfixPackage* | **-f** *ListFile* [**-w** *Directory*] [**-b** | **-k** | **-I**] [**-p**] [**-q**] [**-m**] [**-o**] [**-X**] [-a *path*]

emgr **-i** *interimfixPackage* | **-f** *ListFile* [**-w** *Directory*] [**-C**] [**-p**] [**-q**] [**-X**] [**-a** *path*]

emgr **-C** **-i** *interimfixPackage* | **-f** *ListFile* [**-w** *Directory*] [**-p**] [**-q**] [**-X**] [**-a** *path*]

emgr **-C** **-L** *Label* [**-p**] [**-q**] [**-X**]

emgr **-r** **-L** *Label* | **-n** *interimfixNumber* | **-u** *VUID* | **-f** *ListFile* [**-w** *Directory*] [-a *path*] [**-b** | **-k** | **-I**] [**-p**] [**-q**] [**-X**]

emgr **-c** [**-L** *Label* | **-n** *interimfixNumber* | **-u** *VUID* | **-f** *ListFile*] [**-w** *Directory*] [-a *path*] [**-v**{**1**|**2**|**3**}] [**-X**]

emgr **-M** | **-U** [**-L** *Label* | **-n** *interimfixNumber* | **-u** *VUID* | **-f** *ListFile*] [**-w** *Directory*] [-a *path*] [**-X**]

emgr **-R** *interimfixLabel* [**-w** *Directory*] [-a *path*] [**-X**]

emgr **-P** [*Package*] [-a *path*] [**-X**]

emgr **-d** **-e** *interimfixPackage* | **-f** *ListFile* [**-w** *Directory*] [**-v**{**1**|**2**|**3**}]

Description

The **emgr** (interim fix manager) command can be used to install and manage system interim fixes. The interim fix manager installs packages created with the **epkg** command and maintains a database containing interim fix information. The **emgr** command performs the following operations:

- interim fix package installation
- interim fix removal
- interim fix listing
- interim fix checking
- interim fix mounting

- interim fix unmounting
- package locks displaying
- installed interim fix forced removal

Note:

- If an attempt is made to update a fileset (using the **installp**, **install_all_updates**, or **smit update_all** command) that has been locked by the interim fix manager, a notice will be displayed indicating which filesets are locked. In some cases, there is no notice to indicate why a fileset was prevented from being installed. The **lslpp** command shows that any locked filesets are in the IFIXLOCKED state.
- Any library or executable program updated by an interim fix or service update which is in use by an active process will not be reflected in that process unless it is restarted. For example, an update that changes the ksh will not have the changes reflected in any ksh processes that are already running. Likewise, an update to the **libc.a** library will not be reflected in any process that is already running. In addition, any process that is using a library and does a **dlopen** operation of the same library after the library has been updated could experience inconsistencies if it is not restarted.

Referencing an Ifix

The ways to reference an interim fix are as follows:

Reference by Label

Each interim fix that is installed on a given system will have a unique interim fix label. This is the unique key that binds all of the different database objects. To reference an interim fix by label, pass the label as a parameter to the **-L** flag. For example, to run a check operation on an interim fix with label **ABC123**, enter:

```
emgr -cL ABC123
```

Reference by Ifix ID

Each interim fix that is installed on a given system has an interim fix ID. The interim fix ID is simply the order number in which the interim fix is listed in the interim fix database. Using this option may be convenient if you are performing operations on interim fixes based on interim fix listings. The **emgr** command will convert the interim fix ID into an interim fix label before performing the given operation. To reference an interim fix by ID, pass the ID as an parameter to the **-n** flag.

Note: Ifix IDs can change as interim fixes are removed and added. Always verify the current interim fix ID number by using the **-l** flag to list the specific interim fix or all interim fixes.

For example, to run a check operation on the first interim fix with ID equal to 1, enter:

```
emgr -cn1
```

Reference by VUID

Because interim fix packages are not formally tracked by any entity, it is possible that the same interim fix label could be used for more than one interim fix package. However, the **emgr** command does not accept the installation of more than one interim fix with the same interim fix label at the same time. The VUID (Virtually Unique ID) can be used to differentiate packages with the same interim fix label. The **emgr** command converts the VUID into an interim fix label before performing the given operation. For example, to list an installed interim fix with VUID equal to **000775364C00020316020703**, enter:

```
emgr -l -u 000775364C00020316020703
```

Note: The VUID is displayed in the preview phase of interim fix installation and removal. The VUID is also displayed when listing with verbosity level set to 3 with the **-v** flag.

Ifix Logging

The following operations are logged to the **emgr** command log file, **/var/adm/ras/emgr.log**:

- Installation
- Removal

- Checking
- Mounting
- Unmounting
- Forced Removal

Enabling Automatic Interim Fix Removal by installp

An interim fix can be packaged by the **epkg** command to contain an APAR reference file containing APAR reference numbers. An APAR reference number will allow **installp** to map an interim fix back to the APARs for all the Technology Levels where the fix was shipped. If **installp** determines that the interim fix is contained in the Technology Level, Service Pack, or PTF being applied, **installp** will automatically remove the interim fix prior to applying the updates.

If an interim fix is enabled for automatic removal, the **emgr** command will display the following message during the installation of the interim fix:

```
ATTENTION: Interim fix is enabled for automatic removal by installp.
```

Concurrent Updates

The **emgr** command supports the installation of a new kind of interim fix called a concurrent update. This form of interim fix contains a modification to the AIX kernel, or one of its kernel extensions, that can be applied directly to the system memory and does not require the system to be rebooted. This direct patching to the system memory allows you to safely evaluate and test a kernel modification without modifying the file containing the system's current kernel on the disk. Any concurrent update applied to the system memory will not persist after a system reboot unless you choose to commit the changes introduced by the concurrent update to the disk using the **-C** flag. You can apply a concurrent update directly over another patch for the same module. You do not need to remove the previous patch. However, there must be only one version of the module loaded. Also, you cannot run any concurrent update operations (in-memory or on disk) for interim fixes in the REBOOT_REQUIRED state until the system is rebooted.

The **emgr** command supports the apply of in-memory concurrent updates on NIM thin servers (diskless or dataless clients). Since thin servers share operating system files with other clients (**/usr** directory is read-only), the **emgr** option to commit a concurrent update to disk (**-C** flag) is not valid on a thin servers.

Note: If the shared operating system files of a thin server need to be patched to disk, an interim fix may be applied to the SPOT resource on the NIM master that serves the thin server. Please refer to the Installing an Interim Fix into a SPOT resource section of Installation Guide or the **/usr/lpp/bos.sysmgt/nim/README file** (NIM IFIX/EMGR section) on your NIM master for details on installing an interim fix into a SPOT.

The **emgr** database will be located in the **/var/emgrdata** directory on thin servers, since the **/usr** file system is read-only on thin servers.

Certain **emgr** operations can not be supported in a thin server environment, such as bosboot and file system expansion. As a result, the following **emgr** flags are not supported in a thin server environment: **-C**, **-e**, **-I**, **-k**, and **–X**. Also, the **–b** flag, which skips the bosboot process for interim fixes that require rebooting, will always be utilized when applicable since the bosboot operation cannot be supported for thin servers.

Flags

Item	Description
-a *path*	Specifies an alternative directory path for installation. **Note:** The **-a** flag works during the removal of an interim fix only if the **-e** and **-a** flags of the **emgr** command were used during the installation of the interim fix. If the interim fix was not installed by using the **-e** and **-a** flags, the **emgr** command does not completely remove an interim fix from the alternative directory path. As a workaround, use the following command to remove an interim fix that was installed in the alternative directory: `chroot /alt_inst /usr/sbin/emgr -r -L efix_label`
-b	Causes the **emgr** command to skip the usual AIX **bosboot** process for interim fixes that require rebooting.
-c	Specifies the check operation. Instructs the **emgr** command to run a check operation on the specified interim fix or interim fixes.
-C	Commits an interim fix containing concurrent updates to the disk. This option must be used along with the **-i** option, or can be used after an interim fix has been applied with the **-i** option. This causes the concurrent updates to persist across system reboots. After a concurrent update has been committed, removal will result in the module being restored to its original un-patched state, regardless of whether other patches for the module exist or not. All prior patches for the module are removed at the time the commit is performed.
-d	Displays the contents and topology. This option is useful with the -v flag in displaying verbosity output.
-e *interimfixPackage*	Specifies the path of the interim fix package file, and installs the interim fix package. The interim fix package file must be created with the **epkg** command and must end with the 16-bit compression extension, **.Z**.
-f *ListFile*	Specifies a file that contains one of the following: • A list of package locations for the installation operation (one per line) • A list of interim fix labels for the remove, mount, unmount, and check operations (one per line) The **emgr** command ignores any blank lines or lines where the first non-white-space character is the **#** character.
-i *interimfixPackage*	Specifies the path of an interim fix package file containing a concurrent update, and applies the concurrent update to the system memory. The update does not persist across system reboots unless the **-C** flag is used. You can also use the **-i** flag to apply one concurrent update over another for the same module. Such a concurrent update is termed a "follow-on".
-I	Runs the low-level debugger for AIX **bosboot** by using the **bosboot** command's **-I** flag.
-k	Loads the low-level debugger during AIX **bosboot** using the **bosboot** command's **-D** flag.
-l	Instructs the **emgr** command to run the list operation on the specified interim fix or interim fixes.
-L *Label*	Selects the interim fix for this operation by interim fix label.
-m	Instructs the **emgr** command to perform a mount installation. When and interim fix is mount-installed, the interim fix files are mounted over the target files.

Item	Description
-M	Instructs the **emgr** command to mount an interim fix or interim fixes that have been mount-installed by using the **-m** flag. The **-M** flag can be used to mount an interim fix that was installed using the **-m** flag and has been unmounted by the **-U** flag or by some other means, such as rebooting the system.
-n *interimfixID*	Selects the interim fix for this operation by specifying the interim fix ID.
-o	Specifies that the interim fix installation can overwrite an existing package.
-p	Instructs the **emgr** command to perform a preview for either installation or removal. The preview runs all of the check operations, but does not make any changes.
-P [*Package*]	Specifies the package-view operation, which displays all packages that are locked by the interim fix manager, their installer, and the locking label or labels.
-q	Suppresses all output other than errors and strong warnings.
-r	Instructs the **emgr** command to run a remove operation on the specified interim fix or interim fixes. Removal of an active patch reinstates any prior patch for the module, provided one exists. If no prior patch exists, the module is restored to its original un-patched state.
-R *Label*	Instructs the **emgr** command to run a force-remove operation. This option removes interim fix data and package locks associated with the interim fix label without actually removing interim fix files, running any remove scripts, or boot processing. This option can be used for only one interim fix at a time. The interim fix label is required to identify the target interim fix. **Attention:** • This method of interim fix removal should be considered an emergency procedure. Because this method can create inconsistencies on the target system, the force remove method should be used only if all other methods of removing the interim fix are unsuccessful. • You must use the standard removal process (**-r** flag) to remove an installed interim fix. In emergency procedures, you can use the **-R** flag to force-remove a label. The **-R** flag requires the **-F** flag to remove a label. When you specify the **-F** flag with the **-R** flag, for example, `emgr -FR` *`ifix_label`*, the force-remove option does not delete any of the interim fix files, saved data, or execute remove scripts. This option must be used only if the standard remove process cannot be accomplished.
-u *VUID*	Selects the interim fix for this operation by specifying the VUID.
-U	Instructs the **emgr** command to unmount an interim fix or interim fixes that have been mount-installed by using the **-m** flag.
-v{**1**|**2**|**3**}	Specifies the verbosity level for the listing operation or the verification level for the check operation. Valid levels are 1, 2, and 3.
-w *Directory*	Instructs the **emgr** command to use the specified working directory instead of the default **/tmp** directory.

Item	Description
-X	Attempts to expand any file systems where there is insufficient space to perform the requested **emgr** operation. This option expands file systems based on available space and size estimates that are provided by the interim fix package and the **emgr** command. **Note:** 1. It is possible to exhaust available disk space during an installation even if the **-X** flag is used. This is more likely if other files are being created or expanded in the same file systems during an installation. 2. Remote file systems cannot be expanded by the **emgr** command.

Exit Status

0
All of the **emgr** command operations completed successfully.

>0
An error occurred.

Security

System administrators or users with the **aix.system.install** authorization can run the **emgr** command on a multi-level secure (MLS) system. Ifix data, saved files, and temporary files are accessible only by the root user.

The **emgr** command looks for a supported MD5 generating command on the system. If one is located, the **emgr** command displays the MD5 checksum to the user. The user can then cross check this MD5 sum with a secured source. If an MD5 generating command is not located, the **emgr** command takes no further action.

The user can force set the path to an MD5 command by exporting the EMGR_MD5_CMD shell variable. This variable should contain the absolute path to the MD5 generating command.

Notes:

- This feature is not supported in the original release of interim fix management. It is recommended that the user updates to the latest level of interim fix management by updating **bos.rte.install** to the latest level.
- If Trusted Execution (TE) policy is turned on along with the TSD_LOCK policy or the TSD_FILE_LOCK policy, the **emgr** command fails. To continue with the installation,manually turn off the TSD_LOCK policy or the TSD_FILE_LOCK policy. The **emgr** command runs successfully with TE policies other than the TSD_LOCK policy or the TSD_FILE_LOCK policy.

  Also, when a TE policy is turned on, only one instance of the **emgr** command is supported.

Examples

1. To preview the installation of an interim fix package called **games.020303.epkg.Z**, enter:

   ```
   emgr -p -e games.020303.epkg.Z
   ```

2. To install the interim fix package called **games.020303.epkg.Z** and automatically expand file systems if additional space is needed, enter:

   ```
   emgr -X -e games.020303.epkg.Z
   ```

3. To list all interim fixes on the system, enter:

   ```
   emgr -l
   ```

4. To do a level 3 listing of interim fix label **games**, enter:

```
emgr -lv3 -L games
```

5. To remove the interim fix with label **games**, enter:

```
emgr -r -L games
```

6. To preview the removal of the interim fix labels in file **/tmp/myfixes**, enter:

```
emgr -rp -f /tmp/myfixes
```

7. To check all interim fixes with verification level 2, enter:

```
emgr -cv2
```

8. To check interim fix ID number 3 with verification level 1 (the default verification level), enter:

```
emgr -c -n3
```

9. To check interim fix with VUID of **000775364C00020316020703** and verification level 3, enter:

```
emgr -u 000775364C00020316020703 -c -v3
```

10. To list all locked packages and their interim fix labels, enter:

```
emgr -P
```

11. To list all interim fix labels that have locked the **installp** package **bos.rte.lvm**, enter:

```
emgr -P bos.rte.lvm
```

12. To mount-install the interim fix package called **games.020303.epkg.Z** and suppress AIX **bosboot**, enter:

```
emgr -e games.020303.epkg.Z -mb
```

13. To mount all interim fix files that have been mount-installed on the system by using the **-m** option, enter:

```
emgr -M
```

14. To unmount all interim fix files associated with interim fix label **games**, enter:

```
emgr -U -L games
```

15. To apply an interim fix package called **kernelmod.031007.epkg.Z** with concurrent updates to the system memory, enter:

```
emgr -i kernelmod.031007.epkg.Z
```

16. To commit the concurrent updates associated with the interim fix label **kernelmod** to the disk, enter:

```
emgr -C -L kernelmod
```

17. To apply an interim fix package called **kernelmod2.031007.epkg.Z** with concurrent updates to the system memory, and also to commit the concurrent updates to the disk, enter:

```
emgr -i kernelmod2.031007.epkg.Z -C
```

18. To display level 3 verbosity output on interim fix package `test.102403.epkg.Z`, enter:

```
emgr -v3 -d test.102403.epkg.Z
```

Files

Item	Description
/usr/sbin/emgr	Contains the **emgr** command
/usr/emgrdata/DBS/ifix.db	Contains the interim fix header database
/usr/emgrdata/DBS/files.db	Contains the interim fix files database
/usr/emgrdata/DBS/pkglck.db	Contains the package locks database
/usr/emgrdata/DBS/prereq.db	Contains the prerequisite database
/usr/emgrdata/DBS/e2eprereq.db	Contains the interim fix prerequisite database
/usr/emgrdata/DBS/aparref.db	Contains the APAR reference file database

emstat Command

Purpose

Shows emulation exception statistics.

Syntax

emstat [**-a** | **-v**] [*Interval*] [*Count*]

Description

The **emstat** command displays emulation exception statistics. Emulation exceptions can occur when some existing applications or libraries, which contain instructions that have been deleted from older processor architectures, are executed on newer processors. These instructions may cause illegal instruction program exceptions. The operating system catches these exceptions and emulates the older instruction(s) to maintain program functionality, potentially at the expense of program performance.

The emulation exception count since the last time the machine was rebooted and the count in the current interval are displayed. The user can optionally display alignment exception statistics or individual processor emulation statistics.

The default output displays statistics every second. The sampling interval and number of iterations can also be specified.

Parameters

Item	Description
Interval	Interval between samples.
Count	Number of iterations.

Flags

Item	Description
-a	Displays alignment exception statistics. This flag cannot be used with the **-v** flag.
-v	Display individual processor statistics. This flag cannot be used with the **-a** flag.

Examples

1. To display the emulation statistics every second, type:

```
emstat
```

This produces the following output:

```
  Emulation  Emulation
  SinceBoot      Delta
    8845591          0
    8845591          0
    8845591          0
    8845591          0
    8845591          0
    8845591          0
  ...
```

2. To display emulation and alignment exception statistics every two seconds, a total of 5 times, type:

```
emstat -a 2 5
```

This produces the following output:

```
  Alignment  Alignment  Emulation  Emulation
  SinceBoot      Delta  SinceBoot      Delta
   21260604          0   70091846          0
   23423104    2162500   72193861    2102015
   25609796    2186692   74292759    2098898
   27772897    2163101   76392234    2099475
   29958509    2185612   78490284    2098050
```

3. To display emulation statistics, every 5 seconds, for each processor, type:

```
emstat -v 5
```

This produces the following output:

```
  Emulation  Emulation  Emulation  Emulation
  SinceBoot      Delta    Delta00    Delta01
   88406295          0          0          0
   93697825    5291530          0    5291530
   98930330    5232505    5232505          0
  102595591    3665261     232697    3432564
  102595591          0          0          0
```

emsvcsctrl Command

Purpose

Starts the event management subsystem.

Syntax

`emsvcsctrl [-a | -s | -k | -d | -c | -t | -o | -h ]`

Description

`emsvcsctrl` is a control script that starts the event management subsystem. Event management is a distributed subsystem of RSCT that provides a set of high-availability services for the IBM RS/6000 server. By matching information about the state of system resources with information about resource conditions that are of interest to client programs, it creates events. Client programs can use events to detect and recover from system failures, thus enhancing the availability of the system. The emsvcsctrl control script controls the operation of the Event Management subsystem. The subsystem is under the control of the System Resource Controller (SRC) and belongs to a subsystem group called emsvcs. A daemon is associated with each subsystem. The emsvcsctrl script also controls the operation of the AIX Resource Monitor subsystem. The subsystem is under SRC control and also belongs to the emsvcs subsystem group. A daemon is associated with each subsystem.

Instances of the Event Management and AIX Resource Monitor subsystems execute on each node in the HACMP/ES cluster. From an operational point of view, the Event Management subsystem group is organized as follows:

Subsystem
Event Management

Subsystem Group
emsvcs

SRC Subsystem
The emsvcs subsystem is associated with the haemd daemon.

emaixos
The emaixos is associated with the harmad daemon.

Daemons
The haemd daemon provides the Event Management services. The harmad daemon is the resource monitor for AIX operating system resources.

The emsvcsctrl script is not normally executed from the command line. It is normally called by the HACMP/ES startup script command during installation of the system.

The emsvcsctrl script provides a variety of controls for operating the Event Management subsystem:

- Adding, starting, stopping, and deleting the subsystem
- Cleaning up the subsystems
- Turning tracing on and off

`Adding the Subsystem:` When the -a flag is specified, the control script uses the mkssys command to add the Event Management and AIX Resource Monitor subsystems to the SRC. The control script operates as follows:

1. It makes sure that the emsvcs and emaixos subsystems are stopped.
2. It removes the emsvcs and emaixos subsystems from the SRC (just in case they are still there).
3. It adds the emsvcs subsystem to the SRC.
4. It adds the emaixos subsystem to the SRC.
5. It adds haemrm group using the mkgroup command, if it does not already exist. Any errors that occur are written to a log file named /var/ha/log/em.mkgroup.
6. It creates the /var/ha/lck/haem and /var/ha/soc/haem directories, if they don't already exist. Any errors that occur are written to a log file named /var/ha/log/em.mkdir.
7. It copies the Event Management Configuration Database, (EMCDB) from its install location, /opt/rsct/install/config/em.HACMP.cdb to its run-time location, /etc/ha/cfg/em.HACMP.cdb. Any errors resulting from the copy are written to a log file named /var/ha/log/em.cp.

`Starting the Subsystem:` When the -s flag is specified, the control script uses the startsrc command to start the Event Management subsystem, emsvcs, and the AIX Resource Monitor subsystem, emaixos.

`Stopping the Subsystem:` When the -k flag is specified, the control script uses the stopsrc command to stop the Event Management subsystem, emsvcs, and the AIX Resource Monitor subsystem, emaixos.

`Deleting the Subsystem:` When the -d flag is specified, the control script uses the rmssys command to remove the Event Management and AIX Resource Monitor subsystems from the SRC. The control script operates as follows:

1. It makes sure that the emsvcs and emaixos subsystems are stopped.
2. It removes the emsvcs and emaixos subsystems from the SRC using the rmssys command.

`Cleaning Up the Subsystems:` When the -c flag is specified, the control script stops and removes the Event Management subsystems for all system partitions from the SRC. The control script operates as follows:

1. It stops all instances of subsystems in the subsystem group by using the stopsrc -g emsvcs command.

2. It removes all instances of subsystems in the subsystem group from the SRC using the rmssys command.
3. It removes the Event Management Configuration Database (EMCDB) from its run-time location, /etc/ha/cfg/em.HACMP.cdb.

`Turning Tracing On:` When the -t flag is specified, the control script turns tracing on for the haemd daemon, using the haemtrcon command. Tracing for the harmad daemon is also enabled, using the traceson command.

`Turning Tracing Off:` When the -o flag is specified, the control script turns tracing off for the haemd daemon, using the haemtrcoff command. Tracing for the harmad daemon is also disabled, using the tracesoff command.

`Logging:` While it is running, the Event Management daemon normally provides information about its operation and errors by writing entries to the AIX error log. If it cannot, errors are written to a log file called /var/ha/log/em.default.cluster_name.

Flags

-a
Adds the subsystem.

-s
Starts the subsystem.

-k
Stops the subsystem.

-d
Deletes the subsystem.

-c
Cleans the subsystem.

-t
Turns tracing on for the subsystem.

-o
Turns tracing off for the subsystem.

-h
Displays usage information.

Security

You must be running with an effective user ID of `root`.

Exit Status

0
Indicates the successful completion of the command.

1
Indicates that an error occurred.

Restrictions

This command is valid in an HACMP environment only.

Standard Error

This command writes error messages (as necessary) to standard error.

Examples

1. To add the Event Management subsystem to the SRC, enter:

```
emsvcsctrl -a
```

2. To start the Event Management subsystem, enter:

```
emsvcsctrl -s
```

3. To stop the Event Management subsystem, enter:

```
emsvcsctrl -k
```

4. To delete the Event Management subsystem from the SRC, enter:

```
emsvcsctrl -d
```

5. To clean up the Event Management subsystem, enter:

```
emsvcsctrl -c
```

6. To turn tracing on for the Event Management daemon, enter:

```
emsvcsctrl -t
```

7. To turn tracing off for the Event Management daemon, enter:

```
emsvcsctrl -o
```

Location

/opt/rsct/bin/emsvcsctrl
Contains the `emsvcsctrl` script

Files

/var/ha/log/em.default.cluster_name
Contains the default log of the haemd daemon on the cluster named cluster_name.

/var/ha/log/em.cp
Contains a log of any errors that occurred while copying the Event Management Configuration Database.

/var/ha/log/em.trace.cluster_name
Contains the trace log of the haemd daemon on the cluster named cluster_name.

/var/ha/log/em.mkgroup
Contains a log of any errors that occurred while creating the haemrm group.

/var/ha/log/em.mkdir
Contains a log of any errors that occurred while creating the /var/ha/lck/haem and /var/ha/soc/haem directories.

enable Command

The **enable** command includes information for **AIX Print Subsystem enable** and the **System V Print Subsystem enable**.

AIX Print Subsystem enable Command

Purpose

Enables printer queue devices.

Syntax

enable *PrinterName ...*

Description

The **enable** command brings the printer queue devices specified by the *PrinterName* parameter on line, or enables the printer queue devices to be used with the system.

Notes:

1. You must have root user authority or belong to the printq group to run this command.
2. If you enter `enable -?`, the system displays the following error message:

```
enq: (FATAL ERROR): 0781-048: Bad queue or device name: -?
```

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To enable the print queue device `lp0:lpd0`, enter:

```
enable lp0:lpd0
```

Files

/etc/qconfig
Contains the queue configuration file.

/etc/qconfig.bin
Contains the digested, binary version of the **/etc/qconfig** file.

/usr/sbin/qdaemon
Contains the queuing daemon.

/var/spool/lpd/qdir/*
Contains the queue requests.

/var/spool/lpd/stat/*
Contains information on the status of the devices.

/var/spool/qdaemon/*
Contains temporary copies of enqueued files.

System V Print Subsystem enable Command

Purpose (System V Print Subsystem enable Command)

Enable LP printers

Syntax (System V Print Subsystem enable Command)

enable ***printers***

Description (System V Print Subsystem enable Command)

The **enable** command activates the named ***printers***, enabling them to print requests submitted by the **lp** command. If the printer is remote, the command will only enable the transfer of requests to the remote system; the **enable** command must be run again, on the remote system, to activate the printer. (Run **lpstat -p** to get the status of printers.)

When changes are made to the attributes of a print device, they are recognized by **enable**. Therefore to change the definition or allocation for a device, you must disable the printer on that device, change the device, and then run **enable**. The new device attributes will become effective when **enable** is executed.

Printer names are *system-defined words* and as such should be restricted to uppercase and lowercase ASCII characters.

If you enter `enable -?`, the system displays the command usage message and returns 0.

Security (System V Print Subsystem enable Command)

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Files (System V Print Subsystem enable Command)

/var/spool/lp/*

enotifyevent Command, notifyevent Command

Purpose

Mails event information generated by the event response resource manager (ERRM) to a specified user ID.

Syntax

`enotifyevent` [`-h`] [*user-ID*]

`notifyevent` [`-h`] [*user-ID*]

Description

The `enotifyevent` script always return messages in English. The language in which the messages of the `notifyevent` script are returned depend on the locale settings.

These scripts capture event information that is posted by the event response resource manager (ERRM) in environment variables that are generated by the ERRM when an event occurs. These scripts can be used as actions that are run by an event response resource. They can also be used as templates to create other user-defined actions.

Event information is returned about the ERRM environment variables, and also includes the following:

Local Time
Time when the event or rearm event is observed. The actual environment variable supplied by ERRM is ERRM_TIME. This value is localized and converted to readable form before being displayed.

In AIX, these scripts use the `mail` command to send event information to the specified user ID. When a user ID is specified, it is assumed to be valid, and it is used without verifying it. If a user ID is not specified, the user who is running the command is used as the default.

user-ID is the optional ID of the user to whom the event information will be mailed. If *user-ID* is not specified, the user who is running the command is used as the default.

Flags

-h
Writes the script's usage statement to standard output.

Parameters

log_file
Specifies the name of the file where event information is logged. An absolute path for the *log_file* parameter should be specified.

For AIX, the *log_file* is treated as a circular log and has a fixed size of 64KB. When *log_file* is full, new entries are written over the oldest existing entries.

For other platforms, the size of the *log_file* is not limited, and it will not overwrite itself. The file size will increase indefinitely unless the administrator periodically removes entries.

If *log_file* already exists, event information is appended to it. If *log_file* does not exist, it is created so that event information can be written to it.

Exit Status

0
Command has run successfully.

Restrictions

1. These scripts must be run on the node where the ERRM is running.
2. The `mail` command is used to read the file.

Standard Output

When the `-h` flag is specified, the script's usage statement is written to standard output.

Examples

1. You can use the **mail** command to read the contents of the event information. The following example shows how a warning event for the `/var` file system (a file system resource) is formatted and logged:

```
============================================================================
Event reported at Sun Mar 26 16:38:03 2002

Condition Name:                      /var space used
Severity:                   Warning
Event Type:                    Event
Expression:                    PercentTotUsed>90

Resource Name:                   /var
Resource Class Name:               IBM.FileSystem
Data Type:                    CT_UINT32
Data Value:                    91
```

Location

/opt/rsct/bin/enotifyevent
Contains the `enotifyevent` script

/opt/rsct/bin/notifyevent
Contains the `notifyevent` script

enq Command

Purpose

Enqueues a file.

Syntax

To process a file

enq [**-**] [**-B** *CharacterPair*] [**-c**] [**-C**] [**-G**] [**-j**] [**-m** *Text*] [**-M** *File*] [**-n**] [**-N** *Number*] [**-o** *Option*] [**-P** *Queue*] [**-r**] [**-R** *Number*] [**-t** "*User*"] [**-T** *Title*] [**-Y**] [**-Z** *Name*] *File*

To change the priority of print jobs

enq -a *Number* **-#** *JobNumber*

To display status

enq [**-q** | **-A**] [**-L**] [**-W**] [**-e**] [**-#** *JobNumber*] [**-u** *Name*] [**-w** *Seconds*] [**-s**]

To change queue and queue daemon status

enq [**-d**] [**-D**] [**-G**] [**-K**] [**-L**] [**-q** | **-A**] [**-U**]

To cancel options

enq [**-X**] [**-x***Number*] [**-P***Printer*]

To hold, release, or move a print job to another queue

enq { **-h** | **-p** | **-Q** *NewQueue* } { **-#** *JobNumber* [**-P** *Queue*] | **-u** *User* | **-P** *Queue* }

To queue and hold a print job

enq -H *File ...*

Description

The **enq** command is a general-purpose utility for enqueuing requests to a shared resource, typically a printer device. Use the **enq** command to enqueue requests, cancel requests, alter the priority of a request, and display the status of queues and devices.

The **enq** command has five different syntax diagrams because all the flags are not meant to work together. Some of these flags are meant for file processing and accept *FileName* as an option. The other flags are used for changing the priority of a print job, displaying the status, changing the status of the queue or the queue daemon, and canceling a print job.

To enqueue files on a specific queue, use the **-P** flag (**-P** *Queue*). If more than one device services a queue, you can also request a particular device by specifying that device (*:device*) after the name of the queue. If you do not specify a device, the job is sent to the first available device. If you do not specify a file, the **enq** command copies standard input into a file and enqueues it for printing.

The **enq** command requests can have operator messages that are associated with them. This feature is useful in a distributed environment or on a system with many users. The messages are used to tell the printer operator such information as a request to load a special form or different color paper into the printer before allowing the job to print. These messages are specified with the **-m** and **-M** flags. The **qdaemon** command processes the **enq** command requests. When the **qdaemon** is ready to begin a request that has an associated message, the system displays the message on the console of the machine where the **qdaemon** process is running. The text of the message is accompanied by a prompt that tells the printer operator how to signal the request to continue or how to cancel the request.

The display that is generated by the **enq -A** command contains two entries for remote queues. The first entry contains the client's local queue and local device name and its status information. The second entry follows immediately; it contains the client's local queue name (again), followed by the remote queue

name. Any jobs that are submitted to a remote queue are displayed first on the local side and are moved to the remote device as the job is processed on the remote machine.

Since the status commands communicate with remote machines, the status display might occasionally appear to hang when waiting for a response from the remote machine. The command eventually times out if a connection cannot be established between the two machines.

Notes:

1. Before you can enqueue a file, you must have read access to it. To remove a file, (see the **-r** flag) you must also have write access to the directory that contains the file.
2. If you want to continue changing the file after you issue the **enq** command but before it is printed, you must use the **-c** flag.
3. When enqueuing files on a printer, flags can be interspersed in any order.
4. The **-d** and **-G** flags are acted upon immediately. Syntax error that appears before these flags on the command line are reported. Syntax errors that appears after these flags on the command line are ignored.

Flags

File processing options

If you give the **enq** command a list of file names, it enqueues them all for file processing on the default device or on the specified device.

Item	Description
-	Causes the **enq** command to act as a filter. The **enq** command automatically reads standard input if you do not specify a file or files. However, if you do specify a file, you can also use the dash (**-**) to force the **enq** command to read standard input. The dash (**-**) is actually not a flag, but a special type of file name. Therefore, it must come after all other flags are specified on the command line.

Item	Description
-B *CharacterPair*	Controls the printing of burst pages according to the value of *CharacterPair* as follows. (**n** = never, **a** = always, **g** = group. The first character is for header, the second character is for trailer.) **HT** Description **nn** No headers, no trailers **na** No headers, trailer on every file **ng** No header, trailer at the end of the job **an** Header on every file, no trailers **aa** Headers and trailers on every file in the job **ag** Header on every file, trailer after job **gn** Header at the beginning of job, no trailer **ga** Header at beginning of job, trailer after every file **gg** Header at beginning of job, trailer at end of job The header and trailer stanzas in the **/etc/qconfig** file define the default treatment of burst pages. **Note:** In a remote print environment, the default is to print a header page and not a trailer page.
-c	Copies the file. To save disk space, the **enq** command remembers the name of the file, but does not actually copy the file itself. Use the **-c** flag if you want to continue changing the file while you are waiting for the current copy to be printed.
-C	Specifies that the **mail** command is to be used instead of the **write** command for error messages and job completion notification. (Using this flag is useful for writing PostScript applications since it allows better feedback from the printer.) Error messages and job completion messages (both generated by the **piobe** command) and any data read from the printer are also sent back by mail. The **-C** flag applies only to local print jobs. If you want to be notified when a job sent to a remote printer is completed, use the **-n** flag to receive a mail message. **Note:** There are some messages that cannot be redirected from **qdaemon** and the printer back-end in any way. These messages are system errors and are sent directly to the /**dev**/**console** file.
-j	Specifies that the message `Job number is: nnn`, where nnn is the assigned job number, be displayed to standard output. It occurs only if the job is submitted to a local print queue.
-m *Text*	Submits an operator message with an **enq** command request. The specified text contains the message.

Item	Description
-M *File*	Submits an operator message with an **enq** command request. The specified file contains the text of the message.
-n	Notifies you when your job is finished. If the **-t** flag is also used, the **enq** command also notifies the user for whom the request is intended (see the **-t** flag).
-N *Number*	Prints *Number* copies of the file. Normally, a file is printed only once.
-o *Option*	Specifies that flags that are specific to the backend be passed to the backend. Thus, for each queue there are flags that are not described in this section that can be included on the **enq** command line. See the **piobe** command for a list of these flags.
-P *Queue*	Specifies the queue to which the job is sent. A particular device on a queue can be specified by typing `-P Queue:Device`.
-r	Removes the file after it is successfully printed.
-R *Number*	Sets the priority of the current job to *Number*. This flag is used at job submission time. Use the **-a** flag to alter priority after the job is submitted. Higher numbers assign higher priority. The default priority is 15. The maximum priority is 20 for most users and 30 for the users with root user authority.
-t "*User*"	Labels the output for delivery to *User*. Normally the output is labeled for delivery to the user name of the person who is issuing the **enq** command request. The value of *User* must be a single word that meets the same requirements of a regular user ID.
-T *Title*	Puts title on the header page and displays it when the **-q** flag is specified. Normally the job title is the name of the file. If the **enq** command reads from standard input, the job title is **STDIN.#** where **#** is the process ID of the **enq** command.
-Y	Tells the **enq** command to ignore the rest of the command line after this flag. This is useful for discovering whether a queue is valid (if it is in the **/etc/qconfig** file). For example, typing `enq -P lp4 -Y` returns with an exit value of 0 if the line printer **lp4** is a valid queue; if otherwise, a nonzero value is returned. Using this flag is also good for forcing the **qdaemon** command to redigest the **/etc/qconfig** file.
-Z *Name*	Specifies originator of remote print jobs.

Print job priority options

Item	Description
-a *Number*	Changes the priority of the named job to *Number*. The job must be submitted for printing before entering the **enq** command with this flag. See the **-R** flag for a description of priorities. Use the **-#** flag to specify the job number. This flag is only valid for local print jobs.

Item	Description
-# *JobNumber*	Specifies the job number that is used by the **enq -q** command or the **enq -a** command, and displays only the job that is specified in status output. **Note:** 1. Specify the **-P** *Queue* to override the default destination printer. 2. If jobs 1, 2, and 3 are in the printer queue, and you specify that you want the status of job 3 while job 1 is running, the status information shows job 1 and job 3, not only job 3. 3. If you specify a job number that does not exist, the system displays the current job number on the queue instead of an error message.

Display status options

Item	Description
-A	Provides status for all queues. It is like running the **enq -q** command once for each queue in the **qconfig** file.
-e	Excludes status information from queues that are not under the control of the **qdaemon** command. The status from such queues might be in different formats. The **-e** flag can be used with any combination of flags.
-L	Specifies the long status. This flag can be used with the **-A**, **-q**, or **-W** flag. If the **-L** flag and the **-W** flag are used simultaneously, the result displays the long status of the print job in the semicolon-separated format. Use the **-L** flag to show multiple files to be printed in a single print job.
-q	Displays the status of the default queue. The **LPDEST** and **PRINTER** environment variable control the name of the default printer. If the **LPDEST** environment variable contains a value, that value is always used first. If the **LPDEST** variable has no value, the **enq** command uses the **PRINTER** environment variable. If the **PRINTER** environment variable contains no value, then the **enq** command uses the system default. **Notes:** 1. Use the **-P** *Queue* flag with the **-q** flag to display the status of a particular queue. 2. Any destination command-line options override both the **LPDEST** and the **PRINTER** environment variables.
-s	Obtains the status of print queues without listing any files.
-u *Name*	Specifies the user name for which to print job status.
-w *Seconds*	Specifies continuous output of the queue status, updating the screen every *Seconds* specified until the queue is empty (see the **lpq** command). When the queue is empty, the process halts. This flag is only used with either the **-q** flag, or the **-A** flag, or the **-L** flag.
-W	Specifies the wide status format with longer queue names, device names, and job numbers. This flag can be used with the **-A**, **-q**, or **-L** flag. If the **-L** and **-W** flags are used simultaneously, the result displays the long status of the print job in the semicolon-separated format.

Change the queue and queue daemon status options

Item	Description
-d	Runs the **digest** command on the **/etc/qconfig** file. Once the digest is completed, any changes to the **/etc/qconfig** file are reflected in the **/etc/qconfig.bin** file. A user must have root user authority to run this option.

In addition to the previous flags available to all users, the **enq** command accepts the following flags when they are entered by users that have root user authority. Root user authority means that you are root or you belong to the **printq** group.

Note: The following flags can be used only on local print jobs.

Item	Description
-D	Device DOWN. Turns off the device that is associated with the queue. The **qdaemon** process no longer sends jobs to the device, and entering the **enq -q** command shows its status as DOWN. Any job currently running on the device is allowed to finish.
-G	Die GRACEFULLY. Ends the **qdaemon** process after all currently running jobs are finished. Use of this flag is the only clean way to bring the **qdaemon** process down. Use of the **kill** command might cause problems, such as jobs hanging up in the queue. If the **qdaemon** process is running under **srcmstr** (the default configuration), **enq -G** does not prevent **qdaemon** from being restarted automatically. You must use the **chssys** command, which changes the default configuration and prevents the automatic restart of the **qdaemon** process. The following command: `chssys -s qdaemon -O` issued before the **enq -G** command, prevents the automatic restart of the **qdaemon** command. The following command: `startsrc -s qdaemon` restarts the **qdaemon** process manually.
-K	Acts the same as the **-D** flag, except that all current jobs are killed. They remain in the queue, and are run again when the device is turned on.
-L	Specifies the long status. This flag can be used with the **-A**, **-q**, or **-W** flag. Use the **-L** flag to show multiple files to be printed in a single print job.
-U	Brings UP the device that is associated with a queue. The **qdaemon** process sends jobs to it again and entering the **enq -q** command shows its status as ready.

Note: If more than one device is associated with a queue, you must specify the device and the queue when you use the **-D** flag, the **-K** flag, and the **-U** flags. For example, entering `-P lp:lpd` designates the same device only if there is no other device on that queue.

Cancel options

Item	Description
-X	Cancels the printing of your jobs. If you have root user authority, all jobs on the specified queue are deleted. This flag is only valid on local print jobs.
-x *Number*	Cancels the printing of the specified job *Number*.
-P *Printer*	Specifies the *Printer* where either all jobs or the selected job number is to be canceled.

Attention: If you have root user authority and do not specify a queue, all jobs on all queues are deleted.

Holding and releasing a print job options

Item	Description
-# *JobNumber*	Designates the number of the print job to be held or released.
-h	Holds the specified print job.
-H	Queues and holds the file indicated with the *File* parameter.
-p	Releases the specified print job.
-P *Queue*	Designates the print queue to be held or released.
-u *User*	Designates the user whose print jobs are to be held or released.

Moving print job options

Item	Description
-# *JobNumber*	Designates the number of the print job to be moved.
-P *Queue*	Designates the print queue to be moved. The value of the *Queue* variable can be a queue name or in the form queue:device name.
-Q *NewQueue*	Designates the target queue where the print job is moved to. The value of the *NewQueue* variable can be in the form of a queue name or in the form queue:device name.
-u *User*	Designates the user whose print jobs are to be moved.

Security

Auditing Events:

Event	Information
ENQUE_admin	Queue name, device name, job name, user name

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To print the file `memo` on the default printer, enter:

   ```
   enq memo
   ```

2. To print the file `prog.c` with page numbers, enter:

   ```
   pr prog.c | enq
   ```

   The **pr** command puts a heading at the top of each page that includes the date that the file was last modified, the name of the file, and the page number. The **enq** command then prints the file.

3. To print a file with page numbers, reading from standard input, enter:

   ```
   pr x | enq  -P bill  -n  -r fn1 - fn3
   ```

   The dash (**-**) special file name tells the **enq** command to read from standard input. Normally the **enq** command does not read from standard input if there are file names on the command line. It also indicates the order in which to print things. The **pr** command creates a page numbered version of the file x and passes it to the **enq** command, which creates a temporary file that contains that output in the **/var/spool/qdaemon** file.

The **enq** command creates a job with four files and submits it to the queue named `bill`. It prints the `fn1` file twice. Then, it prints whatever the output of the **pr** command was. Lastly, it prints the file `fn3`. The four files are treated as one job for the purposes of burst pages. Notification is sent (the `-n` flag) when the job is complete. Since the `-r` flag was specified, the `fn1` and `fn3` files are removed at job completion. The temporary file that is created by the dash (**-**) file is always deleted.

The **pr** command puts a heading at the top of each page that includes the date that the file was last modified, the name of the file, and the page number. The **enq** command then prints the file.

4. To print the file `report` on the next available printer that is configured for the `fred` queue, enter:

```
enq -P fred report
```

5. To print several files that begin with the prefix `sam` on the next available printer that is configured for the `fred` queue, enter:

```
enq -P fred sam*
```

All files that begin with the prefix `sam` are included in one print job. Normal status commands show only the title of the print job, which in this case is the name of the first file in the queue unless a different value was specified with the **-T** flag. To list the names of all the files in the print job, use the long status command **enq -A -L**.

6. To check the print queue to see whether a file is still waiting to be printed, enter:

```
enq  -q
```

This command displays the status of the user's default queue. If the file is not yet printed, then it appears in the queue status listing. The system default queue is defined as the first queue in the **/etc/qconfig[.bin]** file. Users can have their own default override by setting and exporting the **PRINTER** environment variable.

7. To display the status of a nondefault queue, `lp0`, enter:

```
enq  -q  -P lp0
```

8. To obtain the long queue status, enter:

```
enq  -L
```

9. To obtain status on all queues, enter:

```
enq  -A
```

10. To obtain long status on all queues, enter:

```
enq  -A  -L
```

11. To obtain the status of the default queue, in wide format, enter:

```
enq  -W
```

12. To obtain the wide status of all queues, enter:

```
enq  -W  -A
```

13. To stop printing a job (a job is one or more files), enter:

```
enq  -x 413
```

This command cancels the request that you made earlier to print a job. The number was obtained from the listing that is obtained by entering the **enq -q** command. If the job is being printed, the printer stops immediately. If the job is not printed yet, it is removed from the queue so that it is not printed. If the job is not in the queue, the **enq** command displays a message similar to the following output:

```
no such request from you -- perhaps it's done?
```

14. To disconnect a printer from the queuing system, enter:

```
enq  -P lp0:dlp0  -D
```

Entering this command stops the **enq** command requests from being sent to the printer that serves the `lp0` queue. If a file is printing, it is allowed to finish. You must be able to run the **qadm** command to run the **enq** command.

Note: The printers that serve a specified queue are named by the device stanza name as it appears in the **/etc/qconfig[.bin]** file.

15. To print a file with page numbers by using the **piobe** command backend on the default printer, enter:

```
enq  -o -p filename
```

The `-p` flag is not looked at by the **enq** command. The `-o` flag tells the **enq** command to pass the next item, which can be in quotes, to the backend unchanged. So, the **enq** command passes the `-p` flag to the **qdaemon** process, which in turn passes it to the backend **piobe**. The `-p` flag causes **piobe** to run the **/usr/bin/pr** filter to apply page numbers to the document before giving data to the device. Multiple options can be given in quotes preceded by one **-o** flag. Multiple options can also be given without quotes with each option preceded by the **-o** flag.

16. Assuming a **qconfig** file with the following information:

```
qname:
                device = fred
fred:
                file = /tmp/hello
                backend = /usr/bin/sh /usr/bin/diff
```

And given the following commands:

```
rm /tmp/hello
touch /tmp/hello
pr /etc/hosts|enq  -P qname:fred - /etc/hosts
```

The **qdaemon** process runs the `/usr/bin/diff` program with two arguments, one of which is a temporary file name and the other being the `/etc/hosts` file. The only difference between the two files is that one was run through the **pr** command. The `/tmp/hello` file contains the differences between the two files. The **qdaemon** process does not create the `/tmp/hello` file if it does not exist.

17. The following command:

```
enq  -m'i want pink paper for this job' /etc/passwd
```

sends the specified operator message to the operator's console just before the print job is to print. The operator must respond to this message to continue or cancel the job.

```
enq  -M pink /etc/passwd
```

This command accomplishes the same thing, only the message is contained in a file called `pink`.

18. To cancel all jobs in the `fred` queue, enter:

```
enq  -X  -P fred
```

If the user who entered this command has root user authority, all the jobs from the `fred` queue are deleted. If the user does not have root user authority, only the users jobs are deleted from that queue.

19. To queue the file named `MyFile` and return the `MyFile` job number to the **jdf** file, enter:

```
enq  -j MyFile
```

20. To hold print job number `310`, enter:

```
enq  -h  -#310
```

To release the hold on print job number `310`, enter:

```
enq  -p -#310
```

21. To hold all the print jobs on queue `lp0`, enter:

    ```
    enq  -h  -P lp0
    ```

    To release the `lp0` queue, enter:

    ```
    enq  -p -P lp0
    ```

22. To hold all print jobs that are created by `fred`, enter:

    ```
    enq  -h  -u fred
    ```

    To release the print jobs that are created by `fred`, enter:

    ```
    enq  -p -u fred
    ```

23. To move job number 318 to queue `lp0`, enter:

    ```
    enq  -Q lp0  -#318
    ```

    The flags that control moving print jobs work in the same way as the flags that hold the print files. The hold flags and variables are illustrated in the preceding examples.

Files

Item	Description
/usr/sbin/qdaemon	Queuing daemon.
/etc/qconfig	Queue configuration file.
/var/spool/lpd/qdir/*	Queue requests.
/var/spool/lpd/stat/*	Information about the status of the devices.
/var/spool/qdaemon/*	Temporary copies of enqueued files.
/etc/qconfig.bin	Digested, binary version of the **/etc/qconfig** file.

enroll Command

Purpose

Sets up a password used to implement a secure communication channel.

Syntax

enroll

Description

The **enroll** command establishes a password and secures a communication channel in which messages can only be read by the intended recipient. The password is used to receive secret mail.

The **enroll** command is used with the **xsend** and **xget** commands to send and receive secret mail. The **xsend** command sends secret mail. The **xget** command asks for your password and gives you your secret mail.

Examples

To set up a password, enter:

```
enroll
```

When prompted, enter your password. This allows other users on your system to send you secret mail. Use the **xget** command to read the secret mail.

Files

Item	Description
/var/spool/secretmail/User.key	Contains the encrypted key for the user.
/usr/bin/enroll	Contains the **enroll** command.

enscript Command

Purpose

Converts text files to PostScript format for printing.

Syntax

enscript [**-1** **-2** **-c** **-g** **-k** **-l** **-m** **-o** **-q** **-r** **-B** **-G** **-K** **-R**] [**-b** *Header*] [**-f** *Font*] [**-f0** *CodeSet*:*Font*] [**-f1** *CodeSet*:*Font*] [**-p** *Out*] [**-F** *Hfont*] [**-F0** *CodeSet*:*Font*] [**-F1** *CodeSet*:*Font*] [**-L** *Lines*] [**-M** *MediaName*] [**-X** *CodesetName*] [*SpoolerOptions*] [*File* ...]

Description

The **enscript** command reads a text file, converts it to PostScript format, and spools the file for printing on a PostScript printer. You can use this command to specify fonts, headings, limited formatting options, and spooling options.

For example:

```
enscript -daleph bubble.txt
```

prints a copy of the **bubble.txt** file on the printer called aleph, and

```
enscript -2r finder.c
```

prints a two-up landscape listing of the **finder.c** file on the default printer.

The **ENSCRIPT** environment variable can be used to specify defaults. The value of **ENSCRIPT** is parsed as a string of arguments before the arguments that are displayed on the command line. For example:

```
ENSCRIPT='-fTimes-Roman8'
```

sets your default body type size and font to 8-point Times Roman.

Information containing various media sizes for the **psdit** command and the **enscript** command are contained in the file **/usr/lib/ps/MediaSizes**.

The information required for each entry in the **MediaSizes** file can be obtained from the **PostScript Printer Description**, or **PPD**, file that matches the PostScript printer used with TranScript. The **PPD** files are available from Adobe Systems, Incorporated. The measurements extracted from the **PPD** files are expressed in a printer's measure called points. A printer's point is 1/72 of an inch.

Any line in the **MediaSizes** file beginning with an ASCII ***** (asterisk) is ignored when matching media-size names provided on the command line to the **enscript** command and the **psdit** command.

Each entry in the **MediaSizes** file contains either 8 or 9 fields. The first 8 fields are required for all entries. The 9th field is optional. Fields are separated by white space. The fields for each entry are as follows:

Field Name	Description
EntryName	Contains a character string to match against a media name provided with the **-M** flag with the **enscript** command or the **psdit** command.
MediaWidth	Specifies the media width in points.
MediaDepth	Specifies the media depth in points.
ImageableLLX	Specifies the imageable lower left-hand corner x coordinate in points.
ImageableLLY	Specifies the imageable lower left-hand corner y coordinate in points.
ImageableURX	Specifies the imageable upper right-hand corner x coordinate in points.
ImageableURY	Specifies the imageable upper right-hand corner y coordinate in points.
PageRegionName	Specifies the PostScript sequence for the particular printer to identify the size of the imageable area.
PaperTrayName	Specifies the PostScript sequence for the particular printer to select a particular paper/media tray. This field is optional. **Note:** The sequence can be multiple PostScript operators or words for both the PageRegionName field and the PaperTrayName field. To specify such a sequence, use the ASCII " (double quotation character) to delimit the entire sequence.

The following table shows examples of field entries in the **MediaSizes** file:

Name	Field Values
Letter	**Width** 612 **Depth** 792 **llx** 18 **lly** 17 **urx** 597 **ury** 776 **Page- Region- Name** Letter **Paper- Tray- Name** Letter

Name	Field Values
Legal	**Width** 612 **Depth** 1008 **llx** 18 **lly** 17 **urx** 597 **ury** 992 **Page- Region- Name** Legal **Paper- Tray- Name** Legal

PostScript Font Information

The PostScript Fonts for Transcript table shows the fonts available for the enscript command. The Font Name is specified with the **-F** and **-f encscipt** command flags. The alphabetic characters are case-sensitive:

PostScript Fonts for Transcript	
Font Name	**Font Family**
AvantGarde-Book	AvantGarde
AvantGarde-Demi	AvantGarde
AvantGarde-DemiOblique	AvantGarde
AvantGarde-BookOblique	AvantGarde
Bookman-Demi	Bookman
Bookman-DemiItalic	Bookman
Bookman-Light	Bookman
Bookman-LightItalic	Bookman
Courier	Courier
Courier-Bold	Courier
Courier-BoldOblique	Courier
Courier-Oblique	Courier
Garamond-Bold	Garamond
Garamond-BoldItalic	Garamond
Garamond-Light	Garamond
Garamond-LightItalic	Garamond
Helvetica	Helvetica

PostScript Fonts for Transcript *(continued)*	
Font Name	**Font Family**
Helvetica-Bold	Helvetica
Helvetica-Oblique	Helvetica
Helvetica-BoldOblique	Helvetica
Helvetica-Narrow	Helvetica
Helvetica-Narrow-Bold	Helvetica
Helvetica-Narrow-BoldOblique	Helvetica
Helvetica-Narrow-Oblique	Helvetica
LubalinGraph-Book	Lubalin
LubalinGraph-BookOblique	Lubalin
LubalinGraph-Demi	Lubalin
LubalinGraph-DemiOblique	Lubalin

Font Name	**Font Family**
Miryam-Iso	Miryam Iso
Miryam-IsoBold	Miryam Iso
Miryam-IsoBoldItalic	Miryam Iso
Miryam-IsoItalic	Miryam Iso
NarkissimIso	Narkissim Iso
NarkissimIso-Bold	Narkissim Iso
NarkissimIso-BoldItalic	Narkissim Iso
NarkissimIso-Italic	Narkissim Iso
NarkissTamIso	Narkiss Tam Iso
NarkissTamIso-Bold	Narkiss Tam Iso
NarkissTamIso-BoldItalic	Narkiss Tam Iso
NarkissTamIso-Italic	Narkiss Tam Iso
NewCenturySchlbk	NewCentury
NewCenturySchlbk-Bold	NewCentury
NewCenturySchlbk-Italic	NewCentury
NewCenturySchlbk-Roman	NewCentury
Optima	Optima
Optima-Bold	Optima
Optima-BoldOblique	Optima
Optima-Oblique	Optima
Palatino-Bold	Palatino
Palatino-BoldItalic	Palatino

Font Name	Font Family
Palatino-Italic	Palatino
Palatino-Roman	Palatino
Rokaa	Rokaa
Rokaa-Bold	Rokaa
Rokaa-BoldItalic	Rokaa
Rokaa-Italic	Rokaa

Font Name	Font Family
Setting	Setting
Setting-Bold	Setting
Setting-BoldItalic	Setting
Setting-Italic	Setting
ShalomIso	ShalomIso Iso
ShalomIso-Bold	ShalomIso Iso
ShalomIso-BoldItalic	ShalomIso Iso
ShalomIso-Italic	ShalomIso Iso
Souvenir-Demi	Souvenir
Souvenir-DemiItalic	Souvenir
Souvenir-Light	Souvenir
Souvenir-LightItalic	Souvenir
Times-Bold	Times
Times-BoldItalic	Times
Times-Italic	Times
Times-Roman	Times
Typing	Typing
Typing-Bold	Typing
Typing-BoldItalic	Typing
Typing-Italic	Typing
Symbol	(none)
ZapfChancery-MediumItalic	Zapf
ZapfDingbats	(none)

Parameters

Item	Description
SpoolerOptions	Provides options for spooling the print file. The following are the *SpoolerOptions* flags: **{-d | -P}*Queue*** Queues the output to the named queue. **-n*Number*** Produces the specified number of copies. The default is 1. **-t*Title*** Sets job title for use on the first banner page. ***File*** Specifies the text file to be converted into PostScript format. If you leave this parameter blank, the **enscript** command reads from standard input.

Flags

Item	Description
-1	Sets in 1 column (the default).
-2	Sets in 2 columns.
-c	Truncates (cuts) lines that are longer than the page width. Normally, long lines are wrapped around to the following line on the page.
-g	Performs no function, but the **-g** flag is still accepted for backwards compatibility.
-k	Enables page prefeed (if the printer supports it). This allows simple documents (such as program listings in a single font) to print somewhat faster by keeping the printer running between pages.
-l	Simulates a line printer printing pages 66 lines long and omitting headers.
-m	Sends mail after the files are printed.
-o	Lists the missing characters if the **enscript** command cannot find characters in a font.
-q	Causes the **enscript** command to not report about what it is doing. The **enscript** command cannot report on pages, destination, omitted characters, and so on. Fatal errors are still reported to the standard error output.
-r	Rotates the output 90 degrees (landscape mode). Use this flag for output that requires a wide page or for program listings when used in conjunction with the **-2** flag. The following example shows one way to get program listings: `enscript -2r File . . .`
-B	Omits page headings.
-G	Prints in gaudy mode, causing page headings, dates, and page numbers to be printed in a flashy style, at some slight performance expense.
-K	Disables page prefeed (the default).
-R	Prints in portrait mode (unrotated), which is the default.

Item	Description
-b*Header*	Sets the string to be used for page headings to the value of the *Header* variable. The default header is constructed from the file name, its last modification date, and a page number.
-f*Font*	Sets the font to be used for the body of each page. The default is Courier10, unless the two-column rotated mode is used, in which case it defaults to Courier7. **Note:** 1. A PostScript font name (such as Times-Roman, Times-BoldItalic, Helvetica, Courier). 2. A point size (1 point = 1/72 inch). Fonts are specified in this fashion: Courier-Bold8 is 8-point Courier Bold; Helvetica12 is 12-point Helvetica.
-f0 *Codeset***:***Font*	Sets the character codeset name, which is written into the PostScript file, and the SBCS font to use for the body of each page. The default is determined by the **/usr/lib/ps/transcript.conf** configuration file for each locale.
-f1 *Codeset***:***Font*	Sets the character codeset name, which is written into the PostScript file, and the MBCS font to use for the body of each page. The default is determined by the **/usr/lib/ps/transcript.conf** configuration file for each locale.
-p*Out*	Causes the PostScript file to be written to the named file rather than being spooled for printing. As a special case, entering the following will send the PostScript file to standard output: `-p -`
-F*Hfont*	Sets the font to be used for page headings. The default is Courier Bold10. **Note:** Font specifications have two parts: • A PostScript font name (such as Times-Roman, Times-BoldItalic, Helvetica, Courier). • A point size (1 point = 1/72 inch). Fonts are specified in this fashion: Courier-Bold8 is 8-point Courier Bold; Helvetica12 is 12-point Helvetica.
-F0 *Codeset***:***Font*	Sets the character codeset name, which is written into the PostScript file, and the SBCS font to use for the header of each page. The default is determined by the **/usr/lib/ps/transcript.conf** configuration file for each locale.
-F1 *Codeset***:***Font*	Sets the character codeset name, which is written into the PostScript file, and the MBCS font to use for the header of each page. The default is determined by the **/usr/lib/ps/transcript.conf** configuration file for each locale.
-L*Lines*	Sets the maximum number of lines to print on a page. The **enscript** command usually computes how many lines to put on a page based on point size. (It might put fewer per page than requested by the **-L** flag.)
-M*MediaName*	Specifies a media name to use to determine the amount of imageable area on the paper. The name provided is matched against entries in the **MediaSizes** file. For instance, `-M legal` would request a legal size of paper as the imageable area. If this flag is not used, the default size is letter size, which is 8.5 inches wide by 11.0 inches deep (21.6 cent. wide by 27.9 cent. deep).

Item	Description
-X*CodesetName*	Specifies the code set for the input data. By default, the input code set is determined by the **nl_langinfo** subroutine. If this flag is used, the codeset is determined by the *CodesetName*.

International Character Support

All characters not found in a font will be replaced with the character ? (question mark). For a complete list of characters that were not found, use the **-o** flag. The **NLSvec** file provides information about character translation.

Environment Variables

Item	Description
ENSCRIPT	Specifies a string of options to be used by the **enscript** command.
LPDEST	Specifies a printer destination. The **-d** spooler option overrides this environment variable.
PSLIBDIR	Provides a path name of a directory to use instead of the **/usr/lib/ps** directory for the **enscript** command prologue and font metric files.
PSTEMPDIR	Provides a path name of temporary directory to use instead of the **/var/tmp** directory of spooled temporary files.
TRANSCRIPT	Provides the absolute path name of a file to use, instead of the **/usr/lib/ps/transcript.conf** configuration file, for MBCS handling.

Files

Item	Description
/usr/lib/ps/*.afm	Contains Adobe Font Metrics (AFM) files.
/usr/lib/ps/font.map	Contains the list of font names with their abbreviations.
/usr/lib/ps/enscript.pro	Contains prologue for **enscript** command files.
/usr/lib/ps/MediaSizes	Contains the default file used for media sizes.

entstat Command

Purpose

Shows ethernet device driver and device statistics.

Syntax

entstat [**-d** **-r** **-t**] *Device_Name*

Description

The **entstat** command displays the statistics gathered by the specified Ethernet device driver. The user can optionally specify that the device-specific statistics be displayed in addition to the device generic statistics. If no flags are specified, only the device generic statistics are displayed.

This command is also invoked when the **netstat** command is run with the **-v** flag. The **netstat** command does not issue any **entstat** command flags.

If an invalid *Device_Name* is specified, the **entstat** command produces an error message stating that it could not connect to the device.

Flags

Item	Description
-d	Displays all the statistics, including the device-specific statistics.
-r	Resets all the statistics back to their initial values. This flag can only be issued by privileged users.
-t	Toggles debug trace in some device drivers.

Parameters

Item	Description
Device_Name	The name of the Ethernet device, for example, **ent0**.

Statistic Fields

> **Note:** Some adapters may not support a specific statistic. The value of non-supported statistic fields is always 0.

The statistic fields displayed in the output of the **entstat** command and their descriptions are:

Title Fields

Item	Description
`Device Type`	Displays the description of the adapter type.
`Hardware Address`	Displays the Ethernet network address currently used by the device.
`Elapsed Time`	Displays the real time period which has elapsed since last time the statistics were reset. Part of the statistics may be reset by the device driver during error recovery when a hardware error is detected. There will be another Elapsed Time displayed in the middle of the output when this situation has occurred in order to reflect the time differences between the statistics.

Transmit Statistics Fields

Item	Description
`Packets`	The number of packets transmitted successfully by the device.
`Bytes`	The number of bytes transmitted successfully by the device.
`Interrupts`	The number of transmit interrupts received by the driver from the adapter.
`Transmit Errors`	The number of output errors encountered on this device. This is a counter for unsuccessful transmissions due to hardware/ network errors.
`Packets Dropped`	The number of packets accepted by the device driver for transmission which were not (for any reason) given to the device.
`Max Packets on S/W Transmit Queue`	The maximum number of outgoing packets ever queued to the software transmit queue.

Item	Description
S/W Transmit Queue Overflow	The number of outgoing packets which have overflowed the software transmit queue.
Current S/W+H/W Transmit Queue Length	The number of pending outgoing packets on either the software transmit queue or the hardware transmit queue.
Broadcast Packets	The number of broadcast packets transmitted without any error.
Multicast Packets	The number of multicast packets transmitted without any error.
No Carrier Sense	The number of unsuccessful transmissions due to the no carrier sense error.
DMA Underrun	The number of unsuccessful transmissions due to the DMA underrun error.
Lost CTS Errors	The number of unsuccessful transmissions due to the loss of the Clear-to-Send signal error.
Max Collision Errors	The number of unsuccessful transmissions due to too many collisions. The number of collisions encountered exceeded the number of retries on the adapter.
Late Collision Errors	The number of unsuccessful transmissions due to the late collision error.
Deferred	The number of outgoing packets deferred during transmission. Deferred means that the adapter had to defer while trying to transmit a frame. This condition occurs if the network is busy when the adapter is ready to transmit. The adapter will only defer the first attempt to send a packet. After that the adapter will transmit the packet without checking. If the network is still busy then a collision will be recorded.
SQE Test	Contains the number of "Signal Quality Error" Tests (i.e. Heartbeat) performed successfully during transmission.
Timeout Errors	The number of unsuccessful transmissions due to adapter reported timeout errors.
Single Collision Count	The number of outgoing packets with single (only one) collision encountered during transmission.
Multiple Collision Count	The number of outgoing packets with multiple (2 - 15) collisions encountered during transmission.
Current HW Transmit Queue Length	The number of outgoing packets which currently exist on the hardware transmit queue.
CRC Errors	The number of incoming packets with the Checksum (FCS) error.
DMA Overrun	The number of incoming packets with the DMA overrun error.
Alignment Errors	The number of incoming packets with the alignment error.
No Resource Errors	The number of incoming packets dropped by the hardware due to the no resource error. This error usually occurs because the receive buffers on the adapter were exhausted. Some adapters may have the size of the receive buffers as a configurable parameter. Check the device configuration attributes (or smit helps) for possible tuning information.

Item	Description
Receive Collision Errors	The number of incoming packets with the collision errors during the reception.
Packet Too Short Errors	The number of incoming packets with the length error indicating that the packet size is less than the Ethernet minimum packet size.
Packet Too Long Errors	The number of incoming packets with the length error indicating that the packet size is bigger than the Ethernet maximum packet size.
Packets Discarded by Adapter	The number of incoming packets dropped by the hardware for any other reasons.
Receiver Start Count	The number of times that the receiver (receive unit) on the adapter has been started.

Receive Statistics Fields

Item	Description
Packets	The number of packets received successfully by the device.
Bytes	The number of bytes received successfully by the device.
Interrupts	The number of receive interrupts received by the driver from the adapter.
Receive Errors	The number of input errors encountered on this device. This is a counter for unsuccessful reception due to hardware/network errors.
Packets Dropped	The number of packets received by the device driver from this device which were not (for any reason) given to a network demuxer.
Bad Packets	The number of bad packets received (i.e. saved) by the device driver.
Broadcast Packets	The number of broadcast packets received without any error.
Multicast Packets	The number of multicast packets received without any error.
CRC Errors	The number of incoming packets with the Checksum (FCS) error.
DMA Overrun	The number of incoming packets with the DMA overrun error.
Alignment Errors	The number of incoming packets with the alignment error.
No Resource Errors	The number of incoming packets dropped by the hardware due to the no resource error.
Receive Collision Errors	The number of incoming packets with the collision errors during the reception.
Packet Too Short Errors	The number of incoming packets with the length error indicating that the packet size is less than the Ethernet minimum packet size.

Item	Description
Packet Too Long Errors	The number of incoming packets with the length error indicating that the packet size is bigger than the Ethernet maximum packet size.
Packets Discarded by Adapter	The number of incoming packets dropped by the hardware for any other reasons.
Receiver Start Count	The number of times that the receiver (receive unit) on the adapter has been started.

General Statistics Fields

Item	Description
No mbuf Errors	The number of times that mbufs were not available to the device driver. This usually occurs during receive operations when the driver must obtain mbuf buffers to process inbound packets. If the mbuf pool for the requested size is empty, the packet will be discarded. The **netstat -m** command can be used to confirm this.
Adapter Reset Count	The number of times that the adapter has been restarted (re-initialized).
Adapter Data Rate	The maximum data rate of the adapter in Mbps (megabits per second).
Driver Flags	The device driver internal status flags that are currently turned on.

Device Specific Statistics Fields

This part of the display may be different for each type of the adapter. It may contain adapter specific information and some extended statistics that were not included in the generic statistics. Some adapters may not have any device specific statistics.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the device generic statistics for **ent0**, enter:

```
entstat ent0
```

This produces the following output:

```
ETHERNET STATISTICS (ent0) :
Device Type: Ethernet High Performance LAN Adapter
Hardware Address: 02:60:8c:2e:d0:1d
Elapsed Time: 0 days 0 hours 8 minutes 41 seconds

Transmit Statistics:     Receive Statistics:
--------------------     -------------------
Packets: 3               Packets: 2
Bytes: 272               Bytes: 146
Interrupts: 3            Interrupts: 2
Transmit Errors: 0       Receive Errors: 0
Packets Dropped: 0       Packets Dropped: 0
Max Packets on S/W       Bad Packets: 0
Transmit Queue:0
S/W Transmit Queue
Overflow: 0
Current S/W+H/W Transmit
```

```
Queue Length: 0

Broadcast Packets: 2     CRC Errors: 0
Multicast Packets: 0     Broadcast Packets: 1
No Carrier Sense: 0      Multicast Packets: 0
DMA Underrun: 0          DMA Overrun: 0
Lost CTS Errors: 0       Alignment Errors: 0
Max Collision Errors: 0  No Resource Errors: 0
Late Collision Errors: 0 Receive Collision Errors: 0
Deferred: 0              Packet Too Short Errors: 0
SQE Test: 0              Packet Too Long Errors: 0
Timeout Errors: 0        Packets Discarded by Adapter: 0
Single Collision         Receiver Start Count: 1
Count: 0
Multiple Collision Count: 0
Current HW Transmit Queue
Length: 0

General Statistics:
-------------------
No mbuf Errors: 0
Adapter Reset Count: 0
Adapter Data Rate: 2000
Driver Flags: Up Broadcast Running Simplex
```

2. To display the Ethernet device generic statistics and the ethernet device-specific statistics for **ent0**, enter:

```
entstat -d ent0
```

This produces the following output:

```
ETHERNET STATISTICS (ent0) :
Device Type: Ethernet High Performance LAN Adapter
Hardware Address: 02:60:8c:2e:d0:1d
Elapsed Time: 0 days 2 hours 6 minutes 30 seconds

Transmit Statistics:      Receive Statistics:
--------------------      -------------------
Packets: 3                Packets: 2
Bytes: 272                Bytes: 146
Interrupts: 3             Interrupts: 2
Transmit Errors: 0        Receive Errors: 0
Packets Dropped: 0        Packets Dropped: 0
Max Packets on S/W        Receiver Start Count: 1
Transmit Queue:0
Bad Packets: 0
S/W Transmit Queue Overflow: 0
Current S/W+H/W Transmit Queue Length: 0

Broadcast Packets: 0      Broadcast Packets: 0
Multicast Packets: 0      Multicast Packets: 0
No Carrier Sense: 0       CRC Errors: 0
DMA Underrun: 0           DMA Overrun: 0
Lost CTS Errors: 0        Alignment Errors: 0
Max Collision Errors: 0   No Resource Errors: 0
Late Collision Errors: 0  Receive Collision Errors: 0
Deferred: 0               Packet Too Short Errors: 0
SQE Test: 0               Packet Too Long Errors: 0
Timeout Errors: 0         Packets Discarded by Adapter: 0
Single Collision Count: 0 Receiver Start Count: 1
Multiple Collision Count: 0
Current HW Transmit Queue Length: 0

General Statistics:
-------------------
No mbuf Errors: 0
Adapter Reset Count: 0
Adapter Data Rate: 2000
Driver Flags: Up Broadcast Running Simplex

Ethernet High Performance LAN Adapter Specific Statistics:
----------------------------------------------------------
Receive Buffer Pool Size: 37
Transmit Buffer Pool Size: 39
In Promiscuous Mode for IP Multicast: No
Packets Uploaded from Adapter: 0
Host End-of-List Encountered: 0
82586 End-of-List Encountered: 0
```

```
Receive DMA Timeouts: 0
Adapter Internal Data: 0x0 0x0 0x0 0x0 0x0
```

env Command

Purpose

Displays the current environment or sets the environment for the execution of a command.

Syntax

To Display Multiple Environment Variables

env [**-i** | **-**] [*Name*=*Value*]... [*Command* [*Argument* ...]]

To Display A Single Environment Variable

env [*Name*]

Description

The **env** command allows you to display your current environment or run a specified command in a changed environment.

If no flags or parameters are specified, the **env** command displays your current environment, showing one *Name*=*Value* pair per line.

Flags

Item	Description
-i	Ignores the inherited environment and invokes the command specified by the *Command* parameter with the environment specified by the *Name*=*Value* parameters.

Parameters

Item	Description
Name=Value	You can run a command in a modified version of the current environment by specifying one or more *Name=Value* parameters. Use the **-i** flag if you wish to replace the entire current environment with the specified *Name =Value* parameters. In either case, environment changes are effective only while the specified command is running.
Command	The *Command* parameter has an optional *Argument* variable. If the specified command is one of the Korn shell special built-in commands, results are unspecified. Korn shell built-in commands are described in the **ksh** command.

Exit Status

If the *Command* parameter is specified, the exit status of the **env** command is the exit status of the command specified in the *Command* parameter. Otherwise, the **env** command exits with one of the following values:

Item	Description
0	The **env** command completed successfully.
1-125	An error occurred in the **env** command.
126	The command specified by the *Command* parameter was found, but could not be invoked.

Item	Description
127	The command specified by the *Command* parameter was not found.

Examples

1. To change the **TZ** environment variable while running the **date** command, type:

   ```
   TZ=MST7MDT date
   ```

   OR

   ```
   env TZ=MST7MDT date
   ```

   Each of these commands displays the time in mountain time and the current date. The two commands shown are equivalent. When the **date** command is finished, the previous value of the **TZ** environment variable takes effect again.
2. To run the **make** command in an environment that consists only of definitions for the **PATH, IDIR,** and **LIBDIR** environment variables, type:

   ```
   env -i PATH=$PATH IDIR=/$HOME/include LIBDIR=/$HOME/lib make
   ```

   You must specify the **PATH** environment variable so that the shell can find the **make** command. When the **make** command is finished, the previous environment takes effect.

Files

Item	Description
/usr/bin/env	Contains the **env** command.

epkg Command

Purpose

Creates interim fix packages that can be installed by the interim fix manager, **emgr**.

Syntax

epkg [**-w** *WorkDirectory*] [**-a** *APARrefFile*] [**-p** *PrerequisiteFile*] [**-d** *DescriptionFile*] [**-e** *interimfixControlFile*] [**-g** *PrerequisiteFile*] [**-l** *LockFile*] [**-S** *SupersedeFile*] [**-u** {**y**|**n**}]] [**-r** {**y**|**n**|o}] [**-s**] [**-T** {**y**|**n**}] [**-X**] [**-v**] *interimfixLabel*

Description

The **epkg** tool can be run in two different modes: *interactive* and *template-based*. The interactive mode prompts you with several questions and constructs the interim fix package based on the answers. The template-based mode uses an interim fix control file that is provided with the answers to questions that are asked in interactive mode. The interim fix package is installed by the interim fix manager, which is started with the **emgr** command.

Interactive mode

The **epkg** command runs in interactive mode by default. The only required parameter is the interim fix label. If you interrupt an **epkg** session, the interim fix control file will be saved. If you start a new session with the same interim fix label, you will be asked whether you want to keep working with the previous interim fix control file. To provide this information before you start the interactive **epkg** session, run **epkg** with the **-u** flag.

The **epkg** command maintains a record of the question order and allows you to navigate between questions by using subcommands. Also, the **epkg** command remembers the previous answer you provided and sets that answer as the default answer. The **epkg** subcommands are described in the Subcommands section.

After you answer all the questions, the **epkg** command verifies the interim fix control file and creates a compressed tar package that can be installed with the **emgr** command.

Using the control file template

You can create interim fix packages noninteractively by using an interim fix control file as a template. The following is an example of a completed interim fix control file:

```
# interim fix control file complete example
ABSTRACT=This is a test of epkg.
PRE_INSTALL=/tmp/pre_install
POST_INSTALL=.
PRE_REMOVE=/tmp/pre_remove
POST_REMOVE=.
REBOOT=yes
PREREQ=.
DESCRIPTION=/tmp/description
EFIX_FILES=2
APARREF=/tmp/aparref
LKU_CAPABLE=no

EFIX_FILE:
        EFIX_FILE_NUM=1
        SHIP_FILE=/home/test/ls
        TARGET_FILE=/usr/bin/ls
        TYPE= 1
        INSTALLER= 1
        ACL= DEFAULT
        AR_MEM=.

EFIX_FILE:
        EFIX_FILE_NUM=2
        SHIP_FILE=/home/test/mystrcat.o
        TARGET_FILE=/usr/ccs/lib/libc.a
        TYPE= 2
        INSTALLER= 1
        ACL= root:system:555
        AR_MEM=strcat.o
```

The interim fix control file values, are as follows:

ABSTRACT
Briefly describes the interim fix package. The abstract is limited to 38 bytes.

PRE_INSTALL
Specifies the location of a script that is run after the installation preview and before any interim fix files are installed. Failure in the PRE_INSTALL script will cause the interim fix package installation to be aborted. This component is optional.

POST_INSTALL
Specifies the location of a script that is run after all interim fix files have been successfully installed. This component is optional.

PRE_REMOVE
Specifies the location of a script that is run after the removal preview and before any interim fix files are removed during a remove operation. This component is optional.

POST_REMOVE
Specifies the location of a script that is run after interim fix files are removed during a remove operation. This component is optional.

REBOOT
Specifies whether a reboot is required for this interim fix. Allowable values are yes or no. If this value is set to yes, the **emgr** command will make changes as necessary to the boot image and issue a message instructing the user to reboot after installation.

PREREQ
Specifies the location of a file that contains **installp** prerequisites. This component is optional.

APARREF
Specifies the location of a file that contains the APAR reference numbers and abstracts associated with this interim fix. Each line of the file contains an APAR reference number, an APAR number, and an APAR abstract. The format of the file is as follows:

```
APAR reference|:|APAR number|:|APAR abstract
```

Not all fields are required to make a valid APARREF file. If a particular field is unknown or not required, simply specify NONE or leave the field blank. Some examples of valid APARREF files follow:

Example 1

```
123456|:|IV12345|:|This is the APAR abstract
789012|:|IV67890|:|This is another APAR abstract
```

Example 2

```
123456|:|NONE|:|NONE
789012|:|NONE
345678
```

Example 3

```
NONE|:|IV12345|:|This is the APAR abstract
```

Example 4

```
NONE
```

If you provide an APAR reference file with the APAR reference numbers, the automatic removal feature by the **installp** command is enabled for the interim fix. The automatic removal by the **installp** command means the capability to automatically remove an interim fix if the fix is present in the Technology Level, Service Pack, or PTF that the **installp** command is applying. If NONE is listed in the APAR reference field, the automatic removal feature is not enabled for the interim fix.

DESCRIPTION
Specifies the location of a file that contains a detailed description of the interim fix package that is being installed.

EFIX_FILES
Specifies the total number of files in the interim fix.

EFIX_FILE_NUM
Specifies the number of the file in the interim fix. Each file in the interim fix must have a unique number, from 1 to 400. The **epkg** command can support a maximum of 400 files per interim fix.

SHIP_FILE
Specifies the location of a file that **epkg** will archive into the interim fix package. You can specify either an absolute path or a relative path to this file.

TARGET_FILE
Specifies the location where the **SHIP_FILE** will be installed. This location is on the system where the interim fix package will be installed. You must specify an absolute path to this file. If this file is part of a registered package, such as an RPM Package Manager (RPM) or **installp** package, you must specify the tracked location.

TYPE
Specifies the type of file that is being installed. The valid choices are as follows:

1
File (standard or executable)

2
Library or archive member

INSTALLER
Specifies the type of installer, if any, that will track the interim fix package. The valid choices are as follows:

1
Currently tracked by **installp**

2
Currently tracked by RPM

3
Currently tracked by **ISMP**

4
Currently tracked by another installer

5
This is a new file that will be tracked by **installp**

6
New file that will be tracked by RPM

7
New file that will be tracked by **ISMP**

8
New file that will be tracked by another installer

9
Not tracked by any installer

ACL
Specifies the access attributes (mode and ownership) for the file. If this attribute is set to **DEFAULT**, the **emgr** command maintains the current permissions of the file to be replaced. However, if the target file is a new file or if the user wants to specify permissions with the **-v** flag, the **ACL** attribute can be entered with the syntax *Owner*:*Group*:*OctalModes*, similar to the following:

```
ACL= root:system:555
```

AR_MEM
Specifies the name of the archive member. This option is only valid if **TYPE**=2. In this case, **SHIP_FILE** is the local location of the archive member that is being shipped, **TARGET_FILE** is the target archive, and **ACL** applies to the archive member. For example, the following value settings would make the local file **myshr.o** the member **shr.o** in the target archive **/usr/ccs/lib/libc.a**:

```
TYPE=2
SHIP_FILE=/home/myshr.o
TARGET_FILE=/usr/ccs/lib/libc.a
AR_MEM=shr.o
```

BUILD_BOOT_IMAGE
Specifies whether the boot image needs to be rebuilt. Allowable values are yes or no. A reboot is required if this field is set to yes. If this field is set to yes and the REB00T field is set to no, **epkg** returns an error.

E2E_PREREQ
Specifies the location of the interim fix prerequisite file in the interim fix control file.

PKGLOCKS
Specifies the local file location of the package lock file in the interim fix control file.

SUPERSEDE
Specifies the local file location of the superseded file in the interim fix control file.

FIXTESTED
Specifies whether this interim fix has been tested. Allowable values are yes or no.

LKU_CAPABLE
Specifies whether this interim fix is compatible with the Live Update operation. This attribute can have a value of yes or no. Ideally, all interim fixes must be marked as Live Update capable. This compatibility is needed to install interim fixes as a group. If an interim fix is not suitable for a Live Update operation, the LKU_CAPABLE attribute is set to the value of no, however most of the interim fixes have this attribute set to the value of yes.

Support for Superseding

The packager can specify a file containing the interim fix label names that are to be superseded when an epkg is installed. This will cause the **emgr** command to remove any interim fix labels that are specified in this file (if they are installed) before installing the interim fix package. Failure to remove an installed superseded interim fix will abort the installation of the interim fix package. The maximum supported number of superseded labels is 32. The packager can specify the supersede file with the **epkg** command in the following ways:

- Specify the file location with the **-S *supersede_file*** flag. For example:

```
epkg -S /tmp/superseded.epkg myefix
```

- The **epkg** command will prompt for the superseded file if the extended options flag (**-v**) is used in interactive mode. For example:

```
Enter the location for the supersede file or "." to skip.
-> /tmp/superseded.epkg
```

- Set the SUPERSEDE attribute to the local file location of the superseded file in the interim fix control file. For example:

```
SUPERSEDE=/tmp/superseded.epkg
```

The format of the superseded file is one interim fix label to be superseded per line. Comments beginning with a *#* sign and leading white space are ignored. For example:

```
# Requisites for efix myefix3
myefix1
myefix2
```

Support for prereqs and xreqs

The packager can specify a file containing the interim fix label names of interim fixes that are requisites to the interim fix package being installed. This will cause the **emgr** command to check if the interim fix label is installed (PREREQ). If the requisite is not installed, the **emgr** command will abort installation of the interim fix package. The user can also specify an XREQ interim fix label. This will cause the **emgr** command *not* to install the interim fix if the named xreq interim fix is installed.

The packager can specify the interim fix prerequisite file with the **epkg** command in the following ways:

- Specify the file location with the **-g *efix_prereq_file*** flag. For example:

```
epkg -g /tmp/efixprereq.epkg myefix
```

- The **epkg** command will prompt for the interim fix prereq file if the extended options flag (**-v**) is used in interactive mode. For example:

```
Enter the location for the efix prerequisite file or "." to skip.
-> /tmp/efixprereq.epkg
```

- Set the E2E_PREREQ attribute to the local file location of the interim fix prerequisite file in the interim fix control file. For example:

```
E2E_PREREQ=/tmp/efixprereq.epkg
```

The format of the interim fix prerequisite file entries is as follows:

```
EfixLabel  RequisiteType: PREREQ/XREQ
```

For example:

```
oldefix1 PREREQ  # Make sure oldefix1 is already installed
oldefix4 XREQ    # Make sure oldefix4 is NOT installed
```

The maximum number of supported interim fix prerequisites is 32.

Support for enabling automatic interim fix removal by installp

The packager can specify an APAR reference file containing APAR reference numbers. An APAR reference number will allow **installp** to map an interim fix back to the APARs for all the Technology Levels where the fix was shipped. If **installp** determines that the interim fix is contained in the Technology Level, Service Pack, or PTF being applied, **installp** will automatically remove the interim fix prior to applying the updates.

Output and Topology

The **emgr -d** flag displays the contents and topology of the interim fix package. The **-d** option will work with the **-v** verbose option. Valid levels of verbosity are 1-3.

Verbosity level 1 (default) will display:

- LABEL
- EFIX FILES
- TARGET LOCATION

Verbosity level 2 will display:

- All level 1 output
- ABSTRACT
- REBOOT
- PRE-REQUISITES
- PRE_INSTALL
- POST_INSTALL
- PRE_REMOVE
- POST_REMOVE
- FILE TYPE

Verbosity level 3 will display:

- All level 2 output
- PACKAGING DATE
- VUID
- SIZE
- ACL
- CKSUM
- PACKAGE
- EFIX DESCRIPTION
- CONTENTS OF INSTALL SCRIPTS (if text files)

For example:

- To get level 1 verbosity output on interim fix package `test.102403.epkg.Z`, type:

```
emgr -d test.102403.epkg.Z
```

- To get level 3 verbosity output on interim fix package `test.102403.epkg.Z`, type:

```
emgr -v3 -d test.102403.epkg.Z
```

Support for Additional Package Locking

The packager can specify a file containing package names that should be locked by the **emgr** command in addition to those that are automatically locked based on file ownership. The packager must specify the name of the package, the installer, and the type of package lock action (ALWAYS/IFINST). The packager can specify the package lock file using the **epkg** command in the following ways:

- Specify the file location with the **-l** ***pkg_locks_file*** flag. For example:

```
epkg -l /tmp/pkglock.epkg myefix
```

- The **epkg** command will prompt for the package locks file if the extended options flag (**-v**) is used. For example:

```
Enter the location for the package locks file or "." to skip.
-> /tmp/pkglock.epkg
```

- Set the PKGLOCKS attribute to the local file location of the package lock file in the interim fix control file. For example:

```
PKGLOCKS=/tmp/pkglock.epkg
```

The format of the package locks file is as follows:

```
PackageName PackageAction PackageType
```

where *PackageName* is the name of the package to be locked and *PackageAction* is one of the following:

Item	Description
ALWAYS	Always attempt to lock this package. Failure to lock the package results in installation failure.
IFINST	Attempt to lock this package only if the package is installed. Failure to lock an *installed* package results in installation failure.

PackageType is installp (default), rpm, ISMP, other.

Note: Only installp locking is supported.

The maximum number of supported package lock entries is 32.

Example:

```
bos.rte.lvm ALWAYS installp
bos.games   IFINST installp
```

In the above example, the **emgr** command will always attempt to lock **bos.rte.lvm** during installation and will unlock it on removal. The **emgr** command will lock **bos.games** if, and only if, it is installed, and will unlock it on removal (if locked).

Support for the bosboot Option

The **epkg** command reboot options include rebooting without rebuilding the boot image.

The user can specify a reboot without bosboot in the following ways:

- The o argument for the **epkg -r** flag indicates that reboot ("only") is required, but the **emgr** command should not call bosboot (that is, rebuild the boot image).

- The reboot prompt in interactive mode indicates the following choices:

```
Select reboot policy for this efix package:
1) Reboot is NOT required.
2) Reboot is required. The boot image will be rebuilt.
3) Reboot is required. The boot image will NOT be rebuilt.
```

- Set the BUILD_BOOT_IMAGE and REBOOT attribute to "yes" or "no" in the interim fix control file. The following REBOOT and BUILD_BOOT_IMAGE options are supported:

Item	Description
REBOOT=no & BUILD_BOOT_IMAGE=no	Reboot is NOT required.
REBOOT=yes & BUILD_BOOT_IMAGE=yes	Reboot is required. The boot image will be rebuilt.
REBOOT=yes & BUILD_BOOT_IMAGE=no	Reboot is required. The boot image will *not* be rebuilt.

Note: `REBOOT=no & BUILD_BOOT_IMAGE=yes` will result in an error from the **epkg** command.

Flags

Item	Description
-a *APARrefFile*	Specifies the file containing APAR reference number(s).
-d *DescriptionFile*	Specifies the file containing the interim fix description.
-e *interimfixControlFile*	Specifies the interim fix control file that controls how the interim fix is constructed.
-g *PrerequisiteFile*	Specifies the location of the interim fix prerequisite file that contains the interim fix label names. These labels are required before an interim fix package is installed.
-l *LockFile*	Specifies the location of the locked file that contains the package names. These packages are locked by the `emgr` command or automatically based on file ownership.
-p *PrerequisiteFile*	Specifies the file containing installp prerequisites.
-r {y|n|o}	Sets the **epkg REBOOT** attribute. This causes the **emgr** command to make changes as necessary to the boot image and issue a message instructing the user to reboot after installation. The y argument specifies that a reboot and a `bosboot` are required. The n argument specifies that a reboot is not required. The o argument indicates that a reboot is required, but `emgr` should not call `bosboot`.
-S *SupersedeFile*	Specifies the location of the interim fix supersede file that contains the interim fix label names. These labels are to be superseded when an `epkg` is installed.
-s	Causes the **epkg** command to skip questions regarding scripts and the prerequisite file.
-T	Specifies whether this interim fix was tested. Allowable values are yes or no. The default is no.
-u {**yes**|**no**}	Specifies whether you will use an existing interim fix control file.
-v	Causes the **epkg** command to ask more questions for extended options. This includes asking you to specify permissions on all interim fix files.
-w *WorkDirectory*	Specifies the alternate work directory that the **epkg** command will use. The default work directory is **$HOME/epkgwork**.

Item	Description
-X	Causes the **emgr** command to automatically expand file systems when the interim fix is installed, if space is required and expansion is possible.

Parameters

interim fixLabel
Specifies a string that uniquely identifies this interim fix package. The maximum length of an interim fix label is 10 bytes.

Note: The interim fix manager requires each interim fix label on the system to be unique.

Subcommands

b!
Returns to the previous question.

s!
Shows the status of the current interim fix control file

q!
Quits without saving the interim fix control file. (Using the Ctrl+C key sequence causes the **epkg** command to ask you whether you want to save the interim fix control file.)

h!
Displays help information for the current question.

Exit Status

0
The **epkg** command operations completed successfully.

>0
An error occurred.

Examples

1. To run the **epkg** command in interactive mode and create an interim fix package with the interim fix label of **myfix**, type:

   ```
   epkg myfix
   ```

2. To create an interim fix package with the interim fix label of **myfix** using an existing interim fix control file named **/tmp/ecfile**, type:

   ```
   epkg -e /tmp/ecfile myfix
   ```

3. To create an interim fix package with the interim fix label of **myfix** and specify prerequisite file **/tmp/prereq**, description **/tmp/description**, and extended options, type:

   ```
   epkg -v -p /tmp/prereq -d /tmp/description myfix
   ```

Files

Item	Description
/usr/sbin/epkg	Contains the **epkg** command.

eqn Command

Purpose

Formats mathematical text for the **troff** command.

Syntax

eqn [**-d** *Delimiter1Delimiter2*] [**-f** *Font*] [**-p** *Number*] [**-s** *Size*] [**-T** *Name*] [**—**] [*File* ... | **-**]

Description

The **eqn** command is a **troff** preprocessor for typesetting mathematical text on a phototypesetter or comparable device. The output of the **eqn** command is generally piped into the **troff** command, as follows:

eqn [*Flag*...] *File*... **| troff** [*Flag*...] **|** [*Typesetter*]

The **eqn** command reads files specified by the *File* parameter. It reads standard input when a - (minus sign) is specified as the last parameter. A line beginning with the **.EQ** macro marks the start of equation text. The end of equation text is marked by a line beginning with the **.EN** macro. These lines are not altered by the **troff** command, so they can be defined in macro packages to provide additional formatting function such as centering and numbering.

Keywords

The following are keywords known to both the **eqn** and **neqn** commands.

```
above     dot       gsize     over      tdefine
back      dotdot    hat       pile      tilde
bar       down      italic    rcol      to
bold      dyad      lcol      right     under
ceiling   fat       left      roman     up
ccol      floor     lineup    rpile     vec
col       font      lpile     size
cpile     from      mark      sqrt
define    fwd       matrix    sub
delim     gfont     ndefine   sup
```

Keywords recognized by the **eqn** command can be set apart with spaces, tabs, new lines, braces, double quotes, tildes, and circumflexes. Use { } (braces) for groupings; anywhere you can use a single character, such as X, you can substitute a complicated construction enclosed in braces. The ~ (tilde) represents a full space in the output, and the ^ (circumflex) represents a half-space.

Produce subscripts and superscripts using the **sub** and **sup** keywords. Produce fractions with the **over**keyword. Produce square roots with the **sqrt** keyword.

Introduce lower and upper limits using the **from** and **to** keywords. Produce delimiters (such as left and right brackets and braces) of the correct height using the **left** and **right** keywords. Legal characters after the **left** and **right** keywords are braces, brackets, bars, **c** and **f** for ceiling and floor, and **“ ”** (double quotes) for nothing at all (which is useful for a right-side-only bracket). A **left** character does not need a matching **right** character, but a **right** character must have a matching **left** character.

Vertical lists (piles) of things are made with the **pile**, **lpile**, **cpile**, and **rpile** keywords. Piles can have arbitrary numbers of elements. The **lpile** keyword left-justifies, the **pile** and **cpile** keywords center (but with different vertical spacing), and the **rpile** keyword right-justifies. Matrices are made with the **matrix** keyword. In addition, there is an **rcol** keyword for a right-justified column.

Diacritical marks are made with the **dot**, **dotdot**, **hat**, **tilde**, **bar**, **vec**, **dyad**, and **under** keywords.

You can change point sizes and fonts with the **size** *Number* (or **size +/-***Number*), **roman**, **italic**, **bold**, and **font** *Number* keywords. You can change point sizes and fonts globally in a document with the **gsize** *Number* and **gfont** *Number* keywords, or with the command-line **-s***Number* and **-f***Number* flags.

Normally, subscripts and superscripts are reduced by three points from the previous size. You can change this with the command-line **-p***Number* flag.

You can line up successive display parameters. Place the **mark** keyword before the desired lineup point in the first equation; place the **lineup** keyword where it is to line up vertically in subsequent equations.

You can define shorthands or redefine existing keywords with the **define** keyword; for example:

```
define  Thing%Replacement%
```

The preceding example defines a new token called *Thing* that is replaced by *Replacement* whenever it appears thereafter. The % (percent sign) can be any character that does not occur in *Replacement*.

Keywords such as **sum**, **int**, **inf**, and shorthands such as **>=**, **!=**, and **->** are recognized. Greek letters are spelled out in the desired case, as in **alpha** or **GAMMA**. Mathematical words such as **sin**, **cos**, and **log** are made Roman automatically. The **troff** command 4-character escapes, such as **\(dd**, which produces the double dagger, can be used anywhere. Strings enclosed in **“ ”** (double quotes) are passed through untouched. This permits keywords to be entered as text, and can always be used to communicate with the **troff** command.

Flags

Item	Description
-d*Delimiter1Delimiter2*	Sets two ASCII characters, *Delimiter1* and *Delimiter2,* as delimiters of the text to be processed by the **eqn** command, in addition to the input enclosed by the **.EQ** and **.EN** macros. The text between these delimiters is treated as input to the **eqn** command. **Note:** Within a file, you can also set delimiters for **eqn** text using the **delim** *Delimiter1Delimiter2*command. They are turned off by the **delim off** command. All text not between **.EQ** and **.EN** macros is passed through unprocessed.
-f*Font*	Changes font in all the **eqn** command processed text to the value specified by the *Font* variable. The *Font* value (a font name or position) must be one or two ASCII characters.
-p*Number*	Reduces subscripts and superscripts the specified number of points in size (the default is 3).
-s*Size*	Changes point size in all the **eqn** command processed text to the value specified by the *Size* variable.
-T*Name*	Prepares the output for the specified printing device. Terminal Names for Phototypesetter or Comparable Devices provides *Name* variables. The default is **ibm3816**.
-	Forces input to be read from standard input.
—	(double dash) Indicates the end of flags.

Files

Item	Description
/usr/share/lib/pub/eqnchar	Contains special character definitions.

errclear Command

Purpose

Deletes entries from the error log.

Syntax

errclear [**-d** *ErrorClassList*] [**-i** *File*] [**-J** *ErrorLabel* [*,Errorlabel*]] | [**-K** *ErrorLabel* [*,Errorlabel*]] [**-l** *SequenceNumber*] [**-m** *Machine*] [**-n** *Node*] [**-N** *ResourceNameList*] [**-R** *ResourceTypeList*] [**-S** *ResourceClassList*] [**-T** *ErrorTypeList*] [**-y** *FileName*] [**-j** *ErrorID* [**,***ErrorID*]] | [**-k** *ErrorID* [**,***ErrorID*]] *Days*

Description

The **errclear** command deletes error-log entries older than the number of days specified by the *Days* parameter. To delete all error-log entries, specify a value of **0** for the *Days* parameter.

If the **-i** flag is not used with the **errclear** command, the error log file cleared by **errclear** is the one specified in the error log configuration database. (To view the information in the error log configuration database, use the **errdemon** command.)

> **Note:** The **errclear** command clears the specified entries, but does not decrease the error log file size.

You can use the System Management Interface Tool (SMIT) **smit errclear** fast path to run this command.

Flags

Item	Description
-d *List*	Deletes error-log entries in the error classes specified by the *List* variable. The *List* variable values can be separated by , (commas), or enclosed in " " (double quotation marks) and separated by , (commas) or space characters. The valid *List* variable values are **H** (hardware), **S** (software), **O** (**errlogger** messages), and **U** (undetermined).
-i *File*	Uses the error-log file specified by the *File* variable. If this flag is not specified, the **errclear** command uses the value from the error-log configuration database.
-j *ErrorID*[,*ErrorID*]	Deletes the error-log entries specified by the *ErrorID* (error identifier) variable. The *ErrorID* variable values can be separated by , (commas), or enclosed in " " (double quotation marks) and separated by , (commas) or space characters.
-J *ErrorLabel*	Deletes the error-log entries specified by the *ErrorLabel* variable. The *ErrorLabel* variable values can be separated by , (commas), or enclosed in " " (double quotation marks) and separated by , (commas) or space characters.
-k *ErrorID*[,*ErrorID*]	Deletes all error-log entries except those specified by the *ErrorID* (error identifier) variable. The *ErrorID* variable values can be separated by , (commas), or enclosed in " " (double quotation marks) and separated by , (commas) or space characters.
-K *ErrorLabel*	Deletes all error-log entries except those specified by the *ErrorLabel* variable. The *ErrorLabel* variable values can be separated by , (commas), or enclosed in " " (double quotation marks) and separated by , (commas) or space characters.

Item	Description
-l *SequenceNumber*	Deletes error-log entries with the specified sequence numbers. The *SequenceNumber* variable values can be separated by , (commas), or enclosed in " " (double quotation marks) and separated by , (commas) or space characters.
-m *Machine*	Deletes error-log entries for the machine specified by the *Machine* variable. The **uname -m** command returns the value of the *Machine* variable.
-n *Node*	Deletes error-log entries for the node specified by the *Node* variable. The **uname -n** command returns the value of the *Node* variable.
-N *List*	Deletes error-log entries for the resource names specified by the *List* variable. The *List* variable is list of names of resources that have detected errors. For software errors, these are the names of resources that have detected errors. For hardware errors, these are names of devices or system components. It does not indicate that the component is faulty or needs replacement. Instead, it is used to determine the appropriate diagnostic modules to be used to analyze the error. The *List* variable values can be separated by , (commas), or enclosed in " " (double quotation marks) and separated by , (commas) or space characters.
-R *List*	Deletes error-log entries for the resource types specified by the *List* variable. For hardware errors, the *List* variable is a device type. For software errors, the value of the *List* variable is **LPP**. The *List* variable values can be separated by , (commas), or enclosed in " " (double quotation marks) and separated by , (commas) or space characters.
-S *List*	Deletes error-log entries for the resource classes specified by the *List* variable. For hardware errors, the *List* variable is a device class. The *List* variable values can be separated by , (commas), or enclosed in " " (double quotation marks) and separated by , (commas) or space characters.
-T *List*	Deletes error-log entries for error types specified by the *List* variable. Valid *List* variable values are: **PERM**, **TEMP**, **PERF**, **PEND**, **INFO**, and **UNKN**. The *List* variable values can be separated by , (commas), or enclosed in " " (double quotation marks) and separated by , (commas) or space characters.
-y *FileName*	Uses the error-record template file specified by the *FileName* variable.

Security

Access Control: Only the root user can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To delete all entries from the error log, enter:

```
errclear 0
```

2. To delete all entries in the error log classified as software errors, enter:

```
errclear -d S 0
```

3. To clear all entries from the alternate error-log file `/var/adm/ras/errlog.alternate`, enter:

```
errclear -i /var/adm/ras/errlog.alternate 0
```

4. To clear all hardware entries from the alternate error-log file `/var/adm/ras/errlog.alternate`, enter:

```
errclear -i /var/adm/ras/errlog.alternate -d H 0
```

Files

Item	Description
/etc/objrepos/SWservAt	Contains the Software Service Aids Attributes object class, which is the error-log configuration database.

errctrl Command

Purpose

Modifies or displays the error-checking attributes of system components. Persistent attribute values can also be specified for components that have not yet been created.

Syntax

errctrl [**-nru**] *ComponentSelector ... subcommand ...*

errctrl -p [**-ru**] *ComponentSelector ... subcommand ...*

errctrl -P [**-ru**] *ComponentSelector ... subcommand ...*

errctrl -x { **-P** | **-p** } [**-ru**] *ComponentSelector ...*

errctrl -q [**-rupP**] {*ComponentSelector ...*}

errctrl {**-h** | **-?**}

errctrl -P {**errcheckon** | **errcheckoff**}

Description

The **errctrl** command modifies or displays the error-checking attribute values of some or all components. Components are selected by name, by alias, or by type or subtype.

The supported value of a *ComponentSelector* is as follows:

-c
componentPatternList

-l
aliasPatternList

-t
typePatternList

Each list consists of one or more patterns separated by quoted spaces or commas. Patterns can contain special characters as described by the **fnmatch** subroutine. The pattern characters question mark (?), asterisk (*), and brackets ([ ]) are supported, but character classes and collation sequences are not allowed inside brackets ([ ]). Specifying **-c all** selects all components, if no other *ComponentSelector* is used.

The **errctrl** command can also be used with the **-p** or **-P** flag to specify persistent attribute customizations. For more information about persistent attribute customizations, see **Persistent Customizations**.

To enable or disable error checking for all components immediately and persistently, specify the **errcheckon** or **errcheckoff** subcommand with the **-P** flag. No other flags or subcommands are allowed with this form of the command. A **bosboot** command is required to make settings persistent across restarts.

The modified attributes depend on the subcommand specified on the command line. Multiple subcommands can be specified in a single invocation. The following subcommands are available:

Item	Description
errcheckon	Turns on error checking.
errcheckoff	Turns off error checking.
errcheckminimal	Sets the error checking level to 1.
errchecknormal	Sets the error checking level to 3.
errcheckdetail	Sets the error checking level to 7.
errchecklevel={0-9}	Sets the error checking level to the specified value.
lowsevdisposition={disp}	Sets the disposition for low-severity errors to the specified value.
medsevdisposition={disp}	Sets the disposition for medium-severity errors to the specified value.

The **disp** error disposition is one of the following values:

- ignore (or 48)
- log (or 64)
- livedump (or 80)
- isolate (or 96)
- sysdump (or 112)

Other subcommands can be recognized by individual components. A subcommand that is not recognized by a component is ignored.

Current attribute values can be displayed with the **-q** flag. If no *ComponentSelector* is used, attribute values are displayed for all components for which error-checking is supported.

Memory overlay detection system for network memory can be enabled by setting the detailed error level for the netmalloc component. Raise the errlevel for the netmalloc component to five or more (default: 3(normal)) to collect complete network memory police buffer information for all network memory allocation and free events. Note that raising the error level to seven (detail) or more also enables network memory overlay detection system. To enable only the **net_malloc_police** option and outstanding memory allocation (OSTD) logging for all network memory allocations and free events, raise the error level to five.

For more information about modifying the errlevel, see the “Examples” on page 1258 . For more information about raising the trace level to collect trace data in the netmalloc component, see the **ctctrl** command.

This command can be used to set the probability (frequency) and values to the following **netmalloc** functions.

- police_frequency
- frag_mask

Probability is the numerator out of 1024 (for example, 10%: 102, 5%: 51, 1%: 10, 0.1%: 1)

Persistent Customizations

The **-p** and **-P** flags allow attribute values to be specified for system components that have not been created yet. Thus, attributes for newly created components can be customized before the components become active. The **-p** flag is used to specify customizations for components that will be created in the future, but before you restart AIX. The **-P** flag is used to specify customizations that will take effect after the next restart. These customizations are added to the **/var/adm/ras/raspertune** file. You must run the **bosboot** command to save these customizations in the boot image and restart AIX for the customizations to take effect.

The *ComponentSelectors* can contain pattern-matching characters. Thus, a persistent customization can apply to more than one component. In addition, multiple customizations can apply to the same component, if different *ComponentSelectors* are used. If conflicting attribute values are specified in multiple customizations, the last customization takes precedence. If a customization already exists for a specified *ComponentSelector*, the new customization replaces the old one.

Multiple *ComponentSelectors* are allowed when persistent customizations are specified, but in all cases, using multiple selectors is equivalent to specifying multiple commands, each with a single component selector. For example, the customization "`errctrl -p -l hdisk0 -l hdisk1 errchecknormal`" is equivalent to the following two customizations:

```
errctrl -p -l hdisk0 errchecknormal
errctrl -p -l hdisk1 errchecknormal
```

Customizations specified with the **-p** or **-P** flag are not deleted after they are used. Therefore, a single customization can affect multiple new components. Persistent customization can be deleted with the **-x** flag. The *ComponentSelector* must be specified identically to the way it was specified when the customization was created. For example, if a customization is created with the *ComponentSelector* `-l hdisk0`, the customization cannot be deleted with the *ComponentSelector* `-l hdisk[0]`, even though both *ComponentSelectors* match the same component alias. When a persistent customization is deleted, no change is made to the attributes of components that were created when the customization was active.

Persistent customizations deleted with the **-x** and **-P** flags will remain in effect unless you run the **bosboot** command and restart AIX. A persistent customization that was created with the **-P** flag can be deleted after the restart by using the **-x** and **-p** flags. In this case, the customization will be active again if you restart AIX.

If you do not know the customizations that have been made but want to restore the default system setting, you can do one of the following:

- In the **/var/adm/ras/raspertune** file, delete the lines relevant to the customizations and run the **bosboot** command to restart AIX.
- Read the **/var/adm/ras/raspertune** file to figure out the appropriate flags and parameters that have been specified. Then use the **-x** flag to delete the customizations as shown in example “6” on page 1258. Run the **bosboot** command and restart AIX.

The **-r** and **-u** flags can be used when specifying persistent customizations. Using one flag specifies a different name space for the specified component selectors. Using both flags at the same time is equivalent to two separate command invocations, each with one of the flags. For example, the persistent customization "`errctrl -p -l hdisk0 -u -r errcheckdetail`" is equivalent to the following two separate customizations:

```
errctrl -p -l hdisk0 -u errcheckdetail
errctrl -p -l hdisk0 -r errcheckdetail
```

The following persistent customizations are all distinct, and can be modified or deleted independently.

```
errctrl -p -l hdisk0 errcheckdetail
errctrl -p -l hdisk0 -r errcheckdetail
errctrl -p -l hdisk0 -u errcheckdetail
```

Recursive-down customizations (specified by the **-r** flag) take precedence over all other customizations, regardless of the order in which they are specified relative to other non-recursive-down customizations.

Persistent customizations can be queried by using the **-q** flag with either the **-P** or **-p** flag. Specifying the **-q** flag with the **-P** flag displays lines from the **/var/adm/ras/raspertune** file. Specifying the **-q** flag with the **-p** and **-r** flags displays the persistent customizations originally specified with the **-r** flag. Without the **-r** flag, the **-q** and **-p** flags display the persistent customizations specified with or without the **-u** flag.

A persistent customization allows multiple subcommands to be specified. If conflicting subcommands are used, the last subcommand is used. For example, the **errchecknormal** and **errcheckdetail** subcommands specify different values for the same error-checking attribute, so the last specified subcommand will be used.

Flags

Item	Description
-c *ComponentList*	Specifies a comma-separated or space-separated list of component names. The **-c all** flag selects all components if it is the only *ComponentSelector*.
-h or **-?**	Displays a usage message.
-l *aliasList*	Specifies a comma-separated or space-separated list of component aliases.
-n	Applies subcommands immediately. This flag is the default if neither the **-p** nor the **-P** flag is used.
-P	Specifies the subcommands that will persist across restarts. You must run the **bosboot** command and restart AIX for these subcommands to be active.
-x	Deletes the persistent customization for the specified components. The *ComponentSelector* must be entered exactly as they were entered when the customization was originally specified.
-p	Specifies persistent subcommands. The specified subcommands will be applied to newly-created components.
-q	Queries the attribute settings of selected components. This flag can also be used with the **-p** or the **-P** flag to display persistent customizations.
-r	Applies the subcommands recursively to all subcomponents of the selected components.
-t *type_subtypeList*	Specifies a space-separated or a comma-separated list of *type* or *type_subtype* names. Valid *type* names include `device`, `filesystem`, `network`, `services`, `storage`, and `ui`. A complete list of *type* and *type_subtype* names is located in the **/usr/include/sys/ras_base.h** header file.
-u	Applies the subcommands recursively to the ancestors of the specified components.

Note: The **-u** and **-r** flags can be used together. Multiple **-c**, **-l** and **-t** flags can be used on the command line.

Exit Status

Item	Description
`0`	Successful completion.
`>0`	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To turn on detailed error checking for all JFS2 userdata components, enter:

   ```
   errctrl -c 'jfs2.filesystem.*.userdata' errcheckdetail
   ```

2. To specify a persistent customization for the userdata component of new JFS2 file systems, enter:

   ```
   errctrl -p -c 'jfs2.filesystem.*.userdata' errcheckminimal
   ```

   The existing userdata components are not affected.

3. To specify a customization that will persist across restarts, enter:

   ```
   errctrl -P -c 'jfs2.filesystem.*.userdata' errcheckminimal
   ```

   If you run the **bosboot** command and restart AIX, minimal error checking will be in effect for all JFS2 userdata components.

4. To set minimal error checking for all current and future JFS2 userdata components., enter:

   ```
   errctrl -npP -c 'jfs2.filesystem.*.userdata' errcheckminimal
   ```

5. To specify multiple persistent attribute values for the ethernet component, enter:

   ```
   errctrl -P -c ethernet errcheckminimal medsevdisposition=80
   ```

6. To delete the customization specified in Example 2, enter:

   ```
   errctrl -p -x -c 'jfs2.filesystem.*.userdata'
   ```

7. To list all persistent, recursive-down attribute customizations, enter:

   ```
   errctrl -q -p -r
   ```

8. To list the current error-checking attribute values for the JFS2 components and its descendants, enter:

   ```
   errctrl -q -c jfs2 -r
   ```

9. To enable memory overlay detection system (MODS) for network memory (raise the error level to detailed level for the netmalloc component), enter:

   ```
   errctrl errcheckdetail -c netmalloc
   ```

   or

   ```
   errctrl errchecklevel=7 -c netmalloc
   ```

   Note: This also enables the **net_malloc_police** option for all network memory allocations and free events.

10. To enable the **net_malloc_police** option for all network memory allocations and free events, raise the netmalloc component error level to five or greater, enter:

    ```
    errctrl errchecklevel=5 -c netmalloc
    ```

    This command also enables outstanding memory allocation (OSTD) logging for network memory.

11. To change the frequency of netmalloc police allocation and free events to 25%, change the probability of police_frequency function to 256, enter:

```
errctrl police_frequency=256 -c netmalloc.police
```

Location

`/usr/sbin/errctrl`

Files

Item	Description
/var/adm/ras/raspertune	File containing persistent attribute customization that will apply after a restart, if you run the **bosboot** command first.

errdead Command

Purpose

Extracts error records from a system dump or live dump.

Syntax

/usr/lib/errdead [**-i** *FileName*] *DumpFile* [*UnixFile*]

Description

The **errdead** command extracts error records from a system dump or live dump containing the internal buffer maintained by the **/dev/error** file. The **errdead** command extracts the error records from the dump file and adds those error records directly to the error log.

The error log daemon need not be running when the **errdead** command is run.

Flag

Item	Description
-i *FileName*	Adds the extracted error records to the error log file specified by the *FileName* variable. If the file does not exist, the **errdead** command creates it. If this flag is not specified, the value from the error log configuration database is used.

Parameters

Item	Description
DumpFile	Specifies the dump image to operate on.
UnixFile	Specifies the UNIX file that is in use when the system dump or live dump is taken. This is not necessary if using **errdead** command on the same system that the dump originated from.

Security

Access Control: Only the root user can run this command.

Example

To capture error log information from a dump image that resides in the `/var/adm/ras/vmcore.0` file, enter:

```
/usr/lib/errdead /var/adm/ras/vmcore.0
```

Error logging information is in the dump image if the **errdemon** daemon was not running when the dump occurred.

File

Item	Description
/etc/objrepos/SWservAt	Contains the software service aids attributes object class; that is, the error log configuration database.

errdemon Daemon

Purpose

Starts error logging daemon (**errdemon**) and writes entries to the error log.

Syntax

errdemon [[**-B** *BufferSize*] [**-d** | **-D**] [**-i** *File*] [**-s** *LogSize*] [**-t** *Time*] [**-m** *MaxDups*] | **-l**] [**-R** *enable* | *disable*]

Description

The error logging daemon reads error records from the **/dev/error** file and creates error log entries in the system error log. Besides writing an entry to the system error log each time an error is logged, the error logging daemon performs error notification as specified in the error notification database. The **/etc/objrepos/errnotify** file is the error notification database. The default system error log is maintained in the **/var/adm/ras/errlog** file. The last error entry is placed in nonvolatile random access memory (NVRAM). During system startup, this last error entry is read from NVRAM and added to the error log when the error logging daemon is started.

The error logging daemon does not create an error log entry for the logged error if the error record template specifies `Log=FALSE`.

If you use the error logging daemon without flags, the system restarts the error logging daemon using the configuration values stored in the error log configuration database. By default, the **errdemon** daemon removes duplicate error log entries when they are logged very rapidly. This is to prevent run away error logging from adversely effecting system performance. The number of duplicate entries can be seen with a detailed error report.

If the PowerHA pureScale® error logging is enabled, error log entries are sent to the PowerHA pureScale logstream, in addition to the local system error log. The PowerHA pureScale error logging status and logstream name are specified with the **errlg_pscale_enabled** and **errlg_pscale_logstream** values of the error log configuration database. The PowerHA pureScale client fileset must be installed on the system and bindings information for the service named **CentralizedLogService** must be setup. The log space and log stream objects specified as the PowerHA pureScale logstream must exist.

Use the **errclear** command to remove entries from the system error log.

Attention: The error logging daemon is normally started during system initialization. Stopping the error logging daemon can cause error data temporarily stored in internal buffers to be overwritten before it can be recorded in the error log file.

Flags

Item	Description
-B *BufferSize*	Uses the number of bytes specified by the *BufferSize* parameter for the error log device driver's in-memory buffer. The specified buffer size is saved in the error log configuration database. If the *BufferSize* parameter is larger than the buffer size currently in use, the in-memory buffer is immediately increased. If the *BufferSize* parameter is smaller than the buffer size currently in use, the new size is put into effect the next time the error logging daemon is started after the system is rebooted. The buffer cannot be made smaller than the hard-coded default of 8KB. If this parameter is not specified, the error logging daemon uses the buffer size from the error log configuration database. The size you specify is rounded up to the next integral multiple of the memory page size (4KB). The memory used for the error log device driver's in-memory buffer is not available for use by other processes. (The buffer is pinned). Be careful not to impact your system's performance by making the buffer excessively large. On the other hand, if you make the buffer too small, the buffer can become full if error entries arrive faster than they can be read from the buffer and put into the log file. When the buffer is full, new entries are discarded until space becomes available in the buffer. When this situation occurs, the error logging daemon creates an error log entry to inform you of the problem. You can correct the problem by enlarging the buffer.
-d	Specifies that duplicate error log entries cannot be removed. The default behavior is to remove duplicates, which is indicated with the **-D** flag.
-D	Specifies that duplicate entries are to be removed. This is the default.
-i *File*	Uses the error log file specified by the *File* variable. The specified file name is saved in the error log configuration database and is immediately put into use.
-l	Displays the values for the error log file name, file size, buffer size, and duplicate handling values from the error log configuration database.
-m *MaxDups*	Specifies the maximum number of duplicate entries allowed before a duplicate error is forced out. The default is 1000. When an error has been duplicated the number of times that is specified in *MaxDups*, a duplicate error is written just as it would be if a unique error was logged. The values allowed for *MaxDups* are 1 to 2147483647.
-R *enable* | *disable*	Restricts the **errpt** command to only the privileged users. If the `-R` option is disabled, the **errpt** command is available to all users. The default value is *disable*. For more information, see the `Security` section of the **errpt** command.
-s *LogSize*	Uses the size specified by the *LogSize* variable for the maximum size of the error log file. The specified log file size limit is saved in the error log configuration database, and it is immediately put into use. If the log file size limit is smaller than the size of the log file currently in use, the error logging daemon renames the current log file by appending **.old** to the file name. The error logging daemon creates a new log file with the specified size limit. Generate a report from the old log file using the **-i** flag of the **errpt** command. If this parameter is not specified, the error logging daemon uses the log file size from the error log configuration database.

Item	Description
-t *Time*	Specifies the approximate time interval (in milliseconds) within which an error is considered a duplicate if it is identical to the previous error. Errors occurring after this time interval are not considered duplicates even if they are identical to the previous error. The default interval is 10000, or 10 seconds. The values allowed for Time are 1 to 2147483647. **Note:** This flag eliminates duplicate entries in the case of an error logger rapidly logging the same error, this usually indicates a loop condition. It is not intended to catch all duplicate errors for which there may be error notification objects. Making this value sufficiently large may compromise error notification by eliminating too many errors. See the **errpt** command for a description of eliminating duplicate errors in an error report.

Examples

1. To start the error-logging daemon, enter:

```
/usr/lib/errdemon
```

2. To view the current maximum error-log size, enter:

```
/usr/lib/errdemon -l
```

3. To change the current maximum error-log size from 1MB to 64KB, enter:

```
/usr/lib/errdemon -s 65536
```

4. To only consider errors that are logged within the last 10 milliseconds to be duplicates, enter

```
/usr/lib/errdemon -t 10
```

Files

Item	Description
/dev/error	Source of error records.
/var/adm/ras/errtmplt	Contains the error template repository.
/usr/lib/errdemon	Contains the **errdemon** daemon.
/etc/objrepos/SWservAt	Contains the software service aids attributes object class; that is, the error log configuration database.

errinstall Command

Purpose

Installs messages in the error logging message sets.

Syntax

errinstall [**-c**] [**-f**] [**-q**] [**-z** *FileName*] *File*

Description

The **errinstall** command is an installation aid that adds or replaces messages in the Error Description, Probable Cause, User Cause, Install Cause, Failure Cause, Recommended Action, and Detailed Data data id message sets of the error log message catalog.

The *File* parameter specifies an input file containing messages to be added or replaced. If you do not specify the *File* parameter or if you specify it as the **-** (minus sign), the **errinstall** command reads from standard input.

Note: Licensed programs and in-house applications must use predefined messages from the error logging message sets. List the predefined messages using the **errmsg -w** command. To add new messages, third-party software vendors should contact IBM Developer Solutions to register new messages. During the development of in-house applications, the **errmsg** command can be used to add new messages, but the new messages must not conflict with the messages added for other in-house applications.

Undo Feature

The **errinstall** command creates an undo file in the current directory named the *File*.**undo** file. (If the **errinstall** command is reading from standard input, the undo file information is written to standard output.) The *File***.undo** file can be used as input to the **errinstall** command to undo the changes the **errinstall** command has just made. To undo changes, run the **errinstall** command with the **-f** flag and specify the *File***.undo** file for the *File* parameter.

Input File (or Standard In) File Format

Two separate lines of information are required to add or replace a single message in the error log message catalog. You can include multiple additions or replacements in a single file. The first line is required to identify the message set to which the message is to be added or replaced. Use the following format:

```
SET MessageSetID
```

where the *MessageSetID* parameter is one of the following single characters:

Item	Description
E	Identifies Error Description
P	Identifies Probable Cause
U	Identifies User Cause
I	Identifies Install Cause
F	Identifies Failure Cause
R	Identifies Recommended Action
D	Identifies Detailed Data

The second line lists the message ID with the message to be added or replaced. At least one line is required, and multiple lines can be included, following a single line that identifies a message set. As described earlier, users should contact their service representative to obtain the message ID, unless it is required for an in-house application only (in which case, use the **errmsg** command to install the error message without a predetermined error message ID).

You must put a space between the message ID and the message text, and enclose the text of the message in double quotes as follows:

```
message ID "message text"
```

In addition to the two required lines of information, you can also include lines of comments. A comment line must have a $ (dollar sign) or an * (asterisk) operator in the first column. The asterisk is the preferred choice.

Note: Messages added to the Error Description, Probable Cause, and Detailed Data ID message sets must not exceed 40 characters in length. Messages added to the User Cause, Install Cause, Failure Cause, and Recommended Action message sets must not exceed 128 characters in length. If messages exceed these lengths, the **errinstall** command displays a warning message, but adds the messages to the codepoint catalogue. These messages will be truncated when displayed by the summary errpt command.

Flags

Item	Description
-c	Checks the input *File* parameter for syntax errors.
-f	Replaces messages having duplicate IDs. When an attempt is made to add a message using a message ID that is already in use, the **-f** flag forces the **errinstall** command to replace the old message text with the new message text. If the **-f** flag is not specified, the old message text is not replaced and a warning message is written to standard error. The **-f** flag is also required to undo a message installation.
-q	Suppresses the creation of an **undo** file.
-z *FileName*	Uses the error logging message catalog specified by the *FileName* parameter.

Security

Access Control: Only the root user can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To install the error log messages for the licensed product lpp, enter:

```
errinstall -f /tmp/lpp.desc
```

2. To undo the changes made to the error log message catalog by the above example of the **errinstall** command, enter:

```
errinstall -f /tmp/lpp.desc.undo
```

3. To install an error message in the Probable Cause message set, enter:

```
errinstall

* Add a probable cause for widget failure:
SET P
E100 "widget adapter"
```

4. To replace a message with a duplicate ID in the Probable Cause message set, enter:

```
errinstall -f

* Replace the message associated with ID E100 in the
* Recommended Action message set
SET R
E100 "Replace disk drive"
```

5. If you name your input file **in_file** and then want to use it to install new error messages, enter:

```
errinstall in_file
```

6. To overwrite existing error messages in message sets, use the previously defined ID numbers in your **in_file**, and specify the **-f** flag with the **errinstall** command as follows:

```
errinstall -f in_file
```

7. The following example illustrates sample contents of an input file to be installed.

```
*
* Add these error messages to the Detailed Data message set:
*
SET D
8105 "Logical channel number"
8106 "Timer reference stamp"
*
* Add these error messages to the Probable Cause message set:
*
SET P
E861 "Bad memory card"
E865 "Unexpected System Halt"
E876 "Fiber Optic Cable"
*
* Add this message to the Recommended Action message set:
*
SET R
E850 "Install updated driver code"
```

Files

Item	Description
/usr/lib/nls/msg/$LANG/codepoint.cat	Contains the error log message catalog. In the United States, the value of the **$LANG** environment variable is **En_US**.

errlogger Command

Purpose

Logs an operator message.

Syntax

errlogger *Message*

Description

The **errlogger** command creates an operator error log entry that contains an operator message up to 1024 bytes in length.

Security

Access Control: Only the root user can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To create an operator message for system drive reconfiguration, enter:

```
errlogger system drive reconfigured
```

errmsg Command

Purpose

Adds a message to the error log message catalog.

Syntax

errmsg [**-c**] [**-z** *FileName*] [**-w** *Set_List* | *File*]

Description

The **errmsg** command updates and displays the error-log message catalog containing the Error Description, Probable Cause, User Cause, Install Cause, Failure Cause, Recommended Action, and Detailed Data ID message sets.

The message sets to which messages are to be added or deleted are listed in the input *File* parameter as follows:

Item	Description
***** or **$**	Comment lines must have an * (asterisk) or $ (dollar sign) comment operator in the first column. The * is the preferred choice.
+	Messages to be added must be preceded by a + (plus sign).
-	Messages to be deleted must be preceded by a - (minus sign).
SET	Message set ID.
"Message Text"	Message text must be enclosed in double quotation marks.
Message ID	Message ID of the message to be deleted.

Messages added to the Error Description, Probable Cause, and Detailed Data ID message sets must not exceed 40 characters in length. Messages added to the User Cause, Install Cause, Failure Cause, and Recommended Action message sets must not exceed 128 characters in length. A maximum of 2047 user-defined messages can be added to each message set.

The **errmsg** command is used by application developers to create new messages used in the Error Record Templates Repository. An existing message should always be used, if possible.

If no flags are specified on the command line, the default operation is an update. Updates are specified in the input *File* parameter. If the input *File* parameter is not specified or if a **-** (minus sign) is specified instead of the *File* parameter, the **errmsg** command reads from standard input. For each message that is added, the **errmsg** command assigns an identifier. In addition to adding the message to the message catalog, the **errmsg** command writes the identifier and message text to the *File***.out** file. The *File***.out** file is also created when deletions are made from the message catalog. If the **errmsg** command is reading from standard input, the identifier and message text are written to standard output.

Flags

Item	Description
-c	Checks the input file for syntax errors.

Item	Description
-w *Set_List*	Displays the error log message sets specified by the *Set_List* variables. This option displays the messages contained in the Error Log message sets and their identifiers. Output is written to standard output. The *Set_List* variables can be separated by commas or enclosed in double-quotation marks and separated by commas or blanks. The *Set_List* variables are the message set IDs or, if the value of the *Set_List* variable **all** is specified, the contents of all of the Error Log message sets are displayed. The valid values of the *Set_List* variables are: **all** Displays all message sets **D** Displays Detailed Data ID message set **E** Displays Error Description message set **F** Displays Failure Cause message set **I** Displays Install Cause message set **P** Displays Probable Cause message set **R** Displays Recommended Action message set **U** Displays User Cause message set
-z *Filename*	Uses the error-logging message catalog specified by the *Filename* variable.

Security

Access Control: Only the root user can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To delete messages from the Probable Cause message set, enter:

```
errmsg
* Delete messages FF1A, FF1B, and FF1C from the Probable Cause
* message set
SET P
- FF1A
- FF1B
- FF1C
```

2. To add a message to the Probable Cause message set for the Widget Failure error, enter:

```
errmsg
* Add a Probable Cause for Widget Failure
SET P
+ "WIDGET ADAPTER"
```

File

Item	Description
/usr/lib/nls/msg/$LANG/codepoint.cat	Contains the error log message catalog. In the United States, the value of **$LANG** is **En_US**.

errpt Command

Purpose

Generates a report of logged errors.

Syntax

To Process a Report from the Error Log

errpt [**-@** *wpar_name*] [**-a**] [**-A**] [**-c**] [**-d** *ErrorClassList*] [**-D**] [**-e** *EndDate*] [**-g**] [**-i** *File*] [**-I** *File*] [**-j** *ErrorID* [**,***ErrorID*]] | [**-k** *ErrorID* [**,***ErrorID*]] [**-J** *ErrorLabel* [**,***ErrorLabel*]] | [**-K** *ErrorLabel* [**,***ErrorLabel*]] [**-l** *SequenceNumber*] [**-m** *Machine*] [**-n** *Node*] [**-s** *StartDate*] [**-F** *FlagList*] [**-N** *ResourceNameList*] [**-P**] [**-R** *ResourceTypeList*] [**-S** *ResourceClassList*] [**-T** *ErrorTypeList*] [**-y** *File*] [**-z** *File*]

To Process a Report from the Error Record Template Repository

errpt [**-a**] [**-A**] [**-I** *File*] [**-t**] [**-d** *ErrorClassList*] [**-j** *ErrorID* [**,***ErrorID*]] | [**-k** *ErrorID* [**,***ErrorID*]] [**-J** *ErrorLabel* [**,***ErrorLabel*]] | [**-K** *ErrorLabel* [**,***ErrorLabel*]] [**-F** *FlagList*] [**-P**] [**-T** *ErrorTypeList*] [**-y** *File*] [**-z** *File*]

Description

The **errpt** command generates an error report from entries in an error log. It includes flags for selecting errors that match specific criteria. By using the default condition, you can display error log entries in the reverse order they occurred and were recorded. By using the **-c** (concurrent) flag, you can display errors as they occur. If the **-i** flag is not used with the **errpt** command, the error log file processed by **errpt** is the one specified in the error log configuration database. (To view the information in the error log configuration database, use the **errdemon** command.)

The default summary report contains one line of data for each error. You can use flags to generate reports with different formats.

> **Note:** The **errpt** command does not perform error log analysis; for analysis, use the **diag** command. When error log analysis is performed, however, diagnostics may add diagnostic information back into the error log. Such information is shown following the detailed data of the corresponding error log entry.

You can use the System Management Interface Tool (SMIT) **smit errpt** fast path to run this command.

Flags

-@*wpar_name*
Selects the error entries for the specified WPAR name.

-a
Displays information about errors in the error log file in detailed format. If used in conjunction with the **-t** flag, all the information from the template file is displayed.

-A
Displays a shortened version of the detailed report produced by the **-a** flag. The **-A** flag is not valid with the **-a**, **-g**, or **-t** flags. The items reported are the label, date and time, type, resource name, description, and detail data. The example output of this flag is in the following format:

```
LABEL:           STOK_RCVRY_EXIT
Date/Time:       Tue Dec 14 15:25:33
Type:            TEMP
Resource Name:   tok0
Description
PROBLEM RESOLVED
Detail Data
FILE NAME
line: 273 file: stok_wdt.c
SENSE DATA
0000 0000 0000 0000 0000 0000
DEVICE ADDRESS
0004 AC62 25F1
```

-c
Formats and displays each of the error entries concurrently, that is, at the time they are logged. The existing entries in the log file are displayed in the order in which they were logged.

-d *ErrorClassList*
Limits the error report to certain types of error records specified by the valid *ErrorClassList* variable: **H** (hardware), **S** (software), **0** (**errlogger** command messages), and **U** (undetermined). The error records in the *ErrorClassList* variable can be separated by a **,** (comma), or enclosed in " " (double quotation marks) and separated by a **,** (comma), or a space character.

-D
Consolidates duplicate errors. The detailed error report, obtained with the **-a** flag, reports the number, and first and last times of the duplicates. See Error Logging Overview in *General Programming Concepts: Writing and Debugging Programs*.

Note: The **-D** flag is not valid with the **-c**, **-g**, **-l**, **-t**, and **-P** flags.

-e *EndDate*
Specifies all records posted prior to and including the *EndDate* variable, where the *EndDate* variable has the form *mmddhhmmyy* (month, day, hour, minute, and year).

-g
Displays the ASCII representation of unformatted error-log entries. The output of this flag is in the following format:

el_sequence
Error-log stamp number

el_label
Error label

el_timestamp
Error-log entry time stamp

el_crcid
Unique cyclic-redundancy-check (CRC) error identifier

el_machineid
Machine ID variable

el_nodeid
Node ID variable

el_class
Error class

el_type
Error type

el_resource
Resource name

el_rclass
Resource class

el_rtype
Resource type

el_vpd_ibm
IBM vital product data (VPD)

el_vpd_user
User VPD

el_in
Location code of a device

el_connwhere
Hardware-connection ID (location on a specific device, such as slot number)

et_label
Error label

et_class
Error class

et_type
Error type

et_desc
Error description

et_probcauses
Probable causes

et_usercauses
User causes

et_useraction
User actions

et_instcauses
Installation causes

et_instaction
Installation actions

et_failcauses
Failure causes

et_failaction
Failure actions

et_detail_length
Detail-data field length

et_detail_descid
Detail-data identifiers

et_detail_encode
Description of detail-data input format

et_logflg
Log flag

et_alertflg
Alertable error flag

et_reportflg
Error report flag

el_detail_length
Detail-data input length

el_detail_data
Detail-data input

-F ***FlagList***
Selects error-record templates according to the value of the `Alert`, `Log`, or `Report` field of the template. The *FlagList* variable can be separated by a **,** (comma), or enclosed in " " (double quotation marks) and separated by a **,** (comma), or a space character. The **-F** flag is used with the **-t** flag only.

Valid values of the *FlagList* variable include:

alert=0
Selects error-record templates with the `Alert` field set to False.

alert=1
Selects error-record templates with the `Alert` field set to True.

log=0
Selects error-record templates with the `Log` field set to False.

log=1
Selects error-record templates with the `Log` field set to True.

report=0
Selects error-record templates with the `Report` field set to False.

report=1
Selects error-record templates with the `Report` field set to True.

-i *File*
Uses the error log file specified by the *File* variable. If this flag is not specified, the value from the error log configuration database is used.

-I *File*
Uses the diagnostic log file specified by *File*. If this flag is not specified, the default pathname, **/var/adm/ras/diag_log**, is used.

-j *ErrorID*[,*ErrorID*]
Includes only the error-log entries specified by the *ErrorID* (error identifier) variable. The *ErrorID* variables can be separated by a **,** (comma), or enclosed in " " (double quotation marks) and separated by a **,** (comma), or a space character. When combined with the **-t** flag, entries are processed from the error-template repository. (Otherwise entries are processed from the error-log repository.)

-J *ErrorLabel*
Includes the error log entries specified by the *ErrorLabel* variable. The *ErrorLabel* variable values can be separated by commas or enclosed in double-quotation marks and separated by commas or blanks. When combined with the **-t** flag, entries are processed from the error template repository. (Otherwise, entries are processed from the error log repository.)

-k *ErrorID*[,*ErrorID*]
Excludes the error-log entries specified by the *ErrorID* variable. The *ErrorID* variables can be separated by a **,** (comma), or enclosed in " " (double quotation marks) and separated by a **,** (comma), or a space character. When combined with the **-t** flag, entries are processed from the error-template repository. (Otherwise entries are processed from the error-log repository.)

-K *ErrorLabel*
Excludes the error log entries specified by the *ErrorLabel* variable. The *ErrorLabel* variable values can be separated by commas or enclosed in double-quotation marks and separated by commas or blanks. When combined with the **-t** flag, entries are processed from the error template repository. (Otherwise, entries are processed from the error log repository).

-l *SequenceNumber*
Selects a unique error-log entry specified by the *SequenceNumber* variable. This flag is used by methods in the error-notification object class. The *SequenceNumber* variable can be separated by a **,** (comma), or enclosed in " " (double quotation marks) and separated by a **,** (comma), or a space character.

-m *Machine*
Includes error-log entries for the specified *Machine* variable. The **uname -m** command returns the *Machine* variable value.

-n *Node*
Includes error-log entries for the specified *Node* variable. The **uname -n** command returns the *Node* variable value.

-N *ResourceNameList*
Generates a report of resource names specified by the *ResourceNameList* variable. The *ResourceNameList* variable is a list of names of resources that have detected errors. For software errors, the *ResourceNameList* variable lists the names of resources that have detected errors. For hardware errors, it lists names of devices or system components. It does not indicate that the component is faulty or needs replacement. Instead, it is used to determine the appropriate diagnostic modules to be used to analyze the error.

The names of the *ResourceNameList* variable can be separated by a **,** (comma), or enclosed in " " (double quotation marks) and separated by a **,** (comma), or a space character.

-P
Shows only errors which are duplicates of the previous error. The **-P** flag applies only to duplicate errors generated by the error log device driver. These errors are duplicates that occurred within the approximate time interval specified by the **errlg_duptime** error logging attribute controlled by the **errdemon** daemon **-t** flag. The **-P** flag is invalid with the **-D** flag.

-R *ResourceTypeList*
Generates a report of resource types specified by the *ResourceTypeList* variable. For hardware errors, the *ResourceTypeList* variable is a device type. For software errors, it is the **LPP** value. The items in the *ResourceTypeList* variable can be each separated by a **,** (comma), or enclosed in " " (double quotation marks) and separated by a **,** (comma), or a space character.

-s *StartDate*
Specifies all records posted on and after the *StartDate* variable, where the *StartDate* variable has the format *mmddhhmmyy* (month, day, hour, minute, and year).

-S *ResourceClassList*
Generates a report of resource classes specified by the *ResourceClassList* variable. For hardware errors, the *ResourceClassList* variable is a device class. The resource classes must be each separated by a **,** (comma), or enclosed in " " (double quotation marks) and separated by a **,** (comma), or a space character.

-t
Processes the error-record template repository instead of the error log. The **-t** flag can be used to view error-record templates in report form.

-T *ErrorTypeList*
Limits the error report to error types specified by the valid *ErrorTypeList* variables: **INFO**, **PEND**, **PERF**, **PERM**, **TEMP**, and **UNKN**. The error types can be each separated by a **,** (comma), or enclosed in " " (double quotation marks) and separated by a **,** or a space character.

-y *File*
Uses the error record template file specified by the *File* variable. When combined with the **-t** flag, entries are processed from the specified error template repository. (Otherwise, entries are processed from the error log repository, using the specified error template repository.)

-z *File*
Uses the error logging message catalog specified by the *File* variable. When combined with the **-t** flag, entries are processed from the error template repository. (Otherwise, entries are processed from the error log repository.)

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

To run the **errpt** command, you must have the following additional authorizations, if **errpt** restriction is enabled in the **errdemon** command:

aix.ras.error.errpt
Required to run the **errpt** command.

Note: By default, the root and system group users are privileged users.

Examples

1. To display a complete summary report, enter:

   ```
   errpt
   ```

2. To display a complete detailed report, enter:

   ```
   errpt  -a
   ```

3. To display a detailed report of all errors logged for the error identifier E19E094F, enter:

   ```
   errpt  -a  -j E19E094F
   ```

4. To display a detailed report of all errors logged in the past 24 hours, enter:

   ```
   errpt  -a  -s mmddhhmmyy
   ```

   where the mmddhhmmyy string equals the current month, day, hour, minute, and year, minus 24 hours.

5. To list error-record templates for which logging is turned off for any error-log entries, enter:

   ```
   errpt  -t  -F log=0
   ```

6. To view all entries from the alternate error-log file /var/adm/ras/errlog.alternate, enter:

   ```
   errpt  -i /var/adm/ras/errlog.alternate
   ```

7. To view all hardware entries from the alternate error-log file /var/adm/ras/errlog.alternate, enter:

   ```
   errpt  -i /var/adm/ras/errlog.alternate -d H
   ```

8. To display a detailed report of all errors logged for the error label ERRLOG_ON, enter:

   ```
   errpt  -a  -J ERRLOG_ON
   ```

9. To display a detailed report of all errors and group duplicate errors, enter:

   ```
   errpt -aD
   ```

10. To display a detailed report of all errors logged for the error labels DISK_ERR1 and DISK_ERR2 during the month of August, enter:

    ```
    errpt -a -J DISK_ERR1,DISK_ERR2 -s 0801000004 -e 0831235904"
    ```

Files

/etc/objrepos/SWservAt
Contains the software service aids attributes object class; that is, the error log configuration database.

errstop Command

Purpose

Terminates the error logging daemon.

Syntax

errstop

Description

Attention: Running the **errstop** command disables diagnostic and recovery functions. Normally the **errdemon** command is started automatically during system initialization and stopped during system shutdown. The error log should never be stopped during normal operations. The **errstop** command should only be used during special circumstances when it is absolutely required and the consequences are clearly understood.

The **errstop** command stops the error logging daemon initiated by the **errdemon** command.

Security

Access Control: Only a root user can run this command.

Examples

To terminate the **errdemon** daemon, enter:

```
/usr/lib/errstop
```

errupdate Command

Purpose

Updates the Error Record Template Repository.

Syntax

errupdate [**-c**] [**-f**] [**-h**] [**-n**] [**-p**] [**-q**] [**-y** *FileName*] [*File*]

Description

The **errupdate** command adds or deletes entries in the Error Record Template Repository, or modifies the log, report, or alert characteristics of existing entries. The **errupdate** command reads from the specified *File* parameter. If the *File* parameter is not specified, the **errupdate** command reads from standard input and writes to standard output.

Each entry to be added, deleted, or modified must be preceded by an operator. The valid operators are:

Item	Description
+	Adds an entry (add operator).
-	Deletes an entry (delete operator).
=	Modifies the log, report, or alert characteristics of an entry.

Entries in the input file must be separated by a blank line.

Comments in the input file can be placed between templates and are indicated by an * (asterisk) in the first column.

If X/Open Portability Guide Issue 4 messages are used in error templates, a message catalog must be specified. This can be done with a line of the form:

```
<*!catalog-name>
```

For example

```
*!mycat.cat
```

The catalog specified applies to XPG4 messages found in subsequent templates, until another "*!" catalog specifier is encountered. Also, the "*!" specifier may be overridden on an individual template basis with the "catname" keyword.

Unless a full pathname to the catalog is specified, the normal rules for retrieving a message catalog are followed. For example, in the above example, mycat.cat is assumed to be in **/usr/lib/nls/msg/%L**.

Entries to be added must be defined in a specific format. The general form of the error record template is:

```
Error Record Template

        + LABEL:
                        Comment=
                        Class=
                        Log=
                        Report=
                        Alert=
                        Err_Type=
                        Err_Desc=
                        Prob_Causes=
                        User_Causes=
                        User_Actions=
                        Inst_Causes=
                        Inst_Actions=
                        Fail_Causes=
                        Fail_Actions=
                        Detail_Data= <data_len>, <data_id>,
                        <data_encoding>
```

Additionally, a catalog name for XPG4 messages can be specified with:

```
catname = <catalog>
```

Any template which contains XPG4 messages, the catname keyword, more than eight detail data items will be referred to as an XPG4 template. An XPG4 template is not alertable, and uses a slightly different calculation for the error id.

The error record template fields are described as follows:

Item	Description
`Alert`	Indicates that the error log entry can be processed by products that conform to the SNA Generic Alert Architecture. The `Alert` field can be set to True or False. If this field is omitted from the template, its value will default to False. If the `Alert` field is set to True, the **errupdate** command does not add the template unless the contents of the `Err_Desc` , `Inst_Actions` , `Fail_Cause, Fail_Actions` , and `Detail_Data data_id` fields are values recognized by the SNA Generic Alert Architecture (in publication GA27-3136). If any of the values used are not recognized by the SNA Generic Alert Architecture or the template is an XPG4 template, and the `Alert` field is set to True, the **-p** flag must be specified to add or update the template.
`Class`	Describes whether the error occurred in hardware or software, is an operator message, or is undetermined. One of the following class descriptors must be specified: **H** Indicates the error is a hardware failure. **O** Indicates the error is an operator message. **S** Indicates the error is a software failure. **U** Indicates the error is undetermined.

Item	Description
Comment	Specifies a comment to be included with the **#define** statement that was created for the Error ID message set. The comment must not exceed 40 characters and must be enclosed in double quotation marks. Comments longer than 40 characters are automatically truncated. The **errupdate** command encloses the comment in the C language comment delimiters, /* (slash, asterisk) and */ (asterisk, slash).
Detail_Data	Describes detailed data, such as detecting module names, sense data, or return codes, that are logged with the error when the error occurs. If no detailed data is logged with the error, this field can be left blank or it can display a message from the Detailed Data ID message set by specifying a **data_len** value of zero. The following three values are required for each **Detail_Data** field and must be separated by commas: **data_len** Number of bytes of data to be associated with the **data_id** value. The **data_len** value is interpreted as a decimal value. To specify environment dependent size, use "W". "W" will be treated as 8 bytes if error is logged from a 64-bit environment, otherwise 4 bytes. **Note:** During detail data length calculation, each "W" is treated as 8 bytes long, and it is not case sensitive. **data_id** Identifies a text message from the Detailed Data ID message set “D” to be printed in the error report in front of the detailed data. The value is interpreted as an unsigned hexadecimal up to 4 digits in length. **data_encoding** Describes how detailed data is to be printed in an error report. Valid values are: **ALPHA** The detailed data is a printable ASCII character string. **DEC** The detailed data is the binary representation of an integer value, and the decimal equivalent is to be printed. **LDEC** The detailed data is the binary representation of a 64-bit value, and the decimal equivalent is to be printed. **HEX** The detailed data is to be printed in hexadecimal. Up to 16 Detail_Data entries may be specified per template. The amount of data logged with an error must not exceed ERR_REC_MAX defined in the **/usr/include/sys/err_rec.h** file. Error data that cannot be contained in an error log entry should be saved elsewhere. Detailed data in the error log entry should contain information that can be used to correlate the error data and the error log entry.
Err_Desc	Describes the error that has occurred. An Error Description message identifier must be specified in this field. This value identifies a text message from the Error Description message set “E” to be displayed for an occurrence of the error. The value is interpreted as an unsigned hexadecimal up to 4 digits in length. The field may also specify an XPG4 style message. This is discussed later.

Item	Description
Err_Type	Describes the severity of the error that has occurred. One of the following values must be specified: **PERF** Condition where the performance of the device or component has degraded to below an acceptable level (performance). **PERM** Condition that cannot be recovered from (permanent). **PEND** Condition signifying that the loss of availability of a device or component is imminent (impending). **TEMP** Condition that was recovered from after a number of unsuccessful attempts (temporary). **UNKN** Condition where it is not possible to determine the severity of the error (unknown). **INFO** Condition for informational error log entry.
Fail_Actions	Describes recommended actions for correcting an error that resulted from a failure cause. A list of up to 4 Recommended Action message identifiers separated by commas can be specified. This value identifies a text message from the Recommended Action message set “R” to be displayed for an occurrence of the error. The value is interpreted as an unsigned hexadecimal up to four digits in length. This field must be blank if the Fail_Causes field is blank. The order in which the recommended actions are listed should be determined by the expense of the action and the probability that the action will correct the error. Always list the actions that have little or no cost (or little or no impact) on the system first. List the actions for which the probability of correcting the error is equal or nearly equal next, with the least expensive actions first. List the remaining actions in order of decreasing probability. The field may also specify an XPG4 style message. This is discussed later.
Fail_Causes	Describes failure causes for the error that has occurred. A failure cause is defined as a condition that resulted from the failure of a resource. This field can list up to four Failure Cause message identifiers separated by commas. This value identifies a text message from the Failure Cause messages set “F” to be displayed for an occurrence of the error. The value is interpreted as an unsigned hexadecimal up to 4 digits in length. List the failure causes in order of decreasing probability. This field can be left blank if it does not apply to the error that has occurred. If this field is blank, either the User_Causes or the Inst_Causes field must not be blank. The field may also specify an XPG4 style message. This is discussed later.

Item	Description
Inst_Actions	Describes recommended actions for correcting an install caused error. This field can list of up to 4 Recommended Action message identifiers separated by commas. This value identifies a text message from the Recommended Action message set "R" to be displayed for an occurrence of the error. The value is interpreted as an unsigned hexadecimal up to four digits in length. This field must be blank if the Inst_Causes field was left blank. The order in which the recommended actions are listed is determined by the expense of the action and the probability that the action will correct the error. The actions that have little or no cost or little or no impact on the system should always be listed first. Actions for which the probability of correcting the error are equal or nearly equal should be listed next, with the least expensive actions first. The remaining actions should be listed in order of decreasing probability. The field may also specify an XPG4 style message. This is discussed later.
Inst_Causes	Describes install causes for the error that has occurred. An install cause is defined to be a condition that resulted from the initial installation or setup of a resource. A list of up to 4 Install Cause message identifiers separated by commas can be specified. This value identifies a text message from the Install Cause message set "I" to be displayed for an occurrence of the error. The value is interpreted as an unsigned hexadecimal up to four digits in length. Install causes should be listed in order of decreasing probability. This field can be left blank if it is not applicable to the error that has occurred. If this field is left blank, the User_Causes or the Fail_Causes field must be non-blank. The field may also specify an XPG4 style message. This is discussed later.
LABEL	Specifies a unique label of up to 19 characters that must be provided for each error logging template. A string containing " #define #ERRID_label Error_ID ", where the **Error_ID** value is the unique ID assigned to the Error Record Template is written to standard output if the **-h** flag was specified at the command line. **Note:** If the LABEL field exceeds 19 characters, the first 19 characters are accepted.
Log	Specifies whether an error log entry should be created for this error when it occurs. The log field can be set to True or False. If this field is omitted from the template, its value will default to True. When this field is set to False, the Report and Alert fields are ignored.
Prob_Causes	Describes 1 or more probable causes for the error that has occurred. A list of up to 4 Probable Cause message identifiers separated by commas can be specified. This value identifies a text message from the Probable Cause message set "P" to be displayed for an occurrence of the error. The value is interpreted as an unsigned hexadecimal up to 4 digits in length. Probable causes should be listed in order of decreasing probability. At least one probable cause is required. The field may also specify an XPG4 style message. This is discussed later.
Report	Specifies whether logged occurrences of this error should be reported when an error report is printed. The Report field can be set to True or False. If this field is omitted from the template, its value will default to True.

Item	Description
User_Actions	Describes recommended actions for correcting a user-caused error. A list of up to 4 Recommended Action message identifiers separated by commas can be specified. This value identifies a text message from the Recommended Action message set “R” to be displayed for an occurrence of the error. The value is interpreted as an unsigned hexadecimal up to 4 digits in length. This field must be left blank if the User_Causes field was left blank. The order in which the recommended actions are listed is determined by the expense of the error and the probability that the action will correct the error. The actions that have little or no cost, or little or no impact on the system should always be listed first. Actions for which the probability of correcting the error are equal or nearly equal should be listed next, with the least expensive actions first. The remaining actions should be listed in order of decreasing probability. The field may also specify an XPG4 style message. This is discussed later.
User_Causes	Describes user causes for the error that has occurred. A user cause is defined as a condition that can be corrected without contacting a service organization. A list of up to four User Cause message identifiers separated by commas can be specified. This value identifies a text message from the User Cause message set “U” to be displayed for an occurrence of the error. The value is interpreted as an unsigned hexadecimal up to four digits in length. User causes should be listed in order of decreasing probability. This field can be left blank if it is not applicable to the error that has occurred. If this field is left blank, the Inst_Causes or the Fail_Causes field must be non-blank. The field may also specify an XPG4 style message. This is discussed later.

The catname is used to specify a message catalog to be used for retrieving XPG4 messages for the current template. This will override a catalog specified with a previous "*!" catalog specifier. Any template containing XPG4 messages must have a catalog specified either with catname or "*!". The catalog name must be enclosed in quotes. Unless a full pathname to the catalog is specified, the normal rules for retrieving a message catalog are followed.

For example, if

```
catname = "mycat.cat"
```

is specified, mycat.cat is assumed to be in **/usr/lib/nls/msg/%L**.

The Error Description, Probable Cause, User Cause, Install Cause, Failure Cause, Recommended Actions, and Detailed Data ID messages must be either an error message identifier maintained in the error log message catalog, or an XPG4 message.

An error message identifier consists of up to 4 hexadecimal digits, without any leading "0x". For example, 1234 or ABCD. The **errmsg -w** command can be used to print these messages along with their identifiers. The **errmsg** command can be used to add new messages.

An XPG4 message is specified using the form

```
{<set>, <number>, <"default text">}
```

The set, number, and default text are all required. Symbolic message references are not supported. Also, templates which contain XPG4 messages are not alertable.

A message catalog must be specified for XPG4 messages. This is done with either the "*!" catalog specifier, or the catname keyword.

Error logging does not support all the features of normal error messaging. Strings used in error log templates must conform to some restrictions.

- Variable substitution is not supported. For example, the strings may not be used as format specifiers to print values. The strings may only contain the formatting characters "\t" and "\n".

- The default text strings may not be longer than 1 kb, 1024 bytes.
- It must be noted that the error description is printed in a 40 character area on the non-detailed reports. No string formatting is done for these reports, and only the first 40 characters will be printed.
- The strings should not contain a trailing new line. This is supplied by errpt.

For each entry added, the errupdate command assigns a unique Error ID that is written to the header file specified by *File*.h (where the *File* parameter is the name of the errupdate command input file). If the errupdate command is reading from standard input, the #`define` statement is written to standard output. The values supplied for the `Class` , `Err_Desc` , `Err_Type` , `Fail_Actions` , `Fail_Causes` , `Inst_Actions` , `Inst_Causes` , `Prob_Causes` , `User_Actions` , `User_Causes` fields, and the `Detail_Data` . `data_id` value, are used to calculate the unique Error ID for that error. For XPG4 templates, the Label is also included in the calculation.

The contents of the `Log` , `Report` , and `Alert` fields are not included in the calculation of the unique Error ID; therefore, the log, report, and alert characteristics of a particular error can be modified at any time in the error entry definition stored in the Error Record Template Repository using the **errupdate** command. Also note that the data_len and data_encode portions of the detail data field are not used.

The **errupdate** command also creates an undo file in the current directory named *File*.**undo**. If the **errupdate** command is reading from standard input, the **undo** file is written to **errids.undo** file. The **undo** file contains inputs to the **errupdate** command to undo changes the **errupdate** command has made.

The **errpt -t** command can be used to view the contents of the Error Record Template Repository. The templates are processed and printed as they would appear in an actual error report.

> **Attention:** If you change the error templates be aware that these templates may be changed by a subsequent update. You should keep a record of all changes made and re-apply the changes when your system is updated. This is usually only necessary after a major system update such as moving to a new level of the operating system. Also, such a record allows you to change your templates if you re-install. The easiest way to keep such a record is to always make your template modifications from one errupdate source file.

Flags

Item	Description
-c	Checks the input file for syntax errors.
-f	Forces all templates to be updated, including any templates with error ids identical to ones in the input templates
-h	Creates a #`define` statement for each Error ID assigned to an error template. If a file name was supplied on the command line, the header file name will be that supplied file name appended with **.h**. Otherwise, the #`define` statements are written to standard output.
-n	Suppresses the addition of the error record template to the Error Record Template Repository.
-p	Adds or updates a template with the `Alert` field set to True that contains Error Description, Probable Cause, User Cause, User Action, Install Cause, Install Action, Failure Cause, Fail Action, or Detailed Data data id values that are not recognized by the SNA Generic Alert Architecture (in publication GA27-3136). The **errupdate** command will not let you add a template with these characteristics unless you specify this flag.
-q	Suppresses the creation of an undo file.
-y *FileName*	Uses the error record template file specified by the *FileName* parameter.

Security

Access Control: None, but you must have write authority to a template file you're changing, /var/adm/ras/errtmplt by default.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To add an entry, define the entry in the input file in the following manner:

```
+ CDROM_ERR22:
        Comment=        “Temporary CDROM read error”
        Class=  H
        Log=            True
        Report= True
        Alert=          False
        Err_Type=       TEMP
        Err_Desc=       E801
        Prob_Causes=    5004
        Fail_Causes=    E800, 6312
        Fail_Actions=   1601, 0000
        Detail_Data=    120, 11, HEX
        Detail_Data=    4, 8058, DEC
        Detail_Data=    4, 8059, DEC
```

   To enter the data,

```
errupdate <input file>
```

2. To modify the log, report, and alert characteristics of entry 99999999 , specify the modify operator = (equal sign) followed by the unique Error ID, and the new characteristics for the entry to be modified:

```
errupdate
 =99999999:
 Report = False
 Log = True
```

3. To delete entry 99999999 from the Error Record Template Repository, specify the delete operator - (minus sign) followed by the unique Error ID of the entry to be deleted:

```
errupdate
 -99999999:
```

4. To override the XPG4 message catalog specified for this input stream with "*!", use the "catname" keyword.

```
*!mycat.cat
```

   `* mycat.cat` is used for all XPG4 messages from now on.

   * except for this one:

```
+ CDROM_ERR23:
         Comment=        "Temporary CDROM read error"
         catname= "othercat.cat"
         Class=  H
         Log=            True
         Report= True
         Alert=          False
         Err_Type=       TEMP
         Err_Desc=       {1, 1, "CD ROM is broken"}
         Prob_Causes=    {2, 1, "cause 1"},\
                        {2, 2, "Cause 2"}
         Fail_Causes=    E800, 6312
         Fail_Actions=   1601, 0000
         Detail_Data=    120, 11, HEX
```

```
        Detail_Data=    4, 8058, DEC
        Detail_Data=    4, 8059, DEC
```

The catalog othercat.cat will be used for the CDROM_ERR23 template only.

Note: A template may contain both XPG4 messages and the traditional error ids or codepoints.

Files

Item	Description
/usr/include/sys/errids.h	Contains the header file that contains Error IDs.
/usr/include/sys/err_rec.h	Contains the header file that contains structures for logging errors.

ethchan_config Command

Purpose

Adds adapters to an EtherChannel or removes adapters from an EtherChannel.

Syntax

ethchan_config { **-a** [**-b**] | **-d** } [**-p** *ParentName*] *EtherChannel Adapter*

ethchan_config -c [**-p** *ParentName*] *EtherChannel Attribute NewValue*

ethchan_config -f [**-p** *ParentName*] *EtherChannel*

Description

This command adds adapters to an EtherChannel or removes adapters from an EtherChannel.
This command can also be used to modify *EtherChannel* attributes. These additions, deletions, or modifications can take place even if the EtherChannel's interface is configured; that is, it is not necessary to detach the interface of EtherChannel to add or remove adapters or modify most EtherChannel attributes.

Flags

Item	Description
-a	Adds the specified *Adapter* to the specified *EtherChannel*. If the adapter must be added as a backup adapter, the **-b** flag must be specified.
-b	Specifies that the *Adapter* is being added as a backup adapter. This flag is only valid when used with the **-a** flag.
-c	Changes the specified *Attribute* of the specified *EtherChannel* attribute to the specified *NewValue*.
-d	Deletes the specified *Adapter* from the specified *EtherChannel*. The **-b** flag must not be used with the **-d** flag.
-f	Forces a failover of the specified *EtherChannel*. The failover occurs only if the adapter in the idle channel is up. If the adapter in the idle channel is down, the *EtherChannel* keeps operating on the active one and no failover takes place.
-p	Specifies the parent adapter of an EtherChannel. If a Shared Ethernet Adapter (SEA) is configured over an EtherChannel, this flag must be used along with the other flags to change any attribute of the EtherChannel (for example, adding or deleting adapters).

Parameters

Item	Description
Adapter	Specifies the adapter to add or delete.
Attribute	Specifies an attribute of the specified EtherChannel.
EtherChannel	Specifies the EtherChannel.
NewValue	Specifies the new value for the specified attribute of the specified EtherChannel.
ParentName	Specifies the parent adapter of an EtherChannel.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Examples

1. To add the adapter `ent0` as the backup adapter in the EtherChannel called `ent7`, enter the following command:

   ```
   /usr/lib/methods/ethchan_config -a -b ent7 ent0
   ```

2. To change the address to ping attribute of an EtherChannel called `ent7` to `10.10.10.10`, enter the following command:

   ```
   /usr/lib/methods/ethchan_config -c ent7 netaddr 10.10.10.10
   ```

3. To force a failover of an EtherChannel called `ent7` from the currently active channel to the idle channel, enter the following command:

   ```
   /usr/lib/methods/ethchan_config -f ent7
   ```

4. To delete the adapter `ent13` from an EtherChannel called `ent18`, which belongs to an SEA called `ent32`, enter the following command:

   ```
   /usr/lib/methods/ethchan_config -d -p ent32 ent18 ent13
   ```

Restrictions

The use of the *use_jumbo_frame* attribute cannot be modified by this command. If you attempt to modify this attribute, this command prints an error message.

Location

/usr/lib/methods

ewallevent Command

Purpose

Broadcasts an event or a rearm event to all users who are logged in.

Syntax

`ewallevent [-c] [-h]`

Description

The `ewallevent` script broadcasts a message on an event or a rearm event to all users who are currently logged in to the host when the event or the rearm event occurs. Event or rearm event information is captured and posted by the event response resource manager in environment variables that are generated by the event response resource manager when an event or a rearm event occurs. This script can be used as an action that is run by an event response resource. It can also be used as a template to create other user-defined actions. This script always returns messages in English.

Messages are displayed in this format at the consoles of all users who are logged in when an event or a rearm event occurs for which this script is a response action :

```
Broadcast message from user@host (tty) at hh:mm:ss...

severity event_type occurred for Condition condition_name
on the resource resource_name of resource_class_name at hh:mm:ss mm/dd/yy
The resource was monitored on node_name and resided on {node_names}.
```

Event information is returned about the ERRM environment variables, and also includes the following:

Local Time
Time when the event or rearm event is observed. The actual environment variable supplied by ERRM is ERRM_TIME. This value is localized and converted to readable form before being displayed.

This script captures the environment variable values and uses the `wall` command to write a message to the currently logged-in user consoles.

Flags

-c
Instructs `ewallevent` to broadcast the ERRM_VALUE of an ERRM event. When the `-c` flag is specified, `ewallevent` broadcasts the SNMP trap message.

-h
Writes the script's usage statement to standard output.

Parameters

log_file
Specifies the name of the file where event information is logged. An absolute path for the *log_file* parameter should be specified.

The *log_file* is treated as a circular log and has a fixed size of 64KB. When *log_file* is full, new entries are written over the oldest existing entries.

If *log_file* already exists, event information is appended to it. If *log_file* does not exist, it is created so that event information can be written to it.

Exit Status

0
Script has run successfully.

1
Error occurred when the script was run.

Restrictions

1. This script must be run on the node where the ERRM is running.

2. The `wall` command is used to write a message to currently logged-in user consoles. Refer to the `wall` man page for more information on the `wall` command.

Standard Output

When the `-h` flag is specified, the script's usage statement is written to standard output.

Examples

1. Suppose the `ewallevent` script is a predefined action in the critical-notification response, which is associated with the `/var space used` condition on the resource `/var`. The threshold of the event expression defined for this condition is met, and an event occurs. The critical-notification response takes place, and `ewallevent` is run. The following message is displayed on the consoles of all users who are logged in:

```
Broadcast message from joe@neverland.com (pts/6) at 18:42:03...

Critical event occurred for Condition /var space used
on the resource /var of filesys of IBM.FileSystem at 18:41:50 03/28/02
The resource was monitored on c174n05 and resided on {c174n05}.
```

2. When a rearm event occurs for the `/var space used` condition on the resource `/var`, the following message is displayed on the consoles of all users who are logged in:

```
Broadcast message from joe@neverland.com (pts/6) at 18:42:03...

Critical rearm event occurred for Condition /var space used
on the resource /var of filesys of IBM.FileSystem at 18:41:50 03/28/02
The resource was monitored on c174n05 and resided on {c174n05}.
```

Location

/opt/rsct/bin/ewallevent

ex Command

Purpose

Editor for text files.

Syntax

ex[**-c** *Subcommand*] [**-l**] [**-R**] [**-s**] [**-t***Tag*] [**-V**] [**-w***Number*] [**-v**| **-**] [**+**[*Subcommand*]] [**-r**[*File*]] [*File...*]

Description

The **ex** command starts the ex editor. The ex editor is part of a family of editors that includes the **edit** command editor, which is a simpler version of the ex editor for novice or casual use, and the **vi** command editor, which is a full-screen display editor. Calling the vi editor directly sets environment variables for screen editing. The ex editor is more powerful than a simple line editor because it is a subset of the vi editor and can access the screen editing capabilities of the vi editor.

The *File* parameter specifies the file or files to be edited. If you supply more than one file name, the ex editor edits each file in the specified order.

Notes:

1. To determine how your workstation can perform more efficiently, the ex editor uses the workstation capability database **terminfo** and the type of the workstation you are using from the **TERM** environment variable.

2. The **ex** command affects the current line unless you specify otherwise. In order to work with different parts of the file, you need to know how to address lines in a file.
3. If the standard input is not a terminal device, it shall be as if you have specified the **-s** flag.

Flags

Item	Description
-c *Subcommand*	Carries out the ex editor subcommand before editing begins. When a null operand is typed, as in `-c ''`, the editor places the current line at the bottom of the file. (Usually, the ex editor sets the current line at the start of the file or at some specified tag or pattern.)
-l	Indents appropriately for LISP code and accepts the ( ) (open or close parenthesis), { } (left or right brace), and the [[ ]] (double left or double right bracket) characters as text rather than interpreting them as vi subcommands. This flag is active in visual and open modes.
-R	Sets the **readonly** option, preventing you from altering the file.
-s	Suppresses all interactive-user feedback. If you use this flag, file input and output errors do not generate a helpful error message. Using this flag is the same as using the **-** flag. Ignore the value of TERM and any implementation default terminal type and assume the terminal is a type incapable of supporting open or visual modes.
-t *Tag*	Loads the file that contains the tag indicated by the parameter *Tag* and positions the editor at that tag. To use this flag, you must first create a database of function names and their locations using the **ctags** command.
-w*Number*	Sets the default window size to *Number*.
-v	Invokes the vi editor. **Note:** When the **-v** flag is selected, an enlarged set of subcommands are available, including screen editing and cursor movement features. See the **vi** command.
-V	Invokes the editor in verbose mode.
-	Suppresses all interactive-user feedback. If you use this flag, file input/output errors do not generate a helpful error message. Using this flag is the same as using the **-s** flag.
+[*Subcommand*]	Begins an edit at the specified editor search or subcommand. When no parameter is typed, the **+***Subcommand* places the current line at the bottom of the file. Usually, the ex editor sets the current line to the start of the file, or to some specified tag or pattern.
-r [*File*]	Recovers a file after an editor or system crash. If you do not specify the *File* parameter, a list of all saved files is displayed.

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	An error occurred.

Files

Item	Description
/usr/lbin/exrecover	Recover subcommand
/usr/lbin/expreserve	Preserve subcommand
$HOME/.exrc	Editor startup file
./.exrc	Editor startup file
/var/tmp/Ex*nnnnn*	Editor temporary
/var/tmp/Rx*nnnnn*	Names buffer temporary
/var/preserve	Preservation directory

execerror Command

Purpose

Writes error messages to standard error.

Syntax

execerror

Description

The **execerror** command is executed by an **exec** subroutine when the load of the real program is unsuccessful. It is passed the name of the file being executed and zero or more loader error message strings. Each loader error message string contains an error number followed by error data.

Examples

The **execerror** command is used as follows:

```
char *buffer[1024];
buffer[0] = "execerror" ;
buffer[1] = "name of program that failed to load";
loadquery(L_GETMESSAGES, &buffer[2], sizeof buffer -8);
execvp("/usr/sbin/execerror",buffer);
```

This sample code causes the application to terminate after the messages are written to standard error.

Files

Item	Description
/usr/sbin/execerror	Contains the **execerror** command.

execrset Command

Purpose

Runs a program or command attached to an rset.

Syntax

execrset [**-P**] [**-F**] **-c** *CPUlist* [**-m** *MEMlist*] **-e** *command* [*parameters*]

or

execrset [**-P**] [**-F**] [**-S**] *rsetname* [**-e**] *command* [*parameters*]

Description

The **execrset** command executes a command with an attachment to an **rset**. It causes the specified command to be limited to running only on the processors and/or memory regions contained in the rset. An **rset** name in the system registry can be used to specify the processors and/or memory regions the command is allowed to use. Or, an **rset** containing the specified processors and memory regions can be attached to the process.

Flags

Item	Description
-F	Force the **execrset** command to occur. This flag removes a bindprocessor bind and all threads' `rset` in the process before issuing the command. If the **-P** flag is also specified, it detaches the effective **rset** and all threads' `rset` from the process before issuing the command.
-P	Attaches an **rset** as a partition rset.
-c *CPUlist*	List of CPUs to be in the **rset** to be attached to the process which executes the program or command. This can be one or more CPUs or CPU ranges.
-m *MEMlist*	List of memory regions to be in the **rset**. This can be one or more memory regions or ranges.
-e *command* [*parameters*]	Specifies the command to run followed by any parameters. The **-e** flag must be the last flag used in the command.
-S	A hint that indicates that the process must be scheduled to run in single-threaded mode. Only one of the hardware threads of each physical processor that is included in the specified rset will be used to schedule the job. If all the hardware threads of a physical processor are not included in the specified rset, that processor will be ignored. The specified rset must be an exclusive rset or the command fails. Specifying this flag allows jobs to run with single-thread behavior.

Parameters

Item	Description
rsetname	The name of the **rset** in the system registry to be attached to the process executing the program or command

Security

The user must have root authority or have **CAP_NUMA_ATTACH** capability. The user must have root authority to attach a partition rset to the command's process (the **-P** flag).

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To run the test1 program on CPUs 0-7, type:

```
execrset -c 0-7 -e test1
```

2. To run the 'test2 parm1 parm2' program with an attachment to rset named **test/cpus0to15**, type:

```
execrset test/cpus0to15 test parm1 parm2
```

3. To run the **ls -l** command on CPU 0, type:

```
execrset -c 0 -e ls -l
```

Files

Item	Description
/usr/bin/execrset	Contains the **execrset** command.

expand Command

Purpose

Writes to standard output with tabs changed to spaces.

Syntax

expand [**-t** *TabList*] [*File ...*]

expand [*-tabstop*]|[*-tab1*,*tab2*,...,*tabn*] [*File ...*]

Description

The **expand** command writes the named files or standard input to standard output, and replaces the tab characters with one or more space characters. Any backspace characters are copied to the output and cause the column position count for tab stop calculations to decrement; the column position count will not decrement below zero.

Note: The *File* parameter must be a text file.

Flags

Item	Description
-t *TabList*	Specifies the position of the tab stops. The default value of a tab stop is 8 column positions. The *TabList* variable must consist of a single positive-decimal integer or multiple positive-decimal integers. The multiple integers must be in ascending order, and must be separated by commas or by blank characters with quotation marks around the integers. The single *TabList* variable sets the tab stops an equal number of column positions apart. The multiple *TabList* variable sets the tab stops at column positions that correspond to the integers in the *TabList* variable. If the **expand** command processes a tab stop beyond the last one specified in the *TabList* variable, the tab stop is replaced by a single-space character in the output.

Parameters

Item	Description
tabstop	Specified as a single argument. It sets *tabstop* SPACE characters apart instead of the default 8.
tab1, tab2,..., tabn	Sets TAB characters at the columns specified by *-tab1,tab2,...,tabn*.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To adjust the tab stops an equidistance amount in `text.fil`, enter:

   ```
   expand -t 3 text.fil
   ```

   If `text.fil` contains:

   ```
   1       2        3456789
   ```

   then the **expand** command displays:

   ```
   1  2        3456789
   ```

2. To adjust the tab stops a varied amount in `text.fil`, enter:

   ```
   expand -t 3,15,22  text.fil
   ```

   OR

   ```
   expand -t "3 15 22" text.fil
   ```

   If `text.fil` contains:

   ```
   1      2       3      456789
   ```

   then the **expand** command displays:

   ```
   1  2       3      456789
   ```

Files

Item	Description
/usr/bin/expand	Contains the **expand** command.

expfilt Command

Purpose

Exports filter rules to an export file.

Syntax

expfilt [**-p**] [**-q**] [**-r**] [**-v 4** | **6**] **-f** *directory* [**-l** *filt_id_list*]

Description

Use the **expfilt** command to export filter rules into export text files, which can be used by the **impfilt** command. This is useful if you want to define similar rules on multiple machines.

Note: The filter description on one machine might be meaningless or misleading in another machine. This field is not exported.

IPsec filter rules for this command can be configured by using the **genfilt** command, or IPsec smit (IP version 4 or IP version 6).

Flags

Item	Description
-f *directory*	Specifies the directory to create the exported text files. The directory will be created if it does not exist.
-l *filt_id_list*	Lists the IDs of the filter rules you want to export. The filter rule IDs can be separated by "**,**" or "**-**". If this flag is not used, all the filter rules defined in the filter rule table for the applicable IP versions will be exported.
-p	Allows predefined rules.
-q	Specifies quiet mode. Suppresses output to **stdout**.
-r	Specifies raw mode. Exports filter rules as is and does not reverse direction on rules. Use this flag when filter rules are exported and imported as is; for example, to save a configuration or replicate a configuration to another machine. With the **-r** flag, the direction of the traffic will be preserved. For instance if there is a rule on host 10.0.0.1 to permit inbound traffic from 10.0.0.2, expfilt with the -r flag will write the same filter rule. Omitting the **-r** flag will cause the direction to be switched from inbound to outbound in the export file.
-v	IP version of the filter rules you want to export. The value of **4** specifies IP version 4 and the value of **6** specifies IP version 6. When this flag is not used, both IP version 4 and IP version 6 rules are exported.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

explain Command

Purpose

Provides an interactive thesaurus.

Syntax

explain

Description

The **explain** command provides an interactive thesaurus for the English-language phrases found by the **diction** command. Before using the **explain** command, use the **diction** command to obtain a list of poorly worded phrases. When you use the **explain** command, the system prompts you for a phrase and responds with a grammatically acceptable alternative. You can continue typing phrases, or you can exit by entering the Ctrl-D key sequence.

No other command line parameters are valid.

Files

Item	Description
/usr/lib/explain.d	Contains thesaurus.

explore Command

Purpose

Starts the WebExplorer World Wide Web browser.

Syntax

explore [**-i***FileName*] [**-t***Number*] [**-q**] [[**-url**] *URL*]

Description

The **explore** command opens the WebExplorer main window and connects to the Uniform Resource Locator (URL) for the home document.

Flags

Item	Description
-i*FileName*	Specifies an alternate initialization file, where *FileName* is the full path name of the file to use instead of the default **$HOME/.explore-preferences**. This allows you to start the WebExplorer with an alternate set of user preferences.
-t*Number*	Specifies the number of threads to use for loading images, where *Number* is the number of image loader threads. Each thread is represented in the status area of the main window. A maximum of eight can be specified, and the default is four.
-q	Specifies quiet mode. This suppresses the WebExplorer title window when you start the application and bypasses the confirmation window when you exit.
-url *URL*	Specifies a particular document to load when starting WebExplorer, where *URL* is the URL of the document to load. If WebExplorer has a home document defined, this URL will override it. You do not have to precede the URL with the **-url** flag. If you specify the URL by itself, WebExplorer will accept it.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.

Item	Description
>0	An error occurred.

Security

Access Control: Any User

Auditing Events: N/A

Examples

To start the browser without the title window appearing and go directly to the Dilbert Zone URL, enter:

```
explore -q  http://www.unitedmedia.com/comics/dilbert/
```

or

```
explore -q  -url http://www.unitedmedia.com/comics/dilbert/
```

Files

Item	Description
/usr/lpp/explorer/bin/explore	Contains the **explore** command.
$HOME/.explore-preferences	Contains the initialization file that specifies user preferences for settings such as the number of colors used.
$HOME/.mailcap	Contains the configuration file that maps mimetype to external viewers.
$HOME/.mimetypes	Contains the user-defined configuration file that maps mimetype to external viewers. It is set through the **Configure Viewers** dialog. this file overrides the **.mailcap** settings.

exportfs Command

Purpose

Exports and unexports directories to NFS clients.

Syntax

/usr/sbin/exportfs [**-a**] [**-v**] [**-u**] [**-i**] [**-f***File*] [**-F**] [**-o***Option* [,*Option* ...]] [**-V** *Exported Version*] [*Directory*]

Description

The **exportfs** command makes local directories available for Network File System (NFS) clients to mount. This command is normally invoked during system startup by the **/etc/rc.nfs**file and uses information in the **/etc/exports** file to export one or more directories, which must be specified with full path names.

The **/etc/xtab** file lists directories that are currently exported. To display this file, enter the **exportfs** command without flags or arguments. To alter the file or to alter the characteristics of one of its directories, root users can edit the **/etc/exports** file and run the **exportfs** command. Such alterations can be done at any time. Never edit the **/etc/xtab** file directly.

Note:

1. You cannot export a directory that is either a parent directory or a subdirectory of one that is currently exported and within the same file system.
2. NFS versions 2 and 3 allow both directories and files to be exported. Only directories can be exported for NFS version 4 access.
3. If two entries for the same directory with different versions 2 (or 3) and 4 exist in the **/etc/exports** file, the **exportfs** command exports both of the entries.
4. If the options for NFS versions 2 (or 3) and 4 are the same for a directory, there can be one entry in the **/etc/exports** file specifying `-vers=3:4`.

Flags

Item	Description
-a	Exports all directories listed in the exports file.
-v	Prints the name of each directory as it is exported or unexported.
-u	Unexports the directories you specify. When used with the **-a** flag, unexports all exported directories. When used with both the **-a** and **-f** flags, unexports all directories in the specified export file.
-i	Allows the exporting of directories not specified in the exports file or ignores the options in the exports file. Unless the **-f** flag is used to specify an alternate file, the **exportfs** command will normally consult the **/etc/exports** file for the options associated with the exported directory."
-f *File*	Specifies an export file, instead of the **/etc/exports** file, that contains a list of directories that you can export. This file should follow the same format as the **/etc/exports** file. NOTE: This alternate file will not be used for exporting directories automatically when the system and NFS is started. The **/etc/exports** file is the only file that is supported for specifying directories to export at system start.
-F	Specifies that a forced unexport should be performed. Use this flag only with the **-u** flag. This flag has no effect when unexporting a V2/V3 export. A V4 unexport can fail due to associated state. This flag forces the release of any state associated with a V4 export.

Item	Description
-o*Options*	Specifies the optional characteristics for the directory being exported. You can enter more than one variable by separating them with commas. For options taking a *Client* parameter, *Client* can specify a hostname, a dotted IP address, a network name, or a subnet designator. A subnet designator is of the form "@*host*/*mask*", where *host* is either a hostname or a dotted IP address and *mask* specifies the number of bits to use when checking access. If *mask* is not specified, a full mask is used. For example, the designator `@client.group.company.com/16` will match all Clients on the `company.com` subnet. A designator of `@client.group.company.com/24` will match only the Clients on the `group.company.com` subnet. Choose from the following options: **ro** Exports the directory with read-only permission. If not specified, the directory is exported with read-write permission. **ro=*Client[:Client]*** Exports the directory with read-only permission to the specified Clients. Exports the directory with read-write permissions to Clients not specified in the list. A read-only list cannot be specified if a read-write list has been specified. **rw** Exports the directory with read-write permission to all Clients. **rw=*Client [:Client]*** Exports the directory with read-write permission to the specified Clients. Exports the directory read-only to Clients not in the list. A read-write list cannot be specified if a read-only list has been specified. **anon =*UID*** Uses the *UID* value as the effective user ID, if a request comes from a root user. The default value for this option is -2. In NFS version 2 and NFS version 3, setting the value of the *anon* option to -1 disables anonymous access. Thus, by default, secure NFS accepts nonsecure requests as anonymous, and users who want more security can disable this feature by setting *anon* to a value of -1. **root=*Client[:Client]*** Allows root access from the specified clients in the list. Putting a host in the root list does not override the semantics of the other options. For example, this option denies the mount access from a host present in the root list but absent in the access list. **access=*Client[:Client,...]*** Gives mount access to each client listed. A client can be either a host name or a net group name. Each client in the list is first checked for in the **/etc/netgroup** database and then in the **/etc/hosts** database. The default value allows any machine to mount the given directory. **secure** Requires clients to use a more secure protocol when accessing the directory.

Item	Description
-o *Options* (continued)	**sec=*flavor[:flavor...]*** This option is used to specify a list of security methods that may be used to access files under the exported directory. Most exportfs options can be clustered using the `sec` option. Options following a `sec` option are presumed to belong with the preceding `sec` option. Any number of `sec` stanzas may be specified, but each security method can be specified only once. Within each `sec` stanza the `ro`, `rw`, `root`, and `access` options may be specified once. Only the `public`, `anon` and `vers` options are considered global for the export. If the `sec` option is used to specify any security method, it must be used to specify all security methods. In the absence of any `sec` option, all authentication flavors are allowed. Allowable flavor values are: **sys** UNIX authentication. This is the default method. **dh** DES authentication. **none** Allow mount requests to proceed with anonymous credentials if the mount request uses an authentication flavor not specified in the export. **krb5** Kerberos. Authentication only. **krb5i** Kerberos. Authentication and integrity. **krb5p** Kerberos. Authentication, integrity, and privacy. The `secure` option may be specified, but not in conjunction with a `sec` option. The `secure` option is deprecated and may be eliminated. Use `sec=dh` instead. **vers=*version_number[:version_number...]*** Specifies which versions of NFS are allowed to access the exported directory. Valid versions are 2, 3, and 4. Versions 2 and 3 cannot be selected exclusively. Specifying either version 2 or version 3 will allow access by both NFS version 2 and NFS version 3. Version 4 can be selected exclusively. The default is to allow access using NFS protocol versions 2 and 3. **exname=*external-name*** Exports the directory by the specified external name. The external name must begin with the `nfsroot` name. See the description of the `/etc/exports` file for a description of the `nfsroot` name. This option applies only to directories exported for access by NFS version 4 protocol. **deleg={yes | no}** Enables or disables file delegation for the specified export. This option overrides the system-wide delegation enablement for this export. The system-wide enablement is done through **nfso**.

Item	Description
-o *Options* (continued)	**refer=*rootpath*@*host*[+*host*][:*rootpath*@*host*[+*host*]]** A namespace referral will be created at the specified path. The referral directs clients to the specified alternate locations where they can continue operations. A referral is a special object. If a nonreferral object exists at the specified path, the export is disallowed and an error message is printed. If nothing exists at the specified path, a referral object is created there that includes the path name directories leading to the object. Multiple referrals can be created within a file system. A referral cannot be specified for the `nfsroot`. The name `localhost` cannot be used as a *hostname*. This `refer` option is allowed only for version 4 exports. If the export specification allows version 2 or version 3 access, an error message will be printed and the export will be disallowed. Unexporting the referral object has the effect of removing the referral locations information from the referral object. The object itself is not removed by unexporting. Use `rm` if you want to remove the object. The administrator must ensure that appropriate data is available at the referral servers. This option is available only on AIX 5L Version 5.3 with the 5300-03 Recommended Maintenance package or later. **Note:** A referral export can only be made if replication is enabled on the server. Use `chnfs -R` on to enable replication.

Item	Description
-o *Options* (continued)	**replicas=*rootpath*@*host*[+*host*][:*rootpath*@*host*[+*host*]]** Replica location information will be associated with the export path. The replica information can be used by NFS version 4 clients to redirect operations to the specified alternate locations if the current server becomes unavailable. The administrator should ensure that appropriate data is available at the replica servers. Because replica information applies to an entire file system, the specified path must be the root of a file system. If the path is not a file system root, the export is disallowed and an error message is printed. The name `localhost` cannot be used as a *hostname*. This `replicas` option is meaningful only for version 4 exports. If the option is used on an export that allows version 2 or version 3 access, the operation is allowed, but the replica information is ignored by the version 2 and version 3 servers. If the directory being exported is not in the replica list, the entry *exported directory*@*current host* will be added as the first replica location. This option is available only on AIX 5.3 with 5300-03 or later. A replica export can only be made if replication is enabled on the server. By default, replication is not enabled. If replica exports will be made at system boot, replication should be enabled by using the `chnfs -R on` command. Replica locations can also be specified for the `nfsroot`. This can be done only using `chnfs -R` *host*[+*host*]. If the current host is not specified in the list, it will be added as the first replica host. The *rootpath* is not needed or allowed in this case because `nfsroot` is replicated only to the `nfsroots` of the specified hosts. The `chnfs` program can be used to enable or disable replication. Changing the replication mode can only be done if no NFS version 4 exports are active. If the server's replication mode is changed, file handles issued by the server during the previous replication mode will not be honored by the server. This can cause application errors on clients holding old file handles. Be careful when changing the replication mode of the server. If possible, all clients who have mounts to the server should unmount them before the server's replication mode is changed. The replica location information associated with the directory can be changed by modifying the replica list and reexporting the directory. The new replica information replaces the old replica information. NFS clients are expected to refresh replica information on a regular basis. If the server changes the replica information for an export, it might take time for the client to notice. This is not much of a problem if new replica locations are added, because clients holding the old information still have correct, if incomplete, replica information. Removing replica information can be problematic because it can result in clients holding incorrect replica information for a period of time. To aid clients in detecting the new information, `exportfs` will attempt to touch the replicated directory. This changes the timestamps on the directory, which in turn causes the client to refetch the directory's attributes. This operation might not be possible, however, if the replicated file system is read-only. When changing replica information for a directory, be aware that there could be some latency between changing the information and clients noticing the new information.

Item	Description
-o *Options* (continued)	**noauto** Accepts the replicas specification as-is. Does not automatically insert the primary hostname as one of the replica locations if it has not been specified. **scatter** Defines how the alternate locations list is generated from the servers specified on the **refer** or **replicas** option. If the **noauto** option is not used, the alternate locations list also includes the primary host name as one of the replica locations. The **scatter** option applies only to directories exported for access by NFS version 4 protocol. The **scatter** option has three allowable values: **full** All of the servers are scattered to form the combinations of alternate locations. **partial** The first location of all the combinations is fixed to the first server specified on the **refer** or **replicas** option. The rest of the locations and the first location are scattered as if they are scattered using the `scatter=full` method. **none** No scatter is to be used. The value can also be used to disable scattering if enabled previously. Whenever the attributes of a Client change, all export entries that contain that Client as a parameter should be exported again. Events that can change a Client's attributes include modifying a netgroup or changing the IP address of a client. Failure to do so can result in the server using old client information.
-V *Exported Version*	Specifies the version number. Valid version numbers are 2, 3 and 4.

Solaris Compatibility

The **exportfs** command may be invoked as **share**, **shareall**, **unshare**, or **unshareall**. When the **exportfs** command is invoked as **share** or **shareall**, the functionality is equivalent to **exportfs** and **exportfs -a**, respectively, except that the **sec** option must be used to specify the security methods. When the **exportfs** command is invoked as **unshare** or **unshareall**, the functionality is equivalent to **exportfs -u** and **exportfs -u -a**, respectively.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Examples

1. To export all directories in the **/etc/exports** file, enter:

   ```
   exportfs -a
   ```

2. To export one directory from the **/etc/exports** file, enter:

   ```
   exportfs /home/notes
   ```

   In this example, the `/home/notes` directory is exported.

Note: For this command to work, the **/home/notes** directory must be specified in the **/etc/exports** file.

3. To unexport a directory, enter:

```
exportfs -u /home/notes
```

   In this example, the /home/notes directory is unexported.

4. To display the name of the directory currently being exported, enter:

```
exportfs -v
```

5. To export a directory that is not specified in the **/etc/exports** file, enter:

```
exportfs -i /home/zeus
```

   In this example, the /home/zeus directory is exported without restrictions.

6. To export a directory and give netgroup members permission to access this directory, enter:

```
exportfs access=cowboys:oilers /home/notes -o
```

   In this example, the /home/notes directory is exported and permits users of cowboys and oilers host machines to have access.

7. To export a directory with different options from the **/etc/exports** file, enter:

```
exportfs -i -o root=zorro:silver /directory
```

   In this example, the /directory directory is exported and allows root user access to zorro and silver host machines, regardless of the access permissions specified in the **/etc/exports** file.

8. To export the **/common/docs** directory with write permissions to clients using Kerberos authentication, but read-only permissions to clients using UNIX authentication, add the following text to the **/etc/exports** file:

```
/common/docs -sec=krb5,rw,sec=sys,ro
```

   Then enter exportfs /common/docs to perform the export.

9. To create a referral at /usr/info to the /usr/info directory on host infoserver, add the following line to /etc/exports and then export /usr/info:

```
/usr/info -vers=4,refer=/usr/info@infoserver
```

10. To specify replicas for the /common/info directory at hosts backup1 and backup2, add the following line to /etc/exports and then export /common/info:

```
/common/info -vers=4,replicas=/common/info@backup1:/common/info@backup2,<other options>
```

11. To export the **/common/docs** directory with both version 3 and version 4, enter the following command:

```
exportfs -V 3:4 /common/docs
```

12. To export all of the version 4 entries in the **/etc/exports** file, enter the following command:

```
exportfs -a -V 4
```

13. To unexport the **/common/docs** directory only for version 3, enter the following command:

```
exportfs -u -V 3 /common/docs
```

14. To unexport all of the version 3 entries in the **/etc/xtab** file, enter the following command:

```
exportfs -ua -V 3
```

15. To specify referrals for the **/common/docs** directory at hosts named s1, s2, and s3 and scatter them fully, add the following line to the **/etc/exports** file and then export the **/common/docs** directory:

```
/common/docs -vers=4,refer=/common/docs@s1:/common/docs@s2:/common/docs@s3,scatter=full
```

16. To specify replicas for the **/common/docs** directory at hosts named s1, s2, s3, and s4 and scatter them partially (the first fail over server is s1 for all combinations), add the following line to the **/etc/exports** file and then export the **/common/docs** directory:

```
/common/docs -vers=4,noauto,replicas=/common/docs@s1:/common/docs@s2:/common/docs@s3:/common/docs@s4,scatter=partial
```

Files

Item	Description
/etc/exports	Lists the directories that the server can export.
/etc/xtab	Lists currently exported directories.
/etc/hosts	Contains an entry for each host on the network.
/etc/netgroup	Contains information about each user group on the network.
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.

exportvg Command

Purpose

Exports the definition of a volume group from a set of physical volumes.

Syntax

```
exportvg [-b] VolumeGroup
```

Description

The **exportvg** command removes the definition of the volume group specified by the *VolumeGroup* parameter from the system. Since all system knowledge of the volume group and its contents are removed, an exported volume group can no longer be accessed. The **exportvg** command does not modify any user data in the volume group.

A volume group is a nonshared resource within the system; it should not be accessed by another processor until it has been explicitly exported from its current processor and imported on another. The primary use of the **exportvg** command, coupled with the **importvg** command, is to allow portable volumes to be exchanged between processors. Only a complete volume group can be exported, not individual physical volumes.

Using the **exportvg** command and the **importvg** command, you can also switch ownership of data on physical volumes shared between two processors.

Note: To use this command, you must either have root user authority or be a member of the **system** group.

You can use the System Management Interface Tool (SMIT) **smit exportvg** fast path to run this command.

Notes:

1. A volume group that has a paging space volume on it cannot be exported while the paging space is active. Before exporting a volume group with an active paging space volume, ensure that the paging space is not activated automatically at system initialization, and then reboot the system.
2. The mount point information of a logical volume would be missing from the LVCB (logical volume control block) if it is longer than 128 characters. If the mount points are longer than 128 characters, you must edit the `/etc/filesystems` file manually when you run the **importvg** command to import this volume group completely.

Flags

-b
Backs up the performance tunable parameters of the volume group that are set with the **lvmo** command before the **exportvg** command exports the volume group. When you use this flag, the **exportvg** command output displays the name of the file in which the performance tunable parameters are saved.

Exit Status

This command returns the following exit values:

Item	Description
`0`	Successful completion.
`>0`	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To remove volume group `vg02` from the system, enter:

```
exportvg vg02
```

Note: The volume group must be varied off before exporting.

The definition of `vg02` is removed from the system and the volume group cannot be accessed.

Files

Item	Description
/usr/sbin	Directory where the **exportvg** command resides.

expr Command

Purpose

Evaluates arguments as expressions.

Syntax

expr *Expression*

Description

The **expr** command reads the *Expression* parameter, evaluates it, and writes the result to standard output.

You must apply the following rules to the *Expression* parameter:

- Separate each term with blanks.
- Precede characters special to the shell with a \ (backslash).
- Quote strings containing blanks or other special characters.

Integers may be preceded by a unary hyphen. Internally, integers are treated as 32-bit, twos complement numbers.

Note: The **expr** command returns 0 to indicate a zero value, rather than the null string.

The following items describe *Expression* parameter operators and keywords. Characters that need to be escaped are preceded by a \ (backslash). The items are listed in order of increasing precedence, with equal precedence operators grouped within { } (braces):

Item	Description
Expression1 **|** *Expression2*	Returns *Expression1* if it is neither a null value nor a 0 value; otherwise, it returns *Expression2*.
Expression1 **\&** *Expression2*	Returns *Expression1* if both expressions are neither a null value nor a 0 value; otherwise, it returns a value of 0.
Expression1 **{ =, \>, \>=, \<, \<=, != }** *Expression2*	Returns the result of an integer comparison if both expressions are integers; otherwise, it returns the result of a string comparison.
Expression1 **{+, - }** *Expression2*	Adds or subtracts integer-valued arguments.
Expression1 **{ *, /, % }** *Expression2*	Multiplies, divides, or provides the remainder from the division of integer-valued arguments.
Expression1 **:** *Expression2*	Compares the string resulting from the evaluation of *Expression1* with the regular expression pattern resulting from the evaluation of *Expression2*. Regular expression syntax is the same as that of the **ed** command, except that all patterns are anchored to the beginning of the string (that is, only sequences starting at the first character of a string are matched by the regular expression). Therefore, a ^ (caret) is not a special character in this context. Normally the matching operator returns the number of characters matched (0 on failure). If the pattern contains a subexpression, that is: `\( Expression \)` then a string containing the actual matched characters is returned. A collating sequence can define equivalence classes for use in character ranges. See "Understanding Locale Environment Variables" in *Globalization Guide and Reference* for more information on collating sequences and equivalence classes.

Note: The following string arguments are extensions beyond that of the standards, and the behavior may be different across operating systems. These string arguments are NOT portable.

Item	Description
match *String1 String2*	Same as *Expression1* **:** *Expression2*.
length *String1*	Returns the length of the *String1*.
index *String1 String2*	Returns the first position in *String1* where any character in *String2* exists.
substr *String1 StartPosition Length*	Returns a string that starts with the character at *StartPosition* in *String1* and continuies for *Length* characters

Exit Status

This command returns the following exit values:

Item	Description
0	The *Expression* parameter evaluates to neither null nor 0.
1	The *Expression* parameter evaluates to null or 0.
2	The *Expression* parameter is not valid.
>2	An error occurred.

Note: After parameter processing by the shell, the **expr** command cannot distinguish between an operator and an operand except by the value. Thus, if the value of $a is `j`, the command:

```
expr $a = j
```

looks like:

```
expr j = j
```

after the shell passes the arguments to the **expr** command. The following is also true:

```
expr X$a = Xj
```

Examples

1. To modify a shell variable, enter:

   ```
   COUNT=`expr $COUNT + 1`
   ```

   This adds 1 to the shell variable $COUNT. The **expr** command is enclosed in grave accents, which causes the shell to substitute the standard output from the **expr** command into the COUNT= command. The $COUNT variable must be initialized before using.

2. To find the length of the **$STR** shell variable, enter:

   ```
   LENGTH=`expr $STR : ".*"`
   ```

   This sets the LENGTH variable to the value given by the: (colon) operator. The pattern .* (dot, asterisk) matches any string from beginning to end, so the colon operator gives the length of the $STR variable as the number of characters matched. Note that `.*` must be within quotes to prevent the shell from treating the * (asterisk) as a pattern-matching character. The quotes are not part of the pattern.

If the $STR variable is set to the null string or contains any white space (blanks or tabs), then the command displays the error message `expr: syntax error`. This happens because the shell does not normally pass null strings to commands. In this case, the **expr** command sees only:

```
:.*
```

The shell also removes the single quotation marks. This does not work because the colon operator requires two values. The problem is fixed by enclosing the shell variable in double quotation marks:

```
LENGTH=`expr "$STR" : ".*"`
```

Now if the value of the $STR variable is null, the LENGTH variable is set to a value of 0. Enclosing shell variables in double quotation marks is generally recommended. Do not enclose shell variables in single quotation marks.

3. To use part of a string, enter:

```
FLAG=`expr "$FLAG" : "-*\(.*\)"`
```

This removes leading hyphens, if any, from the $FLAG shell variable. The colon operator gives the part of the FLAG variable matched by the subexpression enclosed between \(and \) characters (backslash, open parenthesis and backslash, close parenthesis). If you omit the \(and \) subexpression characters, the colon operator gives the number of characters matched.

If the $FLAG variable is set to - (hyphen), the command displays a syntax error message. This happens because the shell substitutes the value of the $FLAG variable before running the **expr** command. The **expr** command does not know that the hyphen is the value of a variable. It can only see:

```
- : -*\(.*\)
```

and it interprets the first hyphen as the subtraction operator. To eliminate this problem, use:

```
FLAG=`expr "x$FLAG" : "x-*\(.*\)"`
```

4. To use the **expr** command in an **if** statement, enter:

```
if expr "$ANSWER" : "[yY]" >/dev/null
then
echo ANSWER begins with "y" or "Y"
fi
```

If the $ANSWER variable begins with y or Y, the `then` part of the **if** statement is performed. If the match succeeds, the result of the expression is 1 and the **expr** command returns an exit value of 0, which is recognized as the logical value True by the **if** statement. If the match fails, the result is 0 and the exit value 1 (False).

Redirecting the standard output of the **expr** command to the **/dev/null** special file discards the result of the expression. If you do not redirect it, the result is written to the standard output, which is usually your workstation display.

5. Consider the following expression:

```
expr "$STR" = "="
```

If the $STR variable has the value = (equal sign), then after the shell processes this command the **expr** command sees the expression:

```
= = =
```

The **expr** command interprets this as three = operators in a row and displays a syntax error message. This happens whenever the value of a shell variable is the same as that of one of the **expr** operators. You can avoid this problem by phrasing the expression as:

```
expr "x$STR" = "x="
```

6. To return the length of the $SHELL environment variable, /usr/bin/ksh, enter:

```
expr length $SHELL
```

   The following is displayed:

```
12
```

7. To return the first position of where any characters in the string "de" is found in "abcdef", enter:

```
expr index abcdef de
```

   The following is displayed:

```
4
```

8. To return the first position of where any characters in the string "fd" is found in "abcdef", enter:

```
expr index abcdef fd
```

   The following is displayed:

```
4
```

9. To return the string starting at position 11, for a length of 6 of the string "Goodnight Ladies", enter:

```
expr substr "Goodnight Ladies" 11 6
```

   The following is displayed:

```
Ladies
```

Files

Item	Description
/usr/bin/expr	Contains the **expr** command.

exptun Command

Purpose

Exports a tunnel definition and, optionally, all the user defined filter rules associated with the tunnel. Creates a tunnel export file and an optional filter rule export file that can be used for the tunnel partner.

Syntax

exptun [**-v 4**|**6**] **-f** *directory* [**-t** *tid_list*] [**-r**] [**-l manual**]

Description

Use the **exptun** command to create a tunnel context export file and, optionally, a filter rule appendage file for a tunnel partner to import. This command does not activate a tunnel, it simply creates the required files for the tunnel partner.

Note: Generated export files contain keys used by the tunnel. Protect these files with the operating system file system protection features.

Flags

Item	Description
-f	Defines the directory where the export files are to be written. The directory will be created if it does not exist. The export files may then be sent to the tunnel partner to be imported. It is recommended that export files for each tunnel partner have a different directory specification.
-l	The type of the tunnel(s) you want to export. If manual is specified, only manual ibm tunnel(s)are exported.
-r	Exports all the user defined filter rules associated with the tunnel(s). If this flag is not used, only the tunnel definitions will be exported.
-t	Specifies the list of tunnel IDs to be used for the export files. The list may be specified as a sequence of tunnel IDs separated by a **","** or **"-"** (1, 3, 10, 50-55). If this flag is not used, all tunnel definitions from the tunnel database will be exported.
-v	The IP version of the tunnels being exported. Value **4** specifies IP version 4 tunnels. Value **6** specifies IP version 6 tunnels. If this flag is not used, both IP version 4 and IP version 6 tunnel definitions will be exported.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

extendlv Command

Purpose

Increases the size of a logical volume by adding deallocated physical partitions from within the volume group.

Syntax

To Add Available Physical Partitions

extendlv [**-a** *Position*] [**-e** *Range*] [**-u** *Upperbound*] [**-s** *Strict*] *LogicalVolume Partitions* [*PhysicalVolume ...*]

To Add Specific Physical Partitions

extendlv [**-m***MapFile*] *LogicalVolume Partitions*

Description

The **extendlv** command increases the number of logical partitions allocated to the *LogicalVolume* by allocating the number of additional logical partitions represented by the *Partitions* parameter. The *LogicalVolume* parameter can be a logical volume name or a logical volume ID. To limit the allocation to specific physical volumes, use the names of one or more physical volumes in the *PhysicalVolume* parameter; otherwise, all the physical volumes in a volume group are available for allocating new physical partitions.

By default, the logical volume is expanded using the existing characteristics that are displayed when you use the lslv command. To override these existing characteristics for the new partitions only, choose different values for these characteristics by using the flags.

The default maximum number of partitions for a logical volume is 512. Before extending a logical volume more than 512 logical partitions, use the **chlv** command to increase the default value.

The default allocation policy is to use a minimum number of physical volumes per logical volume copy, to place the physical partitions belonging to a copy as contiguously as possible, and then to place the physical partitions in the desired region specified by the **-a** flag. Also, by default, each copy of a logical partition is placed on a separate physical volume.

You can specify logical volumes sizes in 512 Blocks/KB/MB/GB when using the **extendlv** command. (See “Examples” on page 1309.)

Note:

1. When extending a striped logical volume, the number of partitions must be in an even multiple of the striping width.
2. It is recommended that a logical volume using a large number of partitions (more than 800MB) be extended gradually in sections.
3. Changes made to the logical volume are not reflected in the file systems. To change file system characteristics, use the **chfs** command.
4. You must either have root user authority or be a member of the **system** group to use this command.
5. The **extendlv** command is not allowed on a snapshot volume group.

You can use the System Management Interface Tool (SMIT) **smit extendlv** fast path to run this command.

Flags

Note: The **-e** and **-s** flags are not valid with a striped logical volume.

Item	Description
-a *Position*	Sets the intraphysical volume allocation policy (the position of the logical partitions on the physical volume). The *Position* variable can be one of the following: **m** Allocates logical partitions in the outer middle section of each physical volume. This is the default position. **c** Allocates logical partitions in the center section of each physical volume. **e** Allocates logical partitions in the outer edge section of each physical volume. **ie** Allocates logical partitions in the inner edge section of each physical volume. **im** Allocates logical partitions in the inner middle section of each physical volume.
-e *Range*	Sets the interphysical volume allocation policy (the number of physical volumes to extend across, using the volumes that provide the best allocation). The value of the *Range* variable is limited by the *Upperbound* variable (set with the -u flag) and can be one of the following: **x** Allocates logical partitions across the maximum number of physical volumes. **m** Allocates logical partitions across the minimum number of physical volumes.

Item	Description
-m *MapFile*	Specifies the exact physical partitions to allocate. Partitions are used in the order given by the file designated by the *MapFile* parameter. All physical partitions belonging to a copy are allocated before allocating for the next copy. The *MapFile* format is: **PVname:PPnum1[-PPnum2]** where *PVname* is a physical volume name (for example, `hdisk0`). It is one record per physical partition or a range of consecutive physical partitions. **PVname** Name of the physical volume as specified by the system. **PPnum** Physical partition number. **Important:** When you use map files, you must understand and adhere to all LV-allocation parameters such as strictness, upperbound, and stripe width. Using map files bypasses the checks done in the LVM-allocation routines. This is important for striped LVs, which are assumed to have a typical striped allocation pattern conforming to the stripe width.
-s *Strict*	Determines the strict allocation policy. Copies of a logical partition can be allocated to share or not to share the same physical volume. The *Strict* variable is represented by one of the following: **y** Sets a strict allocation policy, so copies for a logical partition cannot share the same physical volume. **n** Does not set a strict allocation policy, so copies for a logical partition can share the same physical volume. **s** Sets a super strict allocation policy, so that the partitions allocated for one mirror cannot share a physical volume with the partitions from another mirror. **Note:** When changing a non superstrict logical volume to a superstrict logical volume you must specify physical volumes or use the **-u** flag.
-u *Upperbound*	Sets the maximum number of physical volumes for new allocation. The value of the *Upperbound* variable should be between one and the total number of physical volumes. When using super strictness, the upper bound indicates the maximum number of physical volumes allowed for each mirror copy. When using striped logical volumes, the upper bound must be multiple of *Stripe_width*.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To increase the size of the logical volume represented by the `lv05` directory by three logical partitions, type:

```
extendlv lv05 3
```

2. To request a logical volume named `lv05` with a minimum size of 10MB, type:

```
extendlv lv05 10M #
```

The **extendlv** command will determine the number of partitions needed to create a logical volume of at least that size.

You can use uppercase and lowercase letters as follows:

```
    B/b       512 byte blocks
    K/k       KB
    M/m       MB
    G/g       GB
```

Files

Item	Description
/usr/sbin/	Directory where the **extendlv** command resides.

extendvg Command

Purpose

Adds physical volumes to a volume group.

Syntax

extendvg [**-f**] [**-p** *mirrorpool*] *volumegroup physicalvolume ...*

Description

The **extendvg** command increases the size of the *volumegroup* by adding one or more *physicalvolumes*.

The physical volume is checked to verify that it is not already in another volume group. If the system believes the physical volume belongs to a volume group that is varied on, it exits. But if the system detects a description area from a volume group that is not varied on, it prompts the user for confirmation in continuing with the command. The previous contents of the physical volume are lost, so the user must be cautious when using the override function.

Note: To use this command, you must either have root user authority or be a member of the **system** group.

For volume groups created prior to AIX 5.3, or for volume groups created on AIX 5.3 but varied on with the **varyonvg -M** flag, the **extendvg** will fail if the physical volume has a max transfer size that is smaller than the logical track group size of the volume group. For volume groups created on AIX 5.3 and varied on without the **varyonvg -M** flag, **extendvg** will dynamically lower the logical track group size of the volume group if the physical volume has a max transfer size that is smaller than the logical track group size of the volume group.

Note: The **extendvg** command is not allowed on a snapshot volume group.

You can use the System Management Interface Tool (SMIT) **smit extendvg** fast path to run this command.

Note: This command will fail to add a disk to the volume group if the disk indicates that it is managed by a third party volume manager. To override and clear the disk of the third party volume manger use **chpv -C** *HDiskName*.

Note: When extending a concurrent Volume Group (VG), you must first ensure that each new disk to be added to the VG has a Physical Volume Identifier (PVID) assigned, and that the PVID stored in the

Object Data Manager (ODM) is the same one on every node. When using the Cluster Single Point of Control (C-SPOC) utility to extend the VG, this check is done automatically.

Note: The VG is checked to determine if an existing PV type restriction is in place. If such a restriction exists, the physical volume(s) list on the **extendvg** command line are examined to ensure that they meet the restriction. If one or more of the disks is found to not meet the PV type restriction, the command will fail.

Note: You cannot mix physical volume (PV) of 4 KB block size with PV blocks of other sizes. The block size of all PVs in the volume group must be the same.

Flags

Item	Description
-f	Forces the physical volume to be added to the specified volume group unless it is a member of another volume group in the Device Configuration Database or of a volume group that is active.
-p *mirrorpool*	Assigns each of the physical volumes being added to the specified mirror pool. After mirror pools are enabled in a volume group, the volume group can no longer be imported into a version of AIX that does not support mirror pools.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To add physical volumes hdisk3 and hdisk8 to volume group vg3, enter:

```
extendvg vg3 hdisk3 hdisk8
```

Note: The volume group must be varied on before extending.

Restrictions

The **extendvg** command cannot be run on a snapshot volume group.

Files

Item	Description
/usr/sbin/extendvg	Contains the **extendvg** command.

f

The following AIX commands begin with the letter f.

f Command

Purpose

Shows user information. This command is the same as the **finger** command.

Syntax

{ **f** | **finger** } [[**-b**] [**-h**] [**-l**] [**-p**]] | [**-i**] [**-q**] [**-s**] [**-w**]]

[**-f**] [**-m**] [*User* | *User* **@***Host* | **@***Host*]

Description

The **/usr/bin/f** command displays information about the users currently logged in to a host. The format of the output varies with the options for the information presented.

Default Format

The default format includes the following items:

- Login name
- Full user name
- Terminal name
- Write status (an * (asterisk) before the terminal name indicates that write permission is denied)

For each user on the host, the default information list also includes, if known, the following items:

- Idle time (Idle time is minutes if it is a single integer, hours and minutes if a : (colon) is present, or days and hours if a "d" is present.)
- Login time
- Site-specific information

  The site-specific information is retrieved from the gecos field in the **/etc/passwd** file. The gecos field may contain the Full user name followed by a comma. All information that follows the comma is displayed by the finger command with the Site-specific information.

Longer Format

A longer format is used by the **f** command whenever a list of user's names is given. (Account names as well as first and last names of users are accepted.) This format is multiline, and includes all the information described above along with the following:

- User's **$HOME** directory
- User's login shell
- Contents of the **.plan** file in the user's **$HOME** directory
- Contents of the **.project** file in the user's **$HOME** directory

The **f** command may also be used to look up users on a remote system. The format is to specify the user as *User@Host*. If you omit the user name, the **f** command provides the standard format listing on the remote system.

Create the **.plan** and **.project** files using your favorite text editor and place the files in your **$HOME** directory. The **f** command uses the **toascii** subroutine to convert characters outside the normal ASCII

character range when displaying the contents of the **.plan** and **.project** files. The **f** command displays a `M-` before each converted character.

When you specify users with the *User* parameter, you can specify either the user's first name, last name, or account name. When you specify users, the **f** command, at the specified host, returns information about those users only in long format.

For other information about the **f** command, see "Installation of TCP/IP" in *Networks and communication management*.

Flags

Item	Description
-b	Gives a brief, long-form listing.
-f	Suppresses printing of header line on output (the first line that defines the fields that are being displayed).
-h	Suppresses printing of **.project** files on long and brief long formats.
-i	Gives a quick listing with idle times.
-l	Gives a long-form listing.
-m	Assumes that the *User* parameter specifies a user ID (used for discretionary access control), *not* a user login name.
-p	Suppresses printing of **.plan** files on long-form and brief long-form formats.
-q	Gives a quick listing.
-s	Gives a short format list.
-w	Gives a narrow, short-format list.

Parameters

Item	Description
@Host	Specifies all logged-in users on the remote host.
User	Specifies a local user ID (used for discretionary access control) or local user login name, as specified in the **/etc/passwd** file.
User@Host	Specifies a user ID on the remote host, displayed in long format.

Examples

1. To get information about all users logged in to host `alcatraz`, enter:

   ```
   f @alcatraz
   ```

   Information similar to the following is displayed:

   ```
   [alcatraz.austin.ibm.com]
   Login     Name        TTY Idle         When      Site Info
   brown    Bob Brown   console   2d   Mar 15 13:19
   smith    Susan Smith    pts0  11:   Mar 15 13:01
   jones    Joe Jones      tty0    3   Mar 15 13:01
   ```

   User `brown` is logged in at the `console`, user `smith` is logged in from pseudo teletype line `pts0`, and user `jones` is logged in from `tty0`.

2. To get information about user brown at alcatraz, enter:

```
f brown@alcatraz
```

Information similar to the following is displayed:

```
Login name: brown
Directory: /home/brown    Shell: /home/bin/xinit -L -n Startup
On since May 8 07:13:49 on console
No Plan.
```

3. To get information about user brown at a local host in short form, enter:

```
f -q brown
```

Information similar to the following is displayed:

```
Login           TTY             When
brown           pts/6           Mon Dec 17 10:58
```

Files

Item	Description
/usr/bin/f	Contains the **f** command.
/etc/utmp	Contains list of users currently logged in.
/etc/passwd	Defines user accounts, names, and home directories.
/etc/security/passwd	Defines user passwords.
/var/adm/lastlog	Contains last login times.
$HOME/.plan	Optional file that contains a one-line description of a user's plan.
$HOME/.project	Optional file that contains a user's project assignment.

factor Command

Purpose

Factors a number.

Syntax

factor [*Number*]

Description

When called without specifying a value for the *Number* parameter, the **factor** command waits for you to enter a positive number less than 1E14 (100,000,000,000,000). It then writes the prime factors of that number to standard output. It displays each factor the proper number of times. To exit, enter 0 or any nonnumeric character.

When called with an argument, the **factor** command determines the prime factors of the *Number* parameter, writes the results to standard output, and exits.

Examples

To calculate the prime factors of 123, enter:

```
factor 123
```

The following is displayed:

```
123
     3
     41
```

Files

Item	Description
/usr/bin/factor	Contains the **factor** command.

true or false Command

Purpose

Returns an exit value of zero (true) or a nonzero exit value (false).

Syntax

true

false

Description

The **true** command returns a zero exit value. The **false** command returns a nonzero exit value. These commands are most often used as part of a shell script.

Examples

To construct a loop that displays the date and time once each minute, use the following code in a shell script:

```
while true
do
   date
   sleep 60
done
```

reboot or fastboot Command

Purpose

Restarts the system.

Syntax

{ **reboot** | **fastboot** } [**-l**] [**-n**] [**-q**] [**-t** *mmddHHMM* [*yy*]]

Description

The **reboot** command can be used to perform a reboot operation if no other users are logged into the system. The **lsattr** command and enter `lsattr -D -l sys0`. The default value is **true**. To reset the autorestart attribute value to **false**, use the **/var/adm/wtmp**, the login accounting file. These actions are inhibited if the **-l**, **-n**, or **-q** flags are present.

The **fastboot** command restarts the system by calling the **reboot** command. The **fsck** command runs during system startup to check file systems. This command provides BSD compatibility.

Flags

Item	Description
-l	Does not log the reboot or place a shutdown record in the accounting file. The **-l** flag does not suppress accounting file update. The **-n** and **-q** flags imply **-l**.
-n	Does not perform the **sync** command. Use of this flag can cause file system damage.
-q	Restarts without first shutting down running processes. **Note:** A file system synchronization will not occur if the **-q** flag is used. If you want the file system to be synchronized, manually run the **sync** command or use the **shutdown -r** command.
-t	Shuts down the system immediately and then restarts the system on the specified date. A valid date has the following format: *mmddHHMM* [*yy*] where: ***mm*** Specifies the month. ***dd*** Specifies the day. ***HH*** Specifies the hour. ***MM*** Specifies the minute. ***yy*** Specifies the year (optional). The two digit value represents the value of the year in the current century (based on the system time). For example, if the current year based on the systems time is 1985, 99 means 1999 and if the current year is 2005 then 99 means 2099 and 04 means 2004.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To shut down the system without logging the reboot, enter:

```
reboot  -l
```

Files

Item	Description
/etc/rc	Specifies the system startup script.
/var/adm/wtmp	Specifies login accounting file.

fc Command

Purpose

Processes the command history list.

Syntax

To Open an Editor to Modify and Reexecute Previously Entered Commands

fc [**-r**] [**-e** *Editor*] [*First* [*Last*]]

To Generate a Listing of Previously Entered Commands

fc -l [**-n**] [**-r**] [*First* [*Last*]]

To Generate a Listing of Previously Entered Commands with Time of Execution

fc -t [**-n**] [**-r**] [*First* [*Last*]]

To Re-execute a Previously Entered Command

fc -s [*Old*=*New*] [*First*]

Description

The **fc** command displays the contents of your command history file or invokes an editor to modify and reexecutes commands previously entered in the shell.

The command history file lists commands by number. The first number in the list is selected arbitrarily. The relationship of a number to its command does not change except when the user logs in and no other process is accessing the list. In that case, the system resets the numbering to start the oldest retained command at 1.

If the numbers in the command history file reach a limit greater than the value of the **HISTSIZE** environment variable or 32767, whichever is greater, the shell wraps to 1. Despite this optional number wrapping, the **fc** command maintains the time-ordering sequence of the commands. For example, if three commands in sequence are given the numbers 32766, 32767, and 1 (wrapped), command 32767 is still considered previous to command 1.

The commands in the history file can be displayed using the **-l** (lowercase L) flag. When the **-l** flag is not specified and commands are edited using the **-e** *Editor* flag, the resulting lines are entered at the end of the history file and then reexecuted by the shell (the **fc -e** *Editor* command is not entered into the command history list). If the editor returns a non-zero exit status, this suppresses entry in the history file and command reexecution.

Any command-line variable assignments or redirection operators used with the **fc** command again invoke the previous command, suppressing standard error for both the **fc** command and the previous command. For example:

```
fc -s -- -1 2>/dev/null
```

Flags

Item	Description
-e *Editor*	Edits commands using the specified editor. The *Editor* parameter should be a command name. The command is located using the **PATH** environment variable. The value in the **FCEDIT** environment variable is used as a default when the **-e** flag is not specified. If the **FCEDIT** environment variable is null or unset, the ed editor is used.

Item	Description
-l	(lowercase L) Lists the commands in your history file. No editor is invoked to modify them. The commands are written in the sequence indicated by the *First* and *Last* parameters, as affected by the **-r** flag, with each command preceded by the command number.
-n	Suppresses command numbers when used with the **-l** flag.
-r	Reverses the order of the commands listed (when used with the **-l** flag) or reverses the order of the commands edited (when the **-l** flag is not specified).
-s	Reexecutes a command without invoking an editor. If the *First* parameter is not also specified, the **-s** flag re-executes the previous command.
-t	Lists the commands in your history file along with there time of execution. The working is similar to **-l** flag but the time of execution of the command is displayed. **Note:** If the time field is recorded previously by setting `EXTENDED_HISTORY=ON`, then formatted time field is displayed, else "?".

Parameters

Item	Description
First or *Last*	Selects the commands to list or edit. The number of previous commands that can be accessed is determined by the value of the **HISTSIZE** environment variable. The *First* and *Last* parameters must have one of the following values: **[+] *Number*** Represents a specific command number. Command numbers can be displayed with the **-l** flag. A + (plus sign) is the default. ***-Number*** Represents a command that was previously executed, specified by the number of commands to back up in the history list. For example, -1 indicates the immediately previous command. ***String*** Indicates the most recently entered command that begins with the specified string. If the *Old*=*New* parameter is specified without the **-s** flag, the string from the *First* parameter cannot contain an embedded **=** (equal sign). When using the **-s** flag, omission of the *First* parameter causes the previous command to be used.

When the **-s** flag is not specified, the following rules apply:

- When using the **-l** flag, omission of the *Last* parameter causes a default to the previous command.
- When using the **-r**, **-n**, and **-e** flags, omission of the *Last* parameter causes a default to the *First* parameter.
- If both the *First* and *Last* parameters are omitted, the previous 16 commands are listed or the previous single command is edited (depending on whether or not the **-l** flag is used).
- If both the *First* and *Last* parameters are present, all commands are listed (when the **-l** flag is specified) or edited (when the **-l** flag is not specified). Editing multiple commands is accomplished by presenting to the editor all the commands at one time, each command starting on a new line. If the *First* parameter represents a newer command than the *Last* parameter, the commands are listed or edited in reverse sequence. This is equivalent to using the **-r** flag. For example, the following commands on the first line are equivalent to the corresponding commands on the second line:

```
fc  -r  10  20          fc      30  40
fc      20  10          fc  -r  40  30
```

- When a range of commands is used, it is not an error to specify *First* or *Last* values that are not in the history list. The **fc** command substitutes the value representing the oldest or newest command in the list, as appropriate. For example, if there are only ten commands in the history list, numbered 1 to 10, the commands:

```
fc  -l
fc   1   99
```

  list and edit, respectively, all ten commands.

Item	Description
Old=*New*	In commands to be reexecuted, replaces the fist occurrence of the old string with the new string.

Environment Variables

The following environment variables affect the execution of the **fc** command:

Item	Description
EXTENDED_HISTORY	Used to control the recording of time of command execution in the history file. If the variable is set to ON then the time is recorded, otherwise, it is not recorded.
FCEDIT	When expanded by the shell, determines the default value for the **-e** *editor* variable. If the **FCEDIT** environment variable is null or is not set, the ed editor is the default.
HISTDATEFMT	This is used to control the format of the time displayed by the **fc –t** command. For example, if HISTDATEFMT=%Y, then **fc -t** will display the year when the command is executed. The formatting is similar to that done by **date** command.
HISTFILE	Determines the path name of the command history file. If the **HISTFILE** environment variable is not set, the shell may attempt to access or create the **.sh_history** file in the user's home directory.
HISTSIZE	Determines a decimal number representing the limit to the number of previous commands that are accessible. If this variable is not set, a default value of128 is used.

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion of the listing.
>0	An error occurred.

Otherwise, the exit status is that of the commands executed by the **fc** command.

Examples

1. To invoke the editor defined by the **FCEDIT** environment variable on the most recent command (the default editor is **/usr/bin/ed**), enter:

```
fc
```

   The command is executed when you finish editing.

2. To list the previous two commands that were executed, enter:

```
fc -l -2
```

3. To find the command that starts with `cc`, change `foo` to `bar`, and display and execute the command, enter:

```
fc -s foo=bar cc
```

4. To list the previously executed commands along with there time of execution, type:

```
fc –t
```

Files

Item	Description
/usr/bin/ksh	Contains the Korn shell **fc** built-in command.
/usr/bin/fc	Contains the **fc** command.

fccheck Command

Purpose

Performs basic problem determination on the First Failure Data Capture (FFDC) utilities.

Syntax

`/opt/rsct/bin/fccheck [ -q ] | [ -h ]`

Description

fccheck performs basic problem determination for the First Failure Data Capture utilities. The command checks for the following conditions and information on the local node:

- Checks if FFDC Error Stack usage has been disabled in the current process environment.
- Obtains the IP address that would be currently used by FFDC to identify the local node.
- Checks if **/var/adm/ffdc/stacks** is available, and if so, how much space is available in the file system where the directory resides. Checks to see if there is insufficient space to create FFDC Error Stacks.
- Checks if **/var/adm/ffdc/dumps** is available, and if so, how much space is available in the file system where the directory resides.

Results of these tests are displayed to standard output unless the "quiet" option has been specified. **fccheck** sets an exist status value to indicate the most severe condition it detected during the execution of its tests.

Flags

-h
Displays help and usage information to standard output. No other processing is performed.

-q
Specified "quiet" mode. The command does not display the results of each test to standard output. The exit status of the command must be used to determine the results of the tests. If more than one condition was detected, the exit status will reflect the most severe condition detected by **fccheck**.

Exit Status

The following integer exit status codes can be generated by this command:

0
All conditions tested by **fccheck** were found to be in normal operational parameters.

2
Help information successfully displayed. No further processing is performed.

12
No checking performed. Invalid option specified to this command.

19
The directory **/var/adm/ffdc/stacks** is not mounted or does not exist.

20
Cannot access or examine one or more directories in the path **/var/adm/ffdc/stacks**. Permissions may have been changed on one or more of the directories in this path to prevent access.

24
Cannot access or examine one or more directories in the path **/var/adm/ffdc/dumps**. Permissions may have been changed on one or more of the directories in this path to prevent access.

32
The directory **/var/adm/ffdc/dumps** is not mounted or does not exist.

40
Insufficient space is available in the **/var/adm/ffdc/stacks** directory to create FFDC Error Stacks on the local node.

41
Unable to obtain file system information from the operating system. This indicates a potential problem with the operating system itself.

42
FFDC Error Stack creation and usage has been disabled in this process environment.

Examples

To check for possible problems with the FFDC utilities on the local node:

```
fccheck
fccheck Status: All tests completed
```

If the local node had disabled the creation of FFDC Error Stacks, `fccheck` would indicate this as a problem:

```
fccheck

fccheck Status: Creation and use of FFDC Error Stacks has been expressly
disabled in the current execution environment. Any processes created in
the current execution environment cannot create their own FFDC Error Stacks
or inherit use of existing FFDC Error Stacks.

fccheck Status: All checks completed. Examine the previous status output for
possible FFDC problem conditions and take the recommended actions listed in
these messages.
```

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

fcclear Command

Purpose

Removes FFDC Error Stacks and detail data files from the local node.

Syntax

`/opt/rsct/bin/fcclear -h` | [`-d` *filename* [,*filename*,...]] [`-D` *filename* [,*filename*,...]] [`-f` *FFDC_Failure_ID* [,*FFDC_Failure_ID*,...]] [`-F` *FFDC_Failure_ID* [,*FFDC_Failure_ID*,...]] [`-s` *file_name*[,*filename*,...]] [`-S` *file_name* [,*filename*,...]] [`-t` *days*]]

Description

fcclear is used to remove FFDC Error Stack files that are no longer needed for problem determination efforts from the local node. Specific FFDC Error Stack files can be removed, as well as FFDC Error Stack files containing the records of specific FFDC Failure Identifiers. Individual entries within an FFDC Error Stack cannot be removed.

Using the **-t** option, **fcclear** can be used to remove FFDC Error Stack files older than a specific number of days. To use **fcclear** in an automatic fashion to clean out unneeded FFDC Error Stacks, see the **cron** command for automating the execution of commands.

To remove all FFDC Error Stacks from the local node, specify a value of zero (0) for the number of days option argument.

Flags

-d
Removes detail data files by specifying a list of one or more detail data file names. These file names may be absolute path names, or relative to the **/var/adm/ffdc/dumps** directory. These files are removed if they exist on the local node. Files on remote nodes cannot be removed through this command. If more than one file name is provided, they must be separated by a comma (,) without any intervening white space.

-D
Preserves detail data files by specifying a list of one or more detail data file names. These file names may be absolute path names, or relative to the **/var/adm/ffdc/dumps** directory. These files are retained if they exist on the local node. Files on remote nodes cannot be retained through this command. If more than one file name is provided, they must be separated by a comma (,) without any intervening white space.

-f
Removes FFDC Error Stack files by specifying a list of one or more FFDC Failure Identifiers. The FFDC Error Stacks associated with these FFDC Error Identifiers are located and removed if they are present on the local node. FFDC Error Stacks on remote nodes will not be removed. If more than one FFDC Failure Identifier is supplied, they must be separated by a comma (,) with no intervening white space.

-F
Preserves FFDC Error Stack files by specifying a list of one or more FFDC Failure Identifiers. The FFDC Error Stacks associated with these FFDC Error Identifiers are located and retained if they are present on the local node. FFDC Error Stacks on remote nodes will not be retained. If more than one FFDC Failure Identifier is supplied, they must be separated by a comma (,) with no intervening white space.

-h
Displays help and usage information to the standard output device. No other processing is performed.

-s
Removes FFDC Error Stack files by specifying a list of one or more FFDC Error Stack file names. These file names can be absolute path names or file names relative to the **/var/adm/ffdc/stacks** directory. These files are removed if they exist on the local node. FFDC Error Stacks on remote nodes cannot be

removed through this command. If more than one file name is provided, each must be separated by a comma (,) without any intervening white space.

-S
Removes FFDC Error Stack files by specifying a list of one or more FFDC Error Stack file names. These file names can be absolute path names or file names relative to the **/var/adm/ffdc/stacks** directory. These files are removed if they exist on the local node. FFDC Error Stacks on remote nodes cannot be removed through this command. If more than one file name is provided, each must be separated by a comma (,) without any intervening white space.

-t
Indicates that FFDC Error Stacks and detail data files that are older than a specific number of days should be removed from the local node. This selection criteria is independent of the other selection criteria.

Exit Status

fcclear generates the following exit status values upon completion:

0
Successful completion of the command. The command may complete successfully if no FFDC Error Stack files or detail data files match the selection criteria.

2
Help information successfully displayed. No further processing is performed.

10
No files are removed from the local system. A required option was not specified to this command.

11
No files are removed from the local system. The argument of the -t option is not numeric.

12
No files are removed from the local system. Unknown option specified by the caller.

19
The directory **/var/adm/ffdc/stacks** does not exist or is not mounted.

26
No files are removed from the local system. The same option was specified more than once.

28
No files were removed from the system. The caller provided options that instruct the command to both remove and retain the same file. This condition can occur when the command user specified an FFDC Failure Identifier that is recorded in an FFDC Error Stack file specified by name to this command.

Examples

To remove any FFDC Error Stack and detail data files older than seven days from the local node:

```
fcclear -t 7
```

To remove all FFDC Error Stack and detail data files older than seven days, but retain the FFDC Error Stack that contains information for the FFDC Failure Identifier **/3Iv04ZVVfvp.wtY0xRXQ7.....**, issue the command:

```
fcclear -t 7 -F /3Iv04ZVVfvp.wtY0xRXQ7...................
```

To remove the FFDC Error Stack file that contains the record for the FFDC Failure Identifier **/3Iv04ZVVfvp.wtY0xRXQ7.....**, issue the command:

```
fcclear -f /3Iv04ZVVfvp.wtY0xRXQ7...................
```

To remove the FFDC Error Stack files **myprog.14528.19990204134809** and **a.out.5134.19990130093256** from the system, plus the detail data file **myprog.14528.19990204135227**:

```
fcclear -s myprog.14528.19990204134809,a.out.5134.19990130093256
    -d myprog.14528.19990204135227
```

To extend the previous command to remove the named files plus any FFDC Error Stack and detail data files older that 14 days:

```
fcclear -s myprog.14528.19990204134809,a.out.5134.19990130093256
    -d myprog.14528.19990204135227 -t 14
```

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

fcdecode Command

Purpose

Translates a First Failure Data Capture (FFDC) Failure Identifier from its standard form into its component parts, displaying this information to the standard output device in human readable format.

Syntax

`/opt/rsct/bin/fcdecode` *FFDC_Failure_ID* [,*FFDC_Failure_ID*,...] | `-h`

Description

fcdecode decodes the 42-character FFDC Failure Identifier into its component parts, and displays these parts in human readable format. The output of this command displays the following information, extracted from the FFDC Failure Identifier:

- The network address (in ASCII format) of the node where this report resides
- The time when this recording was made, expressed using the currently active time zone settings
- One of the following, depending on where the information is recorded:
- The AIX Error Log template ID used to make this recording, if the record was filed in the AIX Error Log on that node, or
- The name of the FFDC Error Stack file containing this recording, if the record was file in the FFDC Error Stack and the FFDC Error Stack resides on this node
- A suggested command that can be used to obtain the specific report associated with this FFDC Failure Identifier.

Flags

-h

Displays a help message to standard output and exits. No other processing is performed, regardless of the options specified.

Parameters

FFDC_Failure_ID

An FFDC Failure Identifier, returned from previous calls to the **fcpushstk** and **fclogerr** commands, or returned from previous calls to the **fc_push_stack** or **fc_log_error** subroutines. This identifier

indicates an entry made to report a failure or other noteworthy incident. More than one FFDC Failure Identifier can be provided as an argument to this command, however, each identifier must be separated by a comma (,) with no intervening white space between the identifiers.

Exit Status

fcdecode returns one of the following integer status codes upon completion:

0
FFDC Failure Identifier successfully decoded.

2
Help information displayed and processing ended.

10
An FFDC Failure Identifier was not provided as an argument to this command.

12
Invalid or unsupported option provided to this command.

27
No information written to the standard output device. The FFDC Failure Identifier argument was not valid.

Examples

The FFDC Failure Identifier is represented by a base-64 value, read from right to left. Each dot represents a leading zero. To decode the FFDC Failure Identifier **.3Iv04ZVVfvp.wtY0xRXQ7.**.....into its component parts:

```
fcdecode .3Iv04ZVVfvp.wtY0xRXQ7.....................

Information for First Failure Data Capture identifier
        .3Iv04ZVVfvp.wtY0xRXQ7.....................
        Generated by the local system
        Generated Thu Sep 3 11:40:17 1998 EDT
        Recorded to the AIX Error Log using template 460bb505
        To obtain the AIX Error Log information for this entry, issue
        the following command on the local system:
        TZ=EST5EDT errpt -a -j 460bb505 -s 0903114098 | more
        Search this output for an AIX Error Log entry that contains
        the following ERROR ID code:
         .3Iv04ZVVfvp.wtY0xRXQ7.....................
```

The same command run on a different node has the following results:

```
fcdecode .3Iv04ZVVfvp.wtY0xRXQ7.....................

Information for First Failure Data Capture identifier
        .3Iv04ZVVfvp.wtY0xRXQ7.....................
        Generated on a remote system with the following Internet address:
          9.114.55.125
        Generated Thu Sep 3 11:40:17 1998 EDT
        Recorded to the AIX Error Log using template 460bb505
        TZ=EST5EDT errpt -a -j 460bb505 -s 0903114098 | more
        Search this output for an AIX Error Log entry that contains
        the following ERROR ID code:
         .3Iv04ZVVfvp.wtY0xRXQ7.....................
```

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

fcdispfid Command

Purpose

Displays the First Failure Data Capture Failure Identifier (FFDC Failure Identifier) to the standard error device.

Syntax

`/opt/rsct/bin/fcdispfid` [`-q`]*FFDC_Failure_ID* | `-h`

Description

This command is used by scripts to display an FFDC Failure Identifier value to the standard error device. This interface is provided because script programs do not have a mechanism for passing data back to its client except through exit status codes, signals, standard output, and standard error. To accomplish the task of "passing back" an FFDC Failure Identifier to a client in such an environment, **fcdispfid** uses XPG/4 cataloged message number **2615-000** to display this information to the standard error device. Clients of the script can capture the standard error information, search for the specific message number, and obtain the FFDC Failure Identifier from the script.

The script must indicate that any FFDC Failure Identifiers generated by the script will be directed to the standard error device in the script's user documentation. The client cannot be expected to know this behavior by default.

Flags

-h
Displays a help message to standard output and exits. No other processing is performed, regardless of the options specified.

-q
Suppresses warning messages from this command. If this option is not provided, this command will display messages when an invalid FFDC Failure Identifier is detected.

Parameters

FFDC_Failure_ID

Specifies an FFDC Failure Identifier. This is an identifier returned from a previous call to **fcpushstk** or **fclogerr**, and indicates an entry made to report a failure encountered by the script. This identifier is written to the standard error device using FFDC message **2615-000**.

Exit Status

0
FFDC Failure Identifier displayed to standard error.

2
Help information displayed and processing ended.

12
No information written to the standard error device. An invalid option was specified.

27
No information written to the standard error device. The *FFDC_Failure_ID* argument does not appear to be in a valid format.

Examples

To display an FFDC Failure Identifier to the client through the standard output device:

```
   FID=$(fclogerr -e FFDC_ERROR -t ERRID_SP_FFDCEXMPL_ER -i /usr/lpp/ssp/inc/
myprog.h -r myprog -s myprog.ksh -p $LINEPOS -v "1.1" -l PSSP -d $MINUSDOPTS -x
$MINUSXOPTS -y $MINUSYOPTS -b "myprog Configuration Failure - Exiting")
    RC=$?
    if ((RC == 0))
    then
         fcdispfid $FID
         return 1
    else
         :
    fi
```

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

fcfilter Command

Purpose

Locates and displays any First Failure Data Capture (FFDC) Failure Identifiers in a file or in standard input. More than one file may be specified.

Syntax

`/opt/rsct/bin/fcfilter` [*file_name*] [. . .]

Description

This commands scans any files listed as arguments for First Failure Data Capture (FFDC) Failure Identifiers. If a file name is not provided as an argument, this command examines standard input for FFDC Failure Identifiers. If an FFDC Failure Identifier is detected, **fcfilter** displays the identifier to standard output on its own line.

fcfilter can be used by scripts to extract FFDC Failure Identifiers returned by child processes via the standard error device.

If **fcfilter** detects more than one FFDC Failure Identifier in the input, the command will display all FFDC Failure Identifiers found, each one on a separate output line.

Parameters

file_name

The name of the file to be searched for an FFDC Failure Identifier. More than one file may be provided. If a file name is not provided, **fcfilter** reads from standard input.

Exit Status

fcfilter returns the following integer status codes upon completion:

0
fcfilter completed its execution. This exit status does not necessarily mean that any FFDC Failure Identifiers were detected.

> 0
fcfilter was interrupted or stopped by a signal. The exit status is the integer value of the signal that stopped the command.

Examples

The FFDC Failure Identifier is represented by a base-64 value, read from right to left. Each dot represents a leading zero. To obtain the list of all FFDC Failure Identifiers generated by a run of the command *mycmd*:

```
 mycmd 2> /tmp/errout
 fcfilter /tmp/errout
/.00...JMr4r.p9E.xRXQ7...................
/.00...JMr4r.pMx.xRXQ7...................
```

To obtain the FFDC Failure Identifier from a child process in a parent script, the script can use the **fcfilter** command as follows:

```
RESULTS=$(mychild 2> /tmp/errout)
if (($? != 0))              # mychild ended in failure, get FFDC ID
then
    cat /tmp/errout | fcfilter | read FIRST_FFDCID
else
    rm -f /tmp/errout
fi
```

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

fcinit Command

Purpose

Establishes or inherits a First Failure Data Capture execution environment.

Syntax

For Bourne and Korn shells:

`/opt/rsct/bin/fcinit.sh [ [ -l ] [ -s { c | i } ] ] | [ -h ]`

For C shells:

`source /opt/rsct/bin/fcinit.csh [ [ -l ] [ -s { c | i } ] ] | [ -h ]`

Description

This interface must be used by a script program that wishes to use the FFDC interfaces for recording information to the AIX Error Log, the BSD System Log, or the FFDC Error Stack .

Applications may wish to establish an FFDC Environment for one of the following reasons:

- The script may wish to record information to the AIX Error Log. Scripts can use **fcinit** to establish a basic FFDC Environment
- The script wants to have itself and any descendant processes created by itself or its children to record failure information to the FFDC Error Stack. In this case, the script considers itself a "top-level" application that will cause multiple "lower-level" applications to be created, and the success of the "top-level" application depends upon the success of these "lower-level" applications. When using **fcinit** in this fashion, the process is said to *establish* or *create* the FFDC Error Stack Environment.
- The script uses the FFDC Error Stack or the FFDC Trace only in those cases when the script is invoked by an ancestor process that wants failure information or trace information recorded to these devices. In all other cases, the script does not wish to use these devices. When using **fcinit** in this fashion, the process is said to *inherit* the FFDC Error Stack Environment.

Any process wishing to record information to the AIX Error Log or the BSD System Log through the FFDC interfaces must establish an FFDC Environment. If the process does not wish to make use of an FFDC

Error Stack, the process can establish a basic FFDC Environment that does not make use of an FFDC Error Stack. An FFDC Error Stack Environment, which contains an FFDC Error Stack, is established by a process when that process wants to have failure information from itself, any threads it may create, and any descendant processes it may create to be recorded in an FFDC Error Stack. An *FFDC Error Stack Environment*, which contains an FFDC Error Stack, is inherited by a process when that process wants to record failure information to an FFDC Error Stack file only when one of its ancestors has requested for processes to do so; in all other cases, the process will not record failure information to the FFDC Error Stack.

The FFDC Error Stack Environment, which contains an FFDC Error Stack, reserves an FFDC Error Stack file, so that failure information is recorded to a file in the **/var/adm/ffdc/stacks** directory. These files use the naming format ***script_name.PID.date_and_time***, where *script_name* is the name of the script itself, *PID* is the process identifier of the script, and *date_and_time* is the date and time when the script was executed. Whenever this script or children processes of this script record failure information to the FFDC Error Stack, it will be recorded in this file.

In order for information to be recorded in the FFDC Error Stack by a process, the process must use the **fcpushstk** FFDC interface, and the process has to be operating within an established FFDC Error Stack Environment. If an FFDC Error Stack Environment does not exist, or if the **fcpushstk** interface is not used when an FFDC Error Stack Environment exists, no information is recorded by that process in the FFDC Error Stack. This function permits processes to run in a normal or "silent" mode when failure debugging information is not wanted or needed, but also permits this information to be available when the process is invoked within a special environment for debugging.

fcinit must be executed within the FFDC client's process environment ("sourced") in order for the command to properly set the FFDC Environment for the script. Script-based FFDC clients using this command must "source" the command in order for **fcinit** to execute within the client's process image. If this is not done. the FFDC interface is executed within its own process image; any settings of the FFDC Environment are lost after the FFDC interface completes. To demonstrate how a script-based application would "source" the **fcinit** command, a Korn Shell program would issue the following instruction:

```
. fcinit.sh <options and arguments>
```

A C Shell script would do the following:

```
source fcinit.csh <options and arguments>
```

Processes that use the **fclogerr** FFDC interface must establish an *FFDC Environment*. If the process only wishes to use the **fclogerr** interface, the *FFDC Environment* can be established without an FFDC Error Stack.

If an FFDC Environment already exists when a script attempts to create one, the script inherits the existing FFDC Environment instead of creating its own.

Flags

-h
Displays a help message to standard output and exits. No other processing is performed, regardless of the options specified.

-l
Indicates that the process wishes to make use of the AIX Error Log only. This option is not necessary when the **-s** option is specified, since use of the AIX Error Log is permitted within an FFDC Error Stack Environment.

-s
Indicates that an FFDC Error Stack Environment is to be established. Applications wishing to use the **fcpushstk** interface must specify this flag. Upon successful completion of this command, an FFDC Error Stack file is reserved for the script in the **/var/adm/ffdc/stacks** directory. This flag must be specified with one of two possible options:

c

Requests that the FFDC Error Stack Environment be *created*. If an FFDC Error Stack Environment was not created by an ancestor process, it will be created. If such an environment was previously created by an ancestor process, this process will *inherit* the FFDC Error Stack Environment as if the **i** option had been specified.

i

Specifies that an FFDC Error Stack Environment is to be *inherited* if it was previously established by an ancestor process. If an FFDC Error Stack Environment was not previously established by an ancestor process, an FFDC Error Stack Environment is not established for this process, and this process cannot make use of an FFDC Error Stack (although it may make use of the AIX Error Log and the BSD System Log).

Parameters

file_name

The name of the file to be searched for an FFDC Failure Identifier. More than one file may be provided. If a file name is not provided, **fcfilter** reads from standard input.

Exit Status

fcinit returns the following exit status codes upon completion:

0

FFDC Environment successfully established.

1

FFDC Environment successfully inherited.

2

Help information displayed and processing ended.

fcinit returns the following exit status codes upon detection of a failure:

12

FFDC Environment not established or inherited - Unknown function parameter provided.

13

FFDC Error Stack Environment not established or inherited - caller indicated that the FFDC Environment should be both created and inherited.

14

FFDC Environment not established in this call - the caller already has an FFDC Environment established for itself - this routine may have been executed multiple times.

15

FFDC Error Stack Environment not established or inherited - an FFDC Error Stack Environment did not exist, and the FC_INHERIT option was specified.

16

FFDC Environment not established or inherited - the client's process environment could not be modified by this routine.

17

FFDC Environment not established or inherited - the FFDC Environment appears to be corrupted and should be considered unusable.

18

FFDC Environment not established or inherited - the routine could not allocate the memory required to modify the client's process environment.

19

FFDC Error Stack Environment not established or inherited - Unable to reserve the FFDC Error Stack file for the calling process - the FFDC Error Stack directory does not exist or cannot be used.

21
FFDC Error Stack Environment not established or inherited - Unable to reserve the FFDC Error Stack file for the calling process - the file already exists

42
FFDC Error Stack Environment not established or inherited - creation and use of FFDC Error Stacks has been disabled by the system administrator. Scripts can establish only a basic FFDC Environment that makes use of the AIX Error Log and the BSD System Log.

99
FFDC Environment not established or inherited - an unexpected internal failure occurred within **fcinit**. This condition may require the attention of customer and application-support services.

Examples

For a Korn Shell script to establish a basic FFDC Environment for using the AIX Error Log and the BSD System Log only (an FFDC Error Stack is not to be used or reserved):

```
# Set up an FFDC Environment to use the AIX Error Log only. An FFDC Error
# Stack is not needed for this script.
. fcinit.sh -1
rc=$?
if ((rc != 0))
    then
         print "fcinit failed with exit code of $rc"
         exit 1
fi
# Normal processing starts
```

For a Korn Shell script to establish an FFDC Error Stack Environment that causes the script and any descendant process to record failure information to the FFDC Error Stack:

```
# Set up FFDC Environment to record failure information to the FFDC Error
# Stack
. fcinit.sh -sc
rc=$?
if ((rc != 0))
then
    print "fcinit failed with a code of $rc"
    exit 1
fi
# Normal processing starts
```

Note: The FFDC client may receive an indication that an FFDC Error Stack Environment was inherited, instead of created by the **fcinit** call. This occurs when an FFDC Error Stack Environment was already established by one of the process's ancestors.

To inherit an FFDC Error Stack Environment from the process's parent process:

```
# Inherit an FFDC Environment from parent process if it exists - otherwise,
# operate in a normal "silent" mode
. fcinit.sh -si
rc=$?
if ((rc != 0))
then
    print "fcinit failed with a code of $rc"
    exit 1
fi
# Normal processing starts
```

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

fclogerr Command

Purpose

Records information about failure or noteworthy conditions to the AIX error log and the BSD system log.

Syntax

`/opt/rsct/bin/fclogerr` { `-e` *event* `-t` *error_template_label* `-i` *error_template_headerfile* `-r` *resource* `-s` *source_filename* `-p` *line_of_code_pos* `-v` *sidlevel* `-l` *lpp_name* `-a` *assoc_fid* { [`-d` *detail_data_item*[,*detail_data_item*,...] `-x` *detail_data_type*[,*detail_data_type*,...] `-y` *detail_data_len*[,*detail_data_len*,...]] | [`-f` *detail_data_file*] } `-b` *BSD_syslog_message_text* } | `-h`

Description

This interface is used by any script program that wishes to record information to the AIX Error Log and the BSD System Log. The information written to this device is intended for use by the system administrator or operator to determine what failure conditions or other noteworthy conditions have occurred on the system that require attention. The purpose of the AIX Error Log and the BSD System Log is to record enough information about a condition so that the nature, impact, and response to the condition can be determined from the report, without requiring a recreation of the condition to detect what condition occurred and where. Any software that encounters permanent failure conditions that will persist until some type of direct intervention occurs, or encounters a condition that should be brought to the attention of the system administrator, should use **fclogerr** to record this information in the AIX Error Log and the BSD System Log.

Scripts should establish a basic FFDC Environment or an FFDC Error Stack Environment before using **fclogerr**, either by creating or inheriting the environment. **fclogerr** records information to the AIX Error Log and the BSD System Log even if these environments are not established, but the interface will not be capable of generating an FFDC Failure Identifier unless one of these environments exists.

Processes designed to use the FFDC Error Stack can also make use of the **fclogerr** interface, and should make use of it if they encounter conditions that require administrator attention or intervention to resolve.

To ensure proper identification of the condition and the location at which it was encountered, the FFDC Policy recommends that **fclogerr** should be called in-line in the script's source code module and invoked as soon as the condition is detected. **fclogerr** will record source code file name and line of code information to assist in identifying and locating the source code that encountered the condition. **fclogerr** can be invoked by a subroutine or autoloaded routine to record this information if this is necessary, provided that all location information and necessary failure detail information is made available to this external routine. The external recording routine must record the true location where incident was detected.

Although **fclogerr** reports information to both the AIX Error Log and the BSD System Log, different options must be provided to this interface for each recording device. The Detail Data information recorded to the AIX Error Log is not also recorded to the BSD System Log; BSD System Log information is provided through different command options. This may require the **fclogerr** user to duplicate some information in this call.

Flags

-a

Contains the FFDC Failure Identifier for a failure condition reported by software used by this application which causes or influenced the condition being recorded at this time. This identifier should have been returned to this application as part of the software's result indication. The caller provides this identifier here so that the FFDC Error Stack can associate the failure report it is making at this time with the previously recorded failure report. This permits problem investigators to trace the cause of a failure from its various symptoms in this application and others to the root cause in the other

software. If no other software failure is responsible for this condition, or if the other software did not return an FFDC Failure Identifier as part of its result information, this option should be omitted.

-b

Specifies the text message to be written to the BSD System Log.

-d

One or more data items that provides detailed information on the condition, used to provide the Detail Data in the AIX Error Log entry. If details of the information are too lengthy, these details can be written to a file, and the name of that file provided as the *detail_data_file* parameter. If a detail data file name is provided, this option should be omitted. If neither the *detail_data* or the *detail_data_file* parameters are provided or appear valid, null information will be recorded for the detail data in the AIX Error Log.

More than one data item may be provided with this option. Each data item must be separated by commas (,) with no intervening white-space characters. If a data item has imbedded whitespace characters, the data item must be enclosed in double quotes ("). The data items themselves must not contain commas (,), as the command interprets commands a field separators.

This option *must* be accompanied by the **-x** and **-y** options.

-e

Specifies the FFDC Log Event Type. Current valid values are FFDC_EMERG, FFDC_ERROR, FFDC_STATE, FFDC_TRACE, FFDC_RECOV, and FFDC_DEBUG. This code gives a general description of the type of event being logged (emergency condition, permanent condition, informational notification, debugging information, etc.) and the severity of the condition. If this option is not specified, the event type FFDC_DEBUG is assigned to this incident record.

-f

Name of a file containing details about the condition being reported. This option is used when the details are too lengthy to record within the remaining 100 bytes of Detail Data information left to the application by **fclogerr**, or when a utility exists that can analyze the detail information. The contents of this file is copied to the **/var/adm/ffdc/dumps** directory, and the file's new location is recorded as the Detail Data in the AIX Error Log entry.

-h

Displays a help message to standard output and exits. No other processing is performed, regardless of the options specified.

-i

Specifies the absolute path name of the header file (.h) that contains the error logging template identification number that corresponds to the *error_template_label* specified in the **-l** option. This template must also be found in the node's error logging template repository (**/var/adm/ras/errtmplt**). This header file was generated by the **errupdate** command as part of the source code's building procedures, and should have been included in the LPP's packaging to be installed on the node with the software. If this option is not specified or the header file cannot be found when the script is executed, **fclogerr** will record the failure information using its own default error template (label FFDC_DEF_TPLT_TR, identifier code 2B4F5CAB).

-l

Specifies an abbreviation of the name of the licensed programming product in which this software was shipped. This value should be recognizable to both customer and application-support services as an acceptable name for the LPP. Examples of such values are: PSSP, GPFS, LoadLeveler®, and RSCT. If this option is not provided or appears invalid, the character string **PPS_PRODUCT** is used.

-p

Specifies the line of code location within the source code module where the condition is being reported. The value provided must be a valid integer value. To allow for proper identification and location of the condition, this value should be as close to the line of code that detected the condition as possible. Korn Shell scripts can use the value of **$LINENO**. Script languages that do not provide a special line count variable can provide a symbolic value here that a developer can use to locate the spot in the source code where **fclogerr** is being used. If this option is not valid or not provided, the value of **0** is used.

-q
Suppresses the generation of warning messages from the command. Warning are generated when the command must substitute default information for missing information, or when the command is unable to copy the *detail_data_file* to the **/var/adm/ffdc/dumps** directory.

-r
Specifies the software component name. This is a symbolic name for the software making the report and should be a name recognizable to both customer and application-support services. The character string is limited to 16 characters.

-s
Specifies the name of the source file containing the line of code that encountered the condition being reported. For Korn and Borne Shell scripts, the argument to this option should be set to **$0**; C Shell scripts would set this argument to **${0}**. If this option is not provided or not valid, the character string **unknown_file** is used.

-t
Indicates the symbolic label given to the AIX Error Logging template in the error log repository. The **errupdate** command that builds error logging templates creates a macro that maps this label to an integer code. This label begins with the characters **ERRID_** and is a maximum of 19 characters. If this option is not specified or the header file cannot be found when the script is executed, **fclogerr** will invoke the **errlogger** to create a message in the AIX Error Log using the OPMSG template.

-v
Indicates the SCCS version number of the source code module that detected the condition being recorded. For source code built under SCCS control, this should be set to **"1.1"** (the double-quotes are necessary). If this option is not provided or is not valid, the character string **unknown** is used.

-x
Indicates how the data items specified by the **-d** option are to be interpreted when recording this information to the AIX Error Log. These types must agree with the corresponding fields of the AIX Error Logging template specified in the **-t** option. Each type indicates how the corresponding data item in the **-d** list is interpreted. Acceptable values for this option are ALPHA, HEX, and DEC. There must be a matching type listed in the **-x** argument for each argument in the **-d** list.

This option *must* be supplied if the **-d** option is provided.

-y
Indicates the length of the data items (in bytes) specified by the **-d** option. These lengths must agree with the corresponding fields of the AIX Error Logging template specified in the **-t** option. There must be a matching type listed in the **-y** argument for each argument in the **-d** list.

This option *must* be supplied if the **-d** option is provided.

Parameters

file_name
The name of the file to be searched for an FFDC Failure Identifier. More than one file may be provided. If a file name is not provided, **fcfilter** reads from standard input.

Exit Status

fclogerr returns the following exit status codes upon successful completion:

0
Information successfully queued to be written to the AIX Error Log and the BSD System Log. An FFDC Failure Identifier for the record is displayed to standard output. The caller should capture standard output to obtain this value.

2
Help information displayed and processing ended.

12
No information recorded to the AIX Error Log, and no FFDC Failure Identifier is provided by the command. The command user provided an invalid option to this command.

On AIX platforms other than AIX, **fclogerr** returns the following exit status codes when a failure occurs:

38
A record could not be made int he BSD System Log for this incident. The System Log is experiencing a failure condition. On AIX systems, a report was recorded to the AIX Error Log; on other systems, this should be considered a failure.

When **fclogerr** is provided with incomplete information, it substitutes default information for the missing information and attempts to make a record in the FFDC Error Stack. Warnings are generated in these cases, and warning messages are generated unless the **-q** option is specified. In cases where more than one warning condition was detected, the command returns an exit status code for the condition it considered the most severe. The following exit status codes are returned by **fclogerr** when warning conditions are detected:

10
The command user failed to provide the **-i** option to this command, or the header file named as the argument to the -i option could not be located. The command will record generic information to the AIX Error Log in this case, using the First Failure Data Capture default template (label FFDC_DEF_TPLT_TR, identifier code 2B4F5CAB).

26
Both a detailed data string and a detail data file were provided to this routine. The routine chose the detail data string and ignored the detail data file.

28
The name of the resource detecting the incident was not provided. The default resource name **ffdc** was substituted for the missing resource name.

29
At least one component of the detecting application information—source code file name, source code file version, LPP name, line of code position—was not provided. Default information was substituted for the missing information.

32
The file named in the *detail_data_file* parameter could not be copied to the **/var/adm/ffdc/dumps** directory. The FFDC Error Stack entry cites the original version of this file. Do not discard the original copy of this file.

33
The **-e** option was not specified, or did not specify a valid FFDC event type. The event type FFDC_DEBUG has been assigned to this incident record.

34
A message was not supplied in the *format* parameter. As a result, a generic message was recorded to the BSD System Log for this incident.

35
No detailed information was provided for this incident. Later problem analysis may be difficult without these details to indicate specifics on the incident.

36
The length of the detail data string was greater than the capacity of the AIX Error Log entry limit. Detail data was truncated to fit in the available space. Some information on the incident may have been lost in this truncation.

37
An FFDC Error Identifier could not be constructed for the report created by this routine. An FFDC Failure Identifier is not written to standard output, but information on the incident was recorded to the AIX Error Log and the BSD System Log.

38
A record could not be made in the BSD System Log for this incident. The System Log may not be enabled, or may be experiencing problems. On AIX systems, a report was recorded to the AIX Error Log; on other systems, this should be considered a failure.

Examples

For this example, a Korn Shell script attempts to access configuration information from a file. If this attempt fails, the code will record a failure to the AIX Error Log using the following template source code:

```
*! mymesgcat.cat
+ SP_FFDCEXMPL_ER:
    Comment        = "Configuration Failed - Exiting"
    Class          = S
    Log       = true
    Report         = true
    Alert          = false
    Err_Type  = PERM
    Err_Desc  = {3, 10, "CONFIGURATION FAILURE - EXITING"}
    Prob_Causes    = E89B
    User_Causes    = E811
    User_Actions   = 1056
    Fail_Causes    = E906, E915, F072, 108E
    Fail_Actions   = {5, 14, "VERIFY USER HAS CORRECT PERMISSIONS TO ACCESS FILE"},
                {5, 15, "VERIFY CONFIGURATION FILE"}
    Detail_Data    = 46, 00A2, ALPHA
    Detail_Data    = 42, EB2B, ALPHA
    Detail_Data    = 42, 0030, ALPHA
    Detail_Data    = 16, EB00, ALPHA
    Detail_Data    = 16, 0027, ALPHA
    Detail_Data    = 4, 8183, DEC
    Detail_Data    = 4, 8015, DEC
    Detail_Data    = 60, 8172, ALPHA
```

This definition yields the following AIX Error Logging Template:

```
LABEL:            ERRID_SP_FFDCEXMPL_ER
IDENTIFIER:        <calculated by errupdate during source code build>

Date/Time:         <filled in by AIX Error Log subsystem>
Sequence Number:   <filled in by AIX Error Log subsystem>
Machine Id:        <filled in by AIX Error Log subsystem>
Node Id:           <filled in by AIX Error Log subsystem>
Class:             S
Type:              PERM
Resource Name:     <filled in by -r option to fclogerr>

Description
CONFIGURATION FAILURE - EXITING

Probable Causes
COULD NOT ACCESS CONFIGURATION FILE

User Causes
USER CORRUPTED THE CONFIGRATION DATABASE OR METHOD

    Recommended Actions
    RE-CREATE FILE

Failure Causes
COULD NOT ACCESS CONFIGURATION FILE
PERMISSIONS ERROR ACCESSING CONFIGURATION DATABASE
FILE READ ERROR
FILE IS CORRUPT

    Recommended Actions
    VERIFY USER HAS CORRECT PERMISSIONS TO ACCESS FILE
    VERIFY CONFIGURATION FILE

Detail Data
DETECTING MODULE
<filled in by fclogerr options>
ERROR ID
<The FFDC Failure Identifier created by fclogerr>
REFERENCE CODE
```

```
<The -a option value to fclogerr>
FILE NAME
<Must be supplied as part of -d option list to fclogerr>
FUNCTION
<Must be supplied as part of -d option list to fclogerr>
RETURN CODE<Must be supplied as part of -d option list to fclogerr>
ERROR CODE AS DEFINED IN sys/errno.h
<Must be supplied as part of -d option list to fclogerr>
USER ID<Must be supplied as part of -d option list to fclogerr>
```

The first three Detail Data Fields are constructed by the **fclogerr** routine from information passed in the parameters. The remaining Detail Data must be supplied with the **-d** option, and the type of data supplied must be indicated by the **-x** option. The example source code segment below demonstrates how this is done, and how **fclogerr** is invoked to record the information in the AIX Error Log and the BSD System Log.

```
typeset CONFIG_FNAME
typeset INBUF
typeset MINUSDOPTS
typeset MINUSXOPTS
typeset MINUSYOPTS
typeset FID
integer MYCLIENT
integer RC
    :
MYCLIENT=$$
CONFIG_FNAME="/configfile.bin"
exec 3< $CONFIG_FNAME
    :
read -u3 INBUF
RC=$?
if ((RC != 0))
then
    # Create Detail Data Memory Block for AIX Error Log Template
    # Need to know the EXACT structure of the Template to do this correctly.
    #    Field 1 - filled in by fc_log_error
    #    Field 2 - filled in by fc_log_error
    #    Field 3 - filled in by fc_log_error
    #    Field 4 - name of configuration file being used - 16 bytes
    #    Field 5 - name of function call that failed - 16 bytes
    #    Field 6 - return code from failing function - 4 byte integer
    #    Field 7 - errno from failing function call (unused) - 4 byte integer
    #    Field 8 - user ID using this software - remaining space (62 bytes)
    # This source code supplied fields 4 through 8 in the "-d" option, and
    # describes the data types for each in the "-x" option.
    MINUSDOPTS=$CONFIG_FNAME
    MINUSXOPTS="ALPHA"
    MINUSYOPTS="16"
    MINUSDOPTS="$MINUSDOPTS,read"
    MINUSXOPTS="$MINUSXOPTS,ALPHA"
    MINUSYOPTS="$MINUSYOPTS,16"
    MINUSDOPTS="$MINUSDOPTS,$RC"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSDOPTS="$MINUSDOPTS,0"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSDOPTS="$MINUSDOPTS,$MYCLIENT"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,60"
    FID=$(fclogerr -e FFDC_ERROR -t ERRID_SP_FFDCEXMPL_ER -i /usr/lpp/ssp/inc/
myprog.h -r myprog -s myprog.ksh -p $LINEPOS -v "1.1" -l PSSP -d $MINUSDOPTS -x
$MINUSXOPTS -y $MINUSYOPTS -b "myprog Configuration Failure - Exiting")
    RC=$?
    if ((RC == 0))
    then
         fcdispfid $FID
         return 1
    else
         :
    fi
fi
```

Now consider a slight variation on the above example, using the same AIX Error Logging template, but this time using an external command to obtain the configuration data from a file that this source code supplies. The command exits with a non-zero exit status and prints an FFDC Failure Identifier to standard

output if it encounters any failure conditions. Also, to demonstrate the use of double-quotes in the **-d** list, the configuration file will have an embedded space in the name:

```
typeset CONFIG_FNAME
typeset INBUF
typeset MINUSDOPTS
typeset MINUSXOPTS
typeset MINUSYOPTS
typeset FID
typeset OUTPUT
integer MYCLIENT
integer RC
    :
MYCLIENT=$$
CONFIG_FNAME="This is a test"
OUTPUT=$(configdabeast $CONFIG_FNAME)
RC=$?
if ((RC != 0))
then
    # Create Detail Data Memory Block for AIX Error Log Template
    # Need to know the EXACT structure of the Template to do this correctly.
    #    Field 1 - filled in by fc_log_error
    #    Field 2 - filled in by fc_log_error
    #    Field 3 - filled in by fc_log_error
    #    Field 4 - name of configuration file being used - 16 bytes
    #    Field 5 - name of function call that failed - 16 bytes
    #    Field 6 - return code from failing function - 4 byte integer
    #    Field 7 - errno from failing function call (unused) - 4 byte integer
    #    Field 8 - user ID using this software - remaining space (62 bytes)
    # This source code supplied fields 4 through 8 in the "-d" option, and
    # describes the data types for each in the "-x" option.
    MINUSDOPTS="\""$CONFIG_FNAME"\""
    MINUSXOPTS="ALPHA"
    MINUSYOPTS="16"
    MINUSDOPTS="$MINUSDOPTS,configdabeast"
    MINUSXOPTS="$MINUSXOPTS,ALPHA"
    MINUSYOPTS="$MINUSYOPTS,16"
    MINUSDOPTS="$MINUSDOPTS,$RC"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSDOPTS="$MINUSDOPTS,0"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSDOPTS="$MINUSDOPTS,$MYCLIENT"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,60"
    FID=$(fclogerr -e FFDC_ERROR -t ERRID_SP_FFDCEXMPL_ER -i /usr/lpp/ssp/inc/
myprog.h -r myprog -s myprog.ksh -p $LINEPOS -v "1.1" -l PSSP -d $MINUSDOPTS -x
$MINUSXOPTS -y $MINUSYOPTS -a $OUTPUT -b "myprog Configuration Failure - Exiting")
    RC=$?
    if ((RC == 0))
    then
         fcdispfid $FID
         return 1
    else
         :
    fi
fi
```

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

fcpushstk Command

Purpose

Records information about failure or noteworthy conditions to the First Failure Data Capture Error Stack.

Syntax

`/opt/rsct/bin/fcpushstk` { [`-a` *assoc_fid*] `-c` *message_catalog_name* `-m` *message_set* `-n` *message_number* [`-o` *message_param* [,*message_param*,...]] `-l` *lpp_name* `-p` *line_of_code_pos* `-r` *resource* `-s` *source_filename* `-v` *sidlevel* {[`-d` *detail_data*] | [`-f` *detail_data_file*]} } *default_message* | `-h`

Description

fcpushstk is used by scripts to record failure information to the FFDC Error Stack. Scripts record descriptive information and debugging data to the FFDC Error Stack for use in later problem determination efforts.

The FFDC Error Stack is used to help understand failure conditions that occur when multiple related processes or threads are executing together on a node to perform a common task. This device is best applied to an application that creates one or more threads or subprocesses, which in turn, may also create threads or subprocesses themselves. To use the FFDC Error Stack, the script establishes an *FFDC Error Stack Environment* using the **fcinit** interface. After this environment is established, the application and any of its descendants can make use of the FFDC Error Stack.

Not all software applications will establish an FFDC Error Stack Environment. However, these applications may be invoked by other applications or scripts that establish FFDC Error Stack Environments. In these cases, the scripts or applications invoking this software may wish to capture the failure information from this software, to analyze it along with other failure information from other software it invokes to discover any relationships or patterns in the failures. For this reason, software that ordinarily would not make use of the FFDC Error Stack under normal operational conditions should at least support the use of the FFDC Error Stack when it is used by any client invoking the software. This is accomplished by *inheriting* the FFDC Error Stack Environment from the parent process through the **fcinit** interface.

fcpushstk records descriptions and details about noteworthy conditions to the FFDC Error Stack. If an *FFDC Error Stack Environment* has not been established by the script, either by creation or inheritance, **fcpushstk** does not record any information and returns control back to the caller. This action permits the script to run in a normal "silent" mode when debugging information is not requested, but also permits the script to support the use of the FFDC Error Stack when debugging information is requested.

Scripts must make explicit calls to **fcpushstk** to record information to the FFDC Error Stack when an FFDC Error Stack Environment is established. Merely establishing the environment is not enough to result in failure data being recorded. The **fclogerr** command will not make any records to the FFDC Error Stack.

To ensure proper identification of the condition and the location at which it was encountered, **fcpushstk** should be called in-line in the script's source code module, invoked as soon as the condition is detected. **fcpushstk** will record source code file name and line of code information to assist in identifying and locating the source code that encountered the condition. **fcpushstk** can be invoked by a subroutine or autoloaded routine to record this information if this is necessary, provided that all location information and necessary failure detail information is made available to this external routine. The external recording routine must record the true location where the incident was detected.

The maximum size of an FFDC Error Stack entry is given by the FC_STACK_MAX definition in the **<rsct/ct_ffdc.h>** header file. FC_STACK_MAX defines a length in bytes. This value should be used only as a rough guide, since this length includes data that will be used by **fcpushstk** to record the detecting file information, description information, and FFDC Failure Identifier information. Any records longer than FC_STACK_MAX bytes will be truncated to fit within the FC_STACK_MAX limit.

Flags

-a
: Specifies an FFDC Failure Identifier for a failure condition reported by software used by this application which causes or influenced the condition being recorded at this time. This identifier should have been returned to this application as part of the software's result indication. The caller provides this identifier here so that the FFDC Error Stack can associate the failure report it is making at this time with the previously recorded failure report. This permits problem investigators to trace the cause

of a failure from its various symptoms in this application and others to the root cause in the other software. If no other software failure is responsible for this condition, or if the other software did not return an FFDC Failure Identifier as part of its result information, the **-a** option should not be provided.

-c

Indicates the name of the XPG/4-compliant message catalog that contains a description of the failure being recorded. This name is relative to the **/usr/lib/nls/msg/$LANG** directory. If the message catalog cannot be found, the *default_message* will be displayed to describe the failure. Note that the *default_message* will not be translated between locales.

-d

A character string that provides detailed information on the condition, similar to the Detail Data concept used by the AIX Error Log. If details of the information are too lengthy, these details can be written to a file, and the name of that file provided as an argument to the **-f** option. The **-d** and **-f** options cannot be specified at the same time. If neither the **-d** or the **-f** options are provided or appear valid, the character string **no detail data** is recorded.

-f

Specifies the name of a file containing details about the condition being reported, similar to the Detail Data concept used by the AIX Error Log. This option is used when the details are too lengthy to record within the FFDC Error Stack itself, or when a utility exists that can analyze the detail information. The contents of this file is copied to the **/var/adm/ffdc/dumps** directory, and the file's new location is recorded as the Detail Data in the FFDC Error Stack. If a file containing details of the condition does not exist, do not specify this option. The **-d** and **-f** options cannot be specified at the same time.

-h

Displays a help message to standard output and exits. No other processing is performed, regardless of the options specified.

-l

Specifies an abbreviation of the name of the licensed program in which this software was shipped. This value should be recognizable to customer and application-support services as an acceptable name for the licensed program (AIX, for example). If this option is not provided or does not appear to be valid, the character string **PPS_PRODUCT** is used.

-m

Specifies the message set containing the message describing the failure in the message catalog file. If this message set cannot be located, the *default_message* will be displayed to describe the failure. Note that **default_message** will not be translated to the user's locale.

-n

Specifies the message number that describes the failure being recorded. If this message cannot be located, the *default_message* will be displayed to describe the failure. Note that *default_message* will not be translated to the user's locale.

-o

Specifies a list of substitution parameters within the message indicated by the **-n** option. **fcpushstk** only supports character strings as substitutional parameters (%s) due to the shell operating environment. If multiple substitutional parameters are provided, each one must be separated by a comma (,). If any of these substitution parameters contain imbedded white space, they must be enclosed in double quotes ("").

-q

Suppresses the generation of warning messages from the command. Warning are generated when the command must substitute default information for missing information, or when the command is unable to copy the *detail_data_file* to the **/var/adm/ffdc/dumps** directory.

-r

Specifies the software component name. This is a symbolic name for the software making the report, and should be a name recognizable to both customer and application-support services.

-p

Specifies the line of code location within the source code module where the condition is being reported. The value provided must be a valid integer value. To allow for proper identification and location of the condition, this value should be as close to the line of code that detected the condition

as possible. Korn Shell scripts can use the value of **$LINENO**. Script languages that do not provide a special line count variable can provide a symbolic value here that a developer can use to locate the spot in the source code where **fcpushstk** is being used. If this option is not valid or not provided, the value of **0** is used.

-s
Specifies the name of the source file containing the line of code that encountered the condition being reported. For Korn and Borne Shell scripts, the argument to this option should be set to **$0**; C Shell scripts would set this argument to ${0}. If this option is not provided or not valid, the character string **unknown_file** is used.

-v
Indicates the SCCS version number of the source code module that detected the condition being recorded. For source code under SCCS control, this should be set to **"1.1"** (the double-quotes are necessary). If this option is not provided or is not valid, the character string **unknown** is used.

Parameters

default_message
Indicates a default message to be used as a description of the failure, when the information cannot be retrieved from the message catalog information supplied through the **-c**, **-m**, and **-n** options. If this string contains positional parameters, all positional parameters must be specified to be character strings (%s). The message should be enclosed in double quotes ("") if it contains any embedded white space. **fcpushstk** limits the overall length of this string to 72 characters.

Exit Status

fcpushstk returns the following exit status codes upon successful completion:

0
FFDC Error Stack Environment exists, and failure information successfully recorded in the FFDC Error Stack. An FFDC Failure Identifier for the record is displayed to standard output. The caller should capture standard output to obtain this value.

2
Help information displayed and processing ended.

fcpushstk returns the following exit status codes when a failure occurs:

11
No information recorded to the FFDC Error Stack, and no FFDC Failure Identifier is provided by this command. The client requested to use an option not supported in this release of the FFDC software

12
No information recorded to the FFDC Error Stack, and no FFDC Failure Identifier is provided by this command. Unknown function parameter provided to the interface.

15
FFDC Error Stack Environment does not exist. No information recorded to the FFDC Error Stack. No FFDC Failure Identifier is generated by this command. This is the normal return code to the FFDC client when an FFDC Error Stack Environment did not exist to be inherited via **fcinit**.

17
No information recorded to the FFDC Error Stack, and no FFDC Failure Identifier is provided by this command. The FFDC Error Stack Environment appears to be corrupted and should be considered unusable.

19
No information recorded to the FFDC Error Stack - the FFDC Error Stack directory does not exist or cannot be used. No FFDC Failure Identifier is provided by this command.

20
No information recorded to the FFDC Error Stack, and no FFDC Failure Identifier is provided by this command. Unable to access the FFDC Error Stack file. The file may have been removed, or permissions on the file or its directory have been changed to prohibit access to the FFDC Error Stack.

22
No information recorded to the FFDC Error Stack - the FFDC Error Stack file could not be locked for exclusive use by this interface. Repeated attempts had been made to lock this file, and all attempts failed. Another process may have locked the file and failed to release it, or the other process may be hung and is preventing other processes from using the FFDC Error Stack. No FFDC Failure Identifier is provided by this command.

24
No information recorded to the FFDC Error Stack, and no FFDC Failure Identifier is provided by this command. The FFDC Error Stack file appears to be corrupted. The client should consider the FFDC Error Stack Environment unusable.

25
No information recorded to the FFDC Error Stack, and no FFDC Failure Identifier is provided by this command. The FFDC Error Stack file name is set to a directory name. The FFDC Error Stack Environment should be considered corrupted and unusable.

32
A dump file could not be copied to the **/var/adm/ffdc/dumps** directory. There is insufficient space in the file system containing the **/var/adm/ffdc** directory. The **fcclear** command should be used to remove unneeded FFDC Error Stacks and dump files, or the system administrator needs to add more space to the file system. No FFDC Failure Identifier is provided by this command.

40
No information recorded to the FFDC Error Stack - information could not be recorded in the FFDC Error Stack. There is insufficient space in the file system containing the **/var/adm/ffdc** directory. The **fcclear** command should be used to remove unneeded FFDC Error Stacks and dump files, or the system administrator needs to add more space to the file system. No FFDC Failure Identifier is provided by this command.

41
No information recorded to the FFDC Error Stack, and no FFDC Failure Identifier is provided by this command. A failure occurred when reading control information from the FFDC Error Stack or writing incident information to the FFDC Error Stack. The client should conclude that the entry was not recorded for this incident.

99
No information recorded to the FFDC Error Stack, and no FFDC Failure Identifier is provided by this command. An unexpected internal failure occurred in the **fc_push_stack** routine.This problem may require the attention of application-support services.

When **fcpushstk** is provided with incomplete information, it substitutes default information for the missing information and attempts to make a record in the FFDC Error Stack. Warnings are generated in these cases, and warning messages are displayed to the standard error device unless the **-q** option has been specified. In cases where more than one warning condition was detected, the command generates an exit status code corresponding to the most severe warning condition it detected. The following exit status codes are returned by **fcpushstk** when warning conditions are detected:

26
Both a detailed data string and a detail data file were provided to this routine. The routine chose the detail data string and ignored the detail data file.

28
The name of the resource detecting the incident was not provided. The default resource name was substituted for the missing resource name.

29
At least one component of the detecting application information—source code file name, source code file version, LPP name, line of code position—was not provided. Default information was substituted for the missing information.

30
No default message was provided to describe the nature of the incident. If the XPG/4 message catalog containing the description message cannot be found, no description for this condition will be displayed by the **fcstkrpt** command.

31
No message was provided to describe the nature of the incident, or a component of the XPG/4 information—catalog file name, message set number, message number—was not provided. No description for this condition can be displayed by the **fcstkrpt** command.

32
The file named in the *detail_data_file* parameter could not be copied to the **/var/adm/ffdc/dumps** directory. The FFDC Error Stack entry cites the original version of this file. Do not discard the original copy of this file.

35
No detailed information was provided for this incident. Later problem analysis may be difficult without these details to indicate specifics on the incident.

37
An FFDC Failure Identifier could not be constructed for the report created by this routine. No FFDC Failure Identifier is provided by this command, but information on the incident was recorded to the FFDC Error Stack.

44
The information provided to this command would have caused an FFDC Error Stack record to exceed the FC_STACK_MAX limit. The record was truncated to allow it to be recorded within the system limits. Important information about the failure may have been lost during the truncation process. Modify the script to provide less information, or to record the information to a detail data file and submit the detail data file name to this command instead.

Examples

To record information about a failure to the FFDC Error Stack when the FFDC Environment is established or inherited by the process:

```
#!/bin/ksh
    :
    :
cp /tmp/workfile $FILENAME
RC=$?
if ((RC != 0))
then
    FFDCID=$(fcpushstk -c mymsg.cat -m2 -n10 -o$FILENAME -r myprog
         -d"cp exit status $RC - file being copied /tmp/workfile" -s$0
         -p$LINENO -v"1.1" -lPSSP "Cannot update configuration file %1$s")
    if (($? == 0))
    then
         fcdispfid $FFDCID
         return 1
    fi
fi
    :
    :
```

To make the same recording from a script language that does not have a line of code variable available:

```
#!/bin/bsh
    :
    :
CODESCTN=14             # Used to identify where in the script code we are
cp /tmp/workfile $FILENAME
RC=$?
if test $RC -ne 0
then
    FFDCID=`fcpushstk -c mymsg.cat -m2 -n10 -o$FILENAME -r myprog
         -d"cp exit status $RC - file being copied /tmp/workfile" -s$0
         -p$CODESCTN -v"1.1" -lPSSP "Cannot update configuration file %1$s"`
    if test $? -eq 0
    then
         fcdispfid $FFDCID
         return 1
    fi
fi
CODESECTION=15              # New code section begins - a different task starts
```

```
    :
    :
```

To record information about a failure condition that is related to another failure condition previously recorded to the FFDC Error Stack by an application exploiting FFDC:

```
#!/bin/ksh
    :
    :
ASSOC_FID=$(/usr/lpp/ssp/bin/somecmd -a -b)
RC=$?if ((RC != 0))
then
    FFDCID=$(fcpushstk -a$ASSOC_FID -c mymsg.cat -m2 -n10 -o$FILENAME -r myprog
         -d"cp exit status $RC - file being copied /tmp/workfile" -s$0
         -p$LINENO -v"1.1" -lPSSP "Cannot update configuration file %1$s")
    if (($? == 0))
    then
         fcdispfid $FFDCID
         return 1
    fi
fi
    :
    :
```

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

fcreport Command

Purpose

Locates and displays the report of a failure and any failures associated with the failure.

Syntax

`/opt/rsct/bin/fcreport { [ -a ] FFDC_Failure_ID } | -h`

Description

fcreport decodes an FFDC Failure Identifier, and obtains reports on the failure identified by it. The command also detects if any failure was associated with the FFDC Failure Identifier, and if so, obtains the report on that failure. The command continues to examine the report of each failure it locates for associated failures and to obtain reports on the associated failures until one of the following conditions is met:

- No further associated failures are detected.
- The report for an associated failure cannot be found. This may occur when the associated failure report resides on a remote node that cannot be reached at the moment, or the record of the failure has been removed from the node where it resided.

Using this command, the user can obtain a report for the entire list of failures that caused a specific failure. **fcreport** is not capable of locating reports for any failures that may have been caused by the initial failure provided to the command; it can only obtain reports of failures that caused this failure.

Flags

-a
: Displays all information contained in a report for a failure. The default is to display the network address of the node where the failure report was generated, the time stamp on the failure report, and the description of the incident recorded in the failure report.

-h
Displays a help message to standard output and exits. No other processing is performed, regardless of the options specified.

Parameters

FFDC_Failure_ID
Specifies the FFDC Failure Identifier of the failure to begin the report. **fcreport** will attempt to obtain the failure information for this failure, as well as any failures that this report lists as an associated failure. Only one FFDC Failure Identifier may be provided to this command.

Security

`fcreport` uses `rsh` to obtain failure reports that may reside on remote nodes. The user must have sufficient privilege to execute `rsh` commands to these remote nodes. If the user does not have this permission, `fcreport` can only trace the list of related failures so long as they exist on the local node.

Exit Status

fcreport generates one of the following exit status codes upon completion:

0
Failure report located and displayed for the FFDC Failure Identifier provided. Zero or more related failure reports may have been located and displayed as well.

2
Help information displayed and processing ended.

10
Required options or arguments are not provided.

11
The FFDC Failure Identifier provided to this command was generated by a later release of the FFDC software. The command is not capable of correctly interpreting this identifier.

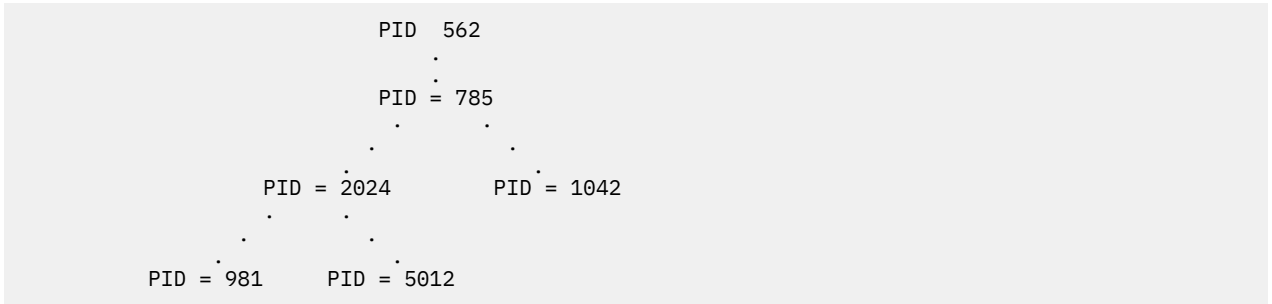
12
Unknown option specified to this command.

20
The FFDC Failure Identifier refers to an entry made in an FFDC Error Stack on this system, but the FFDC Error Stack file cannot be accessed. The file may have been removed, or permissions may have been altered on the file to prevent access to it.

27
The FFDC Failure Identifier provided to this command is not a valid identifier.

Examples

Consider the case where several processes were created in the following parent-child order:



In this example, process 785 generated the FFDC Failure Identifier `.3Iv04ZVVfvp.wtY0xRXQ7.....` and passed it back to Process 562. To obtain

a detailed report for FFDC Failure Identifier `.3Iv04ZVVfvp.wtY0xRXQ7....................` and any previous failures that led to this specific failure:

```
$ fcreport -a .3Iv04ZVVfvp.wtY0xRXQ7...................
```

This report will contain the details of the specified FFDC Failure Identifier, as well as any failures in processes 2024, 1042, 981, and 5012 that may have caused it. The report will not contain any failures in process 562 that may have been caused as a result of process 785's failure.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

fcstat Command

Purpose

Displays statistics gathered by the specified Fibre Channel device driver.

Syntax

`fcstat` [**-z** [**-d** | **-c**] | **-d** | **-e** [**-d** | **-c**] | **-c**] *Device_Name*

Description

The `fcstat` command displays the statistics gathered by the specified Fibre Channel device driver. You can optionally specify that the device-specific statistics are displayed in addition to the device-generic statistics. If you specify no flags, the `fcstat` command displays only the device-generic statistics. The `fcstat` command collects the statistics by using the following procedure:

1. Opens the message catalog of `fcstat` and checks the parameter list.
2. Accesses the Object Data Manager (ODM) database for information relating to the selected adapter.
3. Accesses the ODM database for information relating to ports of the selected adapter.
4. Opens and accesses adapter statistics.
5. Resets some of the statistics if you specify the **-z** flag.
6. Reports statistics and exits.

If an invalid *Device_Name* is specified, the `fcstat` command returns an error message stating that it cannot find the device in the ODM database.

The **fcstat** command also reports the statistics if the specified *Device_Name* is not connected to a network (that is, the link is down), by opening the device in diagnostic mode by using the **-d** flag. When the link is down and the device is opened in non-diagnostic mode, the **fcstat** command delays in generating the output. You can use the **-c** flag to remove this delay. If the device is already opened and the **fcstat** command is started with the **–d** flag, the open operation on the device fails with an EACCESS error.

When the `fcstat` command is not able to extract statistics from the specified *Device_Name*, it still reports the information that it extracted from the ODM database.

Flags

Item	Description
-c	Removes the delay in generating the output when the device is opened in non-diagnostic mode and the link is down.

Item	Description
-d	Displays the statistics by opening the adapter in diagnostic mode.
-e	Displays all the statistics, which includes the device-specific statistics (driver statistics, link statistics, and FC4 types).
-z	Resets some of the statistics back to their initial values. Only privileged users can issue this flag.

Parameters

Item	Description
Device_Name	The name of the Fibre Channel device. For example, `fcs0`.

Statistics fields

Note: Some adapters might not support a specific statistic. The value of non-supported statistic fields is always 0. All the parameters marked with the asterisk (*) are reset to their initial values when you use the **fcstat** command with the **–z** flag.

The statistic fields displayed in the output of the `fcstat` command and their descriptions follow:

	Item	Description
	`Device Type`	Displays the description of the adapter.
	`Serial Number`	Displays the serial number from the adapter.
	`Option ROM Version`	Displays the version of the Options ROM on the adapter.
	`ZA`	Displays the ZA field from the VPD of the adapter.
	`Node WWN`	Displays the worldwide name of the adapter.
	`Port FC ID`	Displays the SCSI ID of the adapter.
	`Port Type`	Displays the connection type of the adapter.
	`Port Speed`	Displays the speed of the adapter.
	`Port WWN`	Displays the worldwide name of the port.
	`Seconds Since Last Reset`	Displays the seconds since last reset of the statistics on the adapter.
*	`Frames`	Displays the number of frames transmitted and received.
*	`Words`	Displays the number of words transmitted and received.
*	`LIP Count`	Displays the LIP count.
*	`NOS Count`	Displays the NOS count.
	`Error Frames`	Displays the number of frames that were in error.
*	`Dumped Frames`	Displays the frames that were dumped.
	`Link Failure Count`	Displays the Link Failure Count.
	`Loss of Sync Count`	Displays the number of times Sync was lost.

	Item	Description
	Loss of Signal	Displays the number of times signal was lost.
	Primitive Seq Protocol Err Count	Displays the number of times a primitive sequence was in error.
	Invalid Tx Word Count	Displays the number of invalid transfers that occurred.
	Invalid CRC Count	Displays the number of CRC errors that occurred.
	FC SCSI Adapter Driver Information: No DMA Resource Count	Displays the number of times DMA resources were not available.
	FC SCSI Adapter Driver Information: No Adapter Elements Count	Displays the number of times there were no adapter elements available.
	FC SCSI Adapter Driver Information: No Command Resource Count	Displays the number of times there were no command resources available.
*	FC SCSI Traffic Statistics: Input Requests	Displays the number of input requests.
*	FC SCSI Traffic Statistics: Output Requests	Displays the number of output requests.
*	FC SCSI Traffic Statistics: Control Requests	Displays the number of control requests.
*	FC SCSI Traffic Statistics: Input Bytes	Displays the number of input bytes.
*	FC SCSI Traffic Statistics: Output Bytes	Displays the number of output bytes.
	Adapter Effective Max Transfer Value	Displays the effective max transfer value.
	FC4 Types: Supported ULP	Displays the supported ULP.
	FC4 Types: Active ULP	Displays the active ULP.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Examples

1. To display the statistics for Fibre Channel device driver fcs0, enter:

```
fcstat fcs0
```

Output similar to the following is displayed.

Note: The output format of various AIX commands is not always static. Do not write programs with the expectation that the output for the fcstat command remains as follows.

```
FIBRE CHANNEL STATISTICS REPORT: fcs0

    Device Type: FC Adapter (df1000f9)
```

```
Serial Number: 1E313BB001
Option ROM Version: 02C82115
ZA: B1F2.10A5
Node WWN: 20000000C9487B04
Port WWN: 10000000C9416DA4

FC4 Types
  Supported: 0x0000010000000000000000000000000000000000000000000000000000000000
  Active:    0x0000010000000000000000000000000000000000000000000000000000000000
Class of Service: 4
Port FC ID: 011400
Port Speed (supported): 2 GBIT
Port Speed (running):   1 GBIT
Port Type: Fabric

Seconds Since Last Reset: 345422

Transmit Statistics    Receive Statistics
-------------------    ------------------
Frames: 1              Frames: 1
Words: 1               Words: 1

LIP Count: 1
NOS Count: 1
Error Frames:  1
Dumped Frames: 1
Link Failure Count: 1
Loss of Sync Count: 1
Loss of Signal: 1
Primitive Seq Protocol Err Count: 1
Invalid Tx Word Count: 1
Invalid CRC Count: 1

FC SCSI Adapter Driver Information
  No DMA Resource Count: 0
  No Adapter Elements Count: 0
  No Command Resource Count: 0

FC SCSI Traffic Statistics
  Input Requests:   16289
  Output Requests:  48930
  Control Requests: 11791
  Input Bytes:  128349517
  Output Bytes: 209883136
```

Location

`/usr/sbin/fcstat`

fcstkrpt Command

Purpose

Displays the contents of an FFDC Error Stack file.

Syntax

`/opt/rsct/bin/fcstkrpt` { [`-a`] [`-p` | `-r`] { `-f` *FFDC_Failure_Identifier* [`-i`] | `-s` *FFDC_Error_Stack_File_Name* } } | [`-h`]

Description

fcstkrpt reads an existing FFDC Error Stack file and displays its contents to the standard output device. The FFDC Error Stack file is indicated either through the name of the file itself, or by using an FFDC Failure Identifier that references a specific record within that file.

Information from the FFDC Error Stack can be displayed in one of two formats: by *related failure conditions* (the default) or by *software layer*.

Flags

-a
Indicates that all information be displayed for entries in the FFDC Error Stack. The default action is to display the time stamp for the record and the description of the incident.

-f
Specifies the FFDC Failure Identifier to use in locating the FFDC Error Stack. **fcstkrpt** decodes the FFDC Failure Identifier, locates the FFDC Error Stack associated with that FFDC Failure Identifier, and processes the FFDC Error Stack. Only one FFDC Failure Identifier can be specified by this flag.

-h
Displays a help message to standard output and exits. No other processing is performed regardless of the options specified.

-i
Displays only the information associated with the specific failure report identified by the **-f** flag. By default, all records in the FFDC Error Stack are displayed.

-p
Displays information from the FFDC Error Stack by process orientation. The output is ordered so that it reflects the order in which the processes were created (parent-child process relationship). Child process information is shown first, followed by parent process information. This view is used to understand which incidents occurred first, and which incidents occurred later because of them.

-r
Displays information from the FFDC Error Stack by incident relationships. Incidents are presented along with those incidents that are related to them. This view is used to understand which incidents occurred because of the occurrence of other incidents. This is the default.

-s
Specifies the name of the FFDC Error Stack to be examined. This name may be either the absolute or relative path name of the FFDC Error Stack. Only one FFDC Error Stack file name can be specified by this flag. If a relative file name is used, the file is assumed to be located in the **/var/adm/ffdc/stacks** directory of the node where the file resides.

Parameters

FFDC_Failure_ID
Specifies the FFDC Failure Identifier of the failure to begin the report. **fcreport** will attempt to obtain the failure information for this failure, as well as any failures that this report lists as an associated failure. Only one FFDC Failure Identifier may be provided to this command.

Security

`fcreport` uses `rsh` to obtain failure reports that may reside on remote nodes. The user must have sufficient privilege to execute `rsh` commands to these remote nodes. If the user does not have this permission, `fcreport` can only trace the list of related failures so long as they exist on the local node.

Exit Status

fcstkrpt issues the following integer exit status codes upon completion:

0
FFDC Error Stack file successfully located, and contents displayed to the standard output device.

2
Help information displayed and processing ended.

12
An invalid option was specified.

14
No information written to the standard output device. The **-f** option was used and the *FFDC Error Identifier* argument was not valid.

20
No information written to the standard output device. The **-s** option was used and the *FFDC Error Stack File* argument was not found.

27
No information written to the standard output device. The caller provided a valid *FFDC Failure Identifier*, but the file referenced by the FFDC Failure Identifier was not recorded on this node. Use the **fcdecode** command to locate the node where this FFDC Error Stack resides.

81
No information written to the standard output device. A failure occurred while writing information to standard output. The application should conclude that standard output cannot accept output.

85
No information written to the standard output device. The caller provided a valid FFDC Failure Identifier, but the file referenced by the FFDC Failure Identifier does not exist.

Examples

To obtain a brief report of the information stored in the FFDC Error Stack file **/var/adm/ffdc/stacks/myprog.562.19981001143052**:

```
$ fcstkrpt -r -s myprog.562.19981001143052
```

To obtain a detailed report of the information contained in the FFDC Error Stack where the FFDC Failure Identifier **.3Iv04ZVVfvp.wtY0xRXQ7.....** was recorded, and present this information in parent-child ordering:

```
$ fcstkrpt -p -f .3Iv04ZVVfvp.wtY0xRXQ7...................
```

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

fcteststk Command

Purpose

Test for the presence of a First Failure Data Capture Error Stack environment.

Syntax

`/opt/rsct/bin/fcteststk [-q] | [-h]`

Description

fcteststk can be called by any application program that wishes to use the FFDC Error Stack to test if these facilities have been activated . By performing this test, applications can avoid the performance burden of collecting failure information in cases where an *FFDC Environment* has not been established. This interface is provided primarily for use by library routines, which would not have any knowledge of whether their client application established or inherited an *FFDC Environment*.

An *FFDC Error Stack Environment* is established by a process when that process wants to have failure information from itself, any threads it may create, and any descendant processes it may create to be recorded in an FFDC Error Stack. An *FFDC Error Stack Environment* is inherited by a process when that process wants to record failure information to an FFDC Error Stack file only when one of its ancestors has

requested for processes to do so; in all other cases, the process will not record failure information to the FFDC Error Stack. Processes use **fcinit** to either establish or inherit the FFDC Error Stack Environment.

The FFDC Error Stack Environment reserves an FFDC Error Stack file, so that failure information is recorded to a file in the **/var/adm/ffdc/stacks** directory. These files use the naming format ***script_name.PID.date_and_time***, where *script_name* is the name of the script itself, *PID* is the process identifier of the script, and *date_and_time* is the date and time when the script was executed. Whenever this script or children processes of this script record failure information to the FFDC Error Stack, it will be recorded in this file.

Applications use the **fcpushstk** interface to record failure information to the FFDC Error Stack. However, the application may need to collect this information from various locations before recording the information, and obtaining this information can impact the application's overall performance. The application should not need to collect this information if the *FFDC Error Stack Environment* was not established or inherited. To avoid this performance impact, the application can issue **fcteststk** to determine if an *FFDC Error Stack Environment* is available, and if so, begin collecting the failure information. If the *FFDC Error Stack Environment* does not exist, the application can avoid collecting this information.

Processes that use the **fclogerr** FFDC interface can use **fclogerr** when an *FFDC Environment* exists, whether or not an FFDC Error Stack is in use by the *FFDC Environment*. Whenever **fclogerr** is used, failure information is recorded to the AIX Error Log and the BSD System Log, regardless of whether an FFDC Error Stack was reserved. Any application that records information using the **fclogerr** interface must *always* collect the failure information and record it, regardless of whether an FFDC Error Stack is in use.

Flags

Item	Description
`-h`	Displays a usage message for this command. No further processing is performed.
`-q`	Suppresses output from this command that explains whether or not an FFDC Environment was established. The command user will be required to test the exit status from the command to determine whether an FFDC Environment is established for this process.

Parameters

FFDC_Failure_ID

Specifies the FFDC Failure Identifier of the failure to begin the report. **fcreport** will attempt to obtain the failure information for this failure, as well as any failures that this report lists as an associated failure. Only one FFDC Failure Identifier may be provided to this command.

Security

`fcreport` uses `rsh` to obtain failure reports that may reside on remote nodes. The user must have sufficient privilege to execute `rsh` commands to these remote nodes. If the user does not have this permission, `fcreport` can only trace the list of related failures so long as they exist on the local node.

Exit Status

0

An FFDC Error Stack Environment exists.

2

Help information displayed and processing ended.

12

No processing performed. An invalid option was specified.

15

FFDC Error Stack Environment has not been established or inherited by the client at this point in time.

17
FFDC Error Stack Environment appears to be corrupted and should be considered unusable.

Examples

To test whether an FFDC Error Stack Environment exists for an application:

```
fcteststk -q
if (($? == 0))
then
    # Collect failure information
          :
          :
    # Use fcpushstk to record failure info
          :
          :
fi
```

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

fddistat Command

Purpose

Shows FDDI device driver and device statistics.

Syntax

fddistat [**-r** **-t**] *Device_Name*

Description

The **fddistat** command displays the statistics gathered by the specified FDDI device driver. If no flags are specified, only the device driver statistics are displayed. This command is also invoked when the **netstat** command is run with the **-v** flag. The **netstat** command does not issue any **fddistat** command flags.

If an invalid *Device_Name* is specified, the **fddistat** command will produce an error message stating that it could not connect to the device.

Flags

Item	Description
-r	Resets all the statistics back to their initial values. This flag can only be issued by privileged users.
-t	Toggles debug trace in some device drivers.

Parameter

Item	Description
Device_Name	The name of the FDDI device, for example, **fddi0**.

Statistic Fields

Note: Some adapters may not support a specific statistic. The value of non-supported statistic fields is always 0.

The statistic fields displayed in the output of the **fddistat** command and their descriptions are:

Title Fields

Item	Description
Elapsed Time	Displays the real time period has elapsed since last time the statistics was reset. Since part of the statistics may be reset by the device driver during error recovery when a hardware error was detected, there will be another Elapsed Time displayed in the middle of the output when this situation has occurred in order to reflect the time differences between the statistics.

Transmit Statistics Fields

Item	Description
Packets	The number of packets transmitted successfully by the device.
Bytes	The number of bytes transmitted successfully by the device.
Interrupt	The number of transmit interrupts received by the driver from the adapter.
Transmit Errors	The number of output errors encountered on this device. This is a counter for unsuccessful transmissions due to hardware/network errors.
Packets Dropped	The number of packets accepted by the device driver for transmission which were not (for any reason) given to the device.
Max Packets on S/W Transmit Queue	The maximum number of outgoing packets ever queued to the software transmit queue.
S/W Transmit Queue Overflow	The number of outgoing packets overflowed the software transmit queue.
Current S/W+H/W Transmit Queue Length	The number of pending outgoing packets on either the software transmit queue or the hardware transmit queue.
Broadcast Packets	The number of broadcast packets has been transmitted without any error.
Multicast Packets	The number of multicast packets has been transmitted without any error.

Receive Statistics Fields

Item	Description
Packets	The number of packets has been received successfully by the device.
Bytes	The number of bytes received successfully by the device.
Interrupts	The number of receive interrupts received by the driver from the adapter.

Item	Description
Receive Errors	The number of input errors encountered on this device. This is a counter for unsuccessful reception due to hardware/network errors.
Packets Dropped	The number of packets received by the device driver from this device which were not (for any reason) given to a network demuxer.
Bad Packets	The number of bad packets received (i.e.saved) by the device driver.
Broadcast Packets	The number of broadcast packets received without any error.
Multicast Packets	The number of multicast packets received without any error.

General Statistics Fields

Item	Description
No mbuf Errors	The number of times that mbufs were not available to the device driver. This usually occurs during receive operations when the driver must obtain mbuf buffers to process inbound packets. If the mbuf pool for the requested size is empty, the packet will be discarded. The **netstat -m** command can be used to confirm this.
SMT Error Word	The adapter's SMT error status.
SMT Event Word	The adapter's SMT event status.
Connection Policy Violation	The status of the adapter's connection to the ring.
Port Event	The adapter's port status.
Set Count	The current set count value.
Adapter Check Code	The adapter's most recent adapter check status.
Purged Frames	Receive frames dropped by the adapter due to lack of available descriptors.
ECM State Machine	Entity Coordination Management State Machine.
PCM State Machine: Port A	Physical Connection Management for the primary adapter State Machine
PCM State Machine: Port B	Physical Connection Management for the secondary adapter State Machine
CFM State Machine: Port A	Configuration Management for the primary adapter State Machine
CFM State Machine: Port B	Configuration Management for the secondary adapter State Machine
CF State Machine	Overall Configuration State Machine.
MAC CFM State Machine	Configuration Management for the MAC State Machine.
RMT State Machine	Ring Management State Machine.
Driver Flags	The device driver internal status flags that are currently turned on.

Example

To display the device driver statistics for **fddi0**, enter:

```
fddistat fddi0
```

This produces the following output:

```
------------------------------------------------------------
FDDI STATISTICS (fddi0) :
Elapsed Time: 0 days 0 hours 1 minutes 3 seconds
```

```
Transmit Statistics:                          Receive Statistics:
--------------------                          -------------------
Packets: 100                                  Packets: 100
Bytes: 113800                                 Bytes: 104700
Interrupts: 100                               Interrupts: 100
Transmit Errors: 0                            Receive Errors: 0
Packets Dropped: 0                            Packets Dropped: 0
Max Packets on S/W Transmit Queue: 0          Bad Packets: 0
S/W Transmit Queue Overflow: 0
Current S/W+H/W Transmit Queue Length: 0
```

```
Broadcast Packets: 0                          Broadcast Packets: 0
Multicast Packets: 0                          Multicast Packets: 0
```

```
General Statistics:
-------------------
No mbuf Errors: 0
SMT Error Word: 00040080                      SMT Event Word: 000004a0
Connection Policy Violation: 0000             Port Event: 0000
Set Count Hi: 0000                            Set Count Lo: 0003
Adapter Check Code: 0000                      Purged Frames: 0
```

```
ECM State Machine:        IN
PCM State Machine Port A: CONNECT
PCM State Machine Port B: ACTIVE
CFM State Machine Port A: ISOLATED
CFM State Machine Port B: CONCATENATED
CF State Machine:         C_WRAP_B
MAC CFM State Machine:    PRIMARY
RMT State Machine:        RING_OP
```

```
Driver Flags: Up Broadcast Running
        Simplex DualAttachStation
```

fdformat Command

Purpose

The **fdformat** command formats diskettes.

Syntax

fdformat [*Device*] [**-h**]

Description

Attention: Formatting a diskette or read/write optical disk destroys any existing data on it.

The **fdformat** command formats diskettes in the diskette drive specified for low density unless the **-h** flag is specified.

All new, blank diskettes must be formatted before they can be used.

Before formatting a diskette or read/write optical disk, the **fdformat** command prompts for verification. This allows you to end the operation cleanly.

Flags

Item	Description
-h	Forces high-density formatting. This flag is used only with the **fdformat** command.

Parameters

Item	Description
Device	Specifies the device containing the diskette to be formatted. The default is the **/dev/rfd0** device for drive 0.

Examples

To force high-density formatting of a diskette when using the **fdformat** command, enter:

```
fdformat -h
```

Files

Item	Description
/usr/sbin/fdformat	Contains the **fdformat** command.
/dev/rfd*	Specifies the device parameters.
/dev/fd*	Specifies the device parameters.
/dev/romd*	Specifies the device parameters.
/dev/omd*	Specifies the device parameters.

fdpr Command

Item	Description
-analyse_asm_csects	Analyze csects written in assembly (when used, must be specified at both the -1 and -3 phases).
-extra_safe_analysis	Do not attempt to analyze unconventional csects containing hand-written assembly code (when used, must be specified at both the -1 and -3 phases).
-ignore_info	Ignore **.info** sections produced with the **-qfdpr** option during compile time (when used, must be specified at both -1 and -3 phases).
-align *bytes*	Align frequently executed code according to given number of bytes, for improving code prefetch buffer ratio. If this option is omitted, the **fdpr** command aligns the code with variable default number of bytes.
-lr_opt	Eliminate stores and restores of the link register in frequently executed procedures.
-bt_csect_anchor_removal	Eliminate load instructions related to the usage of branch tables in the code.
-dead_code_removal	Remove unreachable code.

Item	Description
-selective_inline	Perform selective inlining for functions that are frequently called from a single dominant call site.
-sid_fac *percent*	Set a dominant factor percentage for selective inline optimization. The allowed range is between 50 - 100 (applicable only with the **-selective_inline** flag).
-inline_small_funcs *size*	Inline all functions that are smaller or equal to the given size in bytes.
-inline_hot_funcs *percent*	Inline all functions with an execution frequency equals or greater than the given percentage. The input percent range is between 0 - 100.
-inline	Perform **-inline_small_funcs 12** with **-selective_inline**.
-hco_resched	Relocate instructions from frequently executed code to rarely executed code area, when possible.
-dcbt_opt	Insert **dcbt** instructions to improve data-cache performance.
-killed_regs	Eliminate stores and restores of registers that are *killed* (overwritten) after frequently executed function calls.
-tb	Force the restructuring of traceback tables in reordered code. If **-tb** option is omitted, traceback tables are automatically restored for C++ applications using Try & Catch mechanism.
-pc	Preserve csects' boundaries in reordered code.
-pp	Preserve functions' boundaries in reordered code.
-RD	Perform static data reordering.
-dpnf *factor*	Data Placement Normalization Factor between 0 - 1; where 0 causes static variables to be reordered regardless of their size, whereas 1 will locate only small sized variables first (applicable only with the **-RD** flag).
-dpht *threshold*	Data Placement Hotness Threshold between 0 - 1; where 0 reorders the static variables in large groups based on the control flow, and whereas 1 will reorder the variables in very small groups based on their access frequency (applicable only with the **-RD** flag).
-build_dcg	Build DCG (Data Connectivity Graph) for enhanced data reordering (applicable only with the **-RD** flag).
-tocload	Perform tocload optimization.
-reduce_toc *removal_factor*	Perform TOC entries removal accordingly to removal factor between 0 - 1, where 0 removes only non-accessed TOC entries and 1 removes all non-exported TOC entries.
-strip	Strip the output file (if any is produced).
-ptrgl_opt	Perform optimization of indirect call instructions by way of registers by replacing them with direct jumps.
-no_ptrgl_r11	Do not perform removal of R11 load instruction in _ptrgl csect (the **-ptrgl_r11** optimization is applied by default).
-O	Perform code reordering with branch prediction bit setting, branch folding and NOOP instructions removal. The **-O** flag is applied by default.
-O2	Switch on all less aggressive optimization flags.

Item	Description
-O3	Switch on all aggressive optimization flags.
-O4	Switch on all aggressive optimization flags.

Purpose

A performance tuning utility for improving execution time and real memory utilization of user-level post-link application programs.

Syntax

Most Common Usage:

fdpr -p *ProgramFile* **-x** *WorkloadCommand*

Detailed Usage:

fdpr -p *ProgramFile* [**-M** *SegNum*] [**-fd** *Fdesc*] [**-o** *OutputFile*] [**-armember** *ArchiveMemberList*] [OptimizationFlags] [**-map**] [**-disasm**] [**-disasm_data**] [**-disasm_bss**] [**-profcount**] [**-quiet**] [**-v**] [**-1 | -2** | **-3 | -12** | **-23 | -123**] [**-x** *WorkloadCommand*]

Optimization Flags

[**-tb**] [**-pc**] [**-pp**] [**-O**][**-O2**] [**-O3**] [**-O4**] [**-selective_inline**] [**-sid_fac** *percent*] [**-inline_small_funcs** *size*] [**-inline_hot_funcs** *percent*] [**-hco_resched**] [**-killed_regs**] [**-lr_opt**] [**-align** *bytes*] [**-RD**] [**-dpnf** *factor*] [**-dpht** *threshold*] [**-build_dcg**] [**-tocload**] [**-ptrgl_opt**] [**-no_ptrgl_r11**] [**-dcbt_opt**] [**-ignore_info**] [**-dead_code_removal**] [**-bt_csect_anchor_removal**] [**-strip**] [**-analyse_asm_csects**] [**-extra_safe_analysis**] [**-inline**] [**-reduce_toc** *removal_factor*]

Description

The **fdpr** command (Feedback Directed Program Restructuring) is a performance-tuning utility that may help improve the execution time and the real memory utilization of user-level application programs. The **fdpr** program optimizes the executable image of a program by collecting information on the behavior of the program while the program is used for some typical workload, and then creating a new version of the program that is optimized for that workload. The new program generated by **fdpr** typically runs faster and uses less real memory.

Attention: The **fdpr** command applies advanced optimization techniques to a program which may result in programs that do not behave as expected; programs which are optimized using this tool should be used with due caution and should be rigorously retested with, at a minimum, the same test suite used to test the original program in order to verify expected functionality. The optimized program is not supported.

The **fdpr** command builds an optimized executable program in 3 distinct phases:

- Phase 1 (**-1** flag): Creates an instrumented executable program and an empty template profile file.
- Phase 2 (**-2** flag): Runs the instrumented program and updates the profile data.
- Phase 3 (**-3** flag): Generates the optimized executable program file.

These phases can be run separately or in partial or full combination, but must be run in order (i.e., **-1** then **-2** then **-3** or **-12** then **-3**). The default is to run all three phases.

Note: The instrumented executable, created in phase 1 and run in phase 2, typically runs several times slower than the original program. Due to the increased execution time required by the instrumented program, the executable should be invoked in such a way as to minimize execution duration, while still fully exercising the desired code areas. The **fdpr** command user should also attempt to eliminate, where feasible, any time dependent aspects of the program.

Flags

Item	Description
-1,**-2**, **-3**	Specifies the phase to run. The default is all 3 phases (**-123**). The **-s** flag must be used when running separate phases so that the succeeding phases can access the required intermediate files. The phases must be run in order (for example, **-1**, then **-2**, then **-3**, or **-1**, then **-23**). The **-2** flag must be used along with the invocation flag **-x**.
-M *SegNum*	Specifies where to map shared memory for profiling. The default is **0x30000000**. Specify an alternate shared memory address if the program to be optimized or any of the workload command strings invoked with the **-x** flag use conflicting shared-memory addresses. Typical alternative values are **0x40000000**, **0x50000000**, ... up to **0xC0000000**).
-fd *Fdesc*	Specifies which file descriptor number is to be used for the **profile** file that is mapped to the above shared memory area. The default of *Fdesc* is set to 1999.
-o *OutFile*	Specifies the name of the output file from the optimizer. The default is *program***.fdpr**
-p *ProgramFile*	Contains the name of the executable program file or shared object file or shared library containing shared objects/executables, to optimize. This program must be an unstripped executable.
-armember *ArchiveMemberList*	List of archive members to be optimized, within a shared archive file specified by the **-p** flag. If **-armember** is not specified, all members of the archive file are optimized.
-map	Print a map of basic blocks and static variables with their respective old -> new addresses into a suffixed **.mapper** file.
-disasm	Prints the disassembled text section of the output optimized and instrumented program into a suffixed **.dis_text** file.
-disasm_data	Prints the disassembled data section of the output optimized and instrumented program into a suffixed **.dis_data** file.
-disasm_bss	Prints the disassembled bss section of the output optimized and instrumented program into a suffixed **.dis_bss** file.
-profcount	Prints the profiling counters into a suffixed **.ncounts** file.
-quiet	Quiet output mode.
-v	Verbose output.
-x *WorkloadCommand*	Specifies the command used for invoking the instrumented program. All the arguments after the **-x** flag are used for the invocation. Therefore, the **-x** flag must appear last in the command line. The **-x** flag is required when the **-2** flag is used.

Optimization Flags

Optimization

The **fdpr** command performs, by default, the highest possible level of code reordering optimization together with the optimizations of branch prediction bit setting, branch folding, code alignment and removal of redundant NOOP instructions. The **-pc** flag reorders the entire code while preserving csects' boundaries and therefore, may result in less performance improvement than the default code reordering. Similarly, the **-pp** flag reorders the entire code while preserving procedures' boundaries.

Additional optimizations performed on the entire executable program file are available by the optimization flags above.

Executables built with the **-qfdpr** IBM xl compiler flag contain information to assist **fdpr** in producing reordered programs. Modules which are not compiled with the **-qfdpr** option, are reordered based on the compiler signatures in the symbol table.

Additional performance enhancements may be realized by using static linking when building the program to be reordered. Since the **fdpr** program only reorders the instructions within the executable program specified, any dynamically linked shared library routines called by the program are not optimized. Statically linking these library routines to the executable allows for optimizing both the instructions in the program and all library routines used by the program. There are other advantages as well as disadvantages to building a statically linked program.

Output Files

All files created by the **fdpr** command are stored in the current directory with the exception of any files which may be created by running the workload command specified in the **-x** flag. During the optimization process, the original program is saved by renaming the program, and is only restored to the original program name upon successful completion of the final phase.

The profile file created by the **fdpr** command explicitly uses the full name of the current directory since scripts used to run the program may change the working directory before executing the program.

The files created and/or used by the **fdpr** command are:

Item	Description
program	Name of the unstripped executable to be optimized.
program.**save**	Saved version of the original executable program.
program.**nprof**	Name of the profile file.
program.**instr**	Name of the instrumented version of program.
program.**fdpr**	Default name of optimized executable output file.
program.**instr.dis_text**	Default disassembly file in ASCII format produced by **-disasm** flag after instrumentation phase.
program.**fdpr.dis_text**	Default disassembly file in ASCII format produced by **-disasm** flag after optimization phase.
program.**instr.dis_data**	Default disassembly file in ASCII format produced by **-disasm_data** flag after instrumentation phase.
program.**fdpr.dis_data**	Default disassembly file in ASCII format produced by **-disasm_data** flag after optimization phase.
program.**instr.dis_bss**	Default disassembly file in ASCII format produced by **-disasm_bss** flag after instrumentation phase.
program.**fdpr.dis_bss**	Default disassembly file in ASCII format produced by **-disasm_bss** flag after optimization phase.
program.**instr.mapper**	Default mapping file in ASCII format produced by **-map** flag after instrumentation phase.
program.**fdpr.mapper**	Default mapping file in ASCII format produced by **-map** flag after optimization phase.
program.**ncounts**	Default profile counters file in ASCII format produced by **-profcount** flag.

Enhanced Debugging Capabilities

In order to enable a certain degree of debugging capability for optimized programs, **FDPR** updates the Symbol Table to reflect the changes that were made in the **.text** section.

Entry fields in the Symbol Table that specify addresses of symbols that were relocated during the reordering of **FDPR**, are modified to point to their new addresses in the **.text** section.

In addition, in the case where functions or files are split during reordering, **FDPR** creates new entries in the Symbol Table for each new part of the split function/file. These new parts of the same function are given new symbol names in the Symbol Table according to the following naming convention:

```
<original function name>__fdpr_<function's part number>
```

After code reordering all the new entries are suffixed with the `__fdpr_` string.

Example: Originally, function "main" had the following entry in the Symbol Table:

```
   [Index] m   Value       Scn     Aux   Sclass    Type    Name
    [456]  m  0x00000230    2       1     0x02    0x0000   .main
```

If after code reordering, function `main` was split into 3 parts, then it would have 3 entries in the Symbol Table; one for each part as follows:

```
   [Index] m   Value       Scn     Aux   Sclass    Type    Name
    [456]  m  0x00000304    2       1     0x02    0x0000   .main
   [1447]  m  0x00003328    2       1     0x02    0x0000   .main__fdpr_1
   [1453]  m  0x000033b4    2       1     0x02    0x0000   .main__fdpr_2
```

Examples

The following are typical usage examples of the **fdpr** command.

1. This example allows the user to run all three phases. In this example, `test1` is the unstripped executable and `test2` is a shell script that invokes `test1`. The current working directory is **/tmp/fdpr**.

   ```
   test2 script file:

   # code to exercise test1
   test1 -expand 100 -root $PATH file.jpg -quit
   # the end of test2
   ```

   Execute the **fdpr** command (using the default optimization):

   ```
   fdpr -p test1 -x test2
   ```

   This results in the new reordered executable **test1.fdpr**.

2. To run one phase at a time, execute phase one of **fdpr**.

   ```
   fdpr -1 -p test1
   ```

   This command string creates an instrumented version with the name `test1.instr` and the empty template profile file `test1.nprof`.

   To execute phase two:

   ```
   fdpr -2 -p test1 -x test2
   ```

   This command string executes the script file `test2` that runs the instrumented version of `test1` to collect the profile data.

   To execute phase three:

   ```
   fdpr -3 -p test1
   ```

   Again, this results in the new reordered executable **test1.fdpr**.

3. To run the first two phases followed by phase three, execute phase one and two.

   ```
   fdpr -12 -p test1 -x test2
   ```

Execute phase three using optimization level three.

```
fdpr -3 -O3 -p test1
```

4. If an error occurs while running an **fdpr** optimized program, the **dbx** command can be used to determine what procedure the error occurred in as follows:

   ```
   dbx program.fdpr
   ```

   which produces the output similar to the following:

   ```
   Type 'help' for help.
   reading symbolic information ...warning: no source compiled with -g

   [using memory image in core]

   Segmentation fault in proc_d at 0x10000634
   0x10000634 (???) 98640000        stb   r3,0x0(r4)
   (dbx)
   ```

   A stack traceback, which is used to determine how the program arrived at the current location, is produced as follows:

   ```
   (dbx) where
   ```

   which produces the following output:

   ```
   proc_d(0x0) at 0x10000634
   proc_c(0x0) at 0x10000604
   proc_b(0x0) at 0x100005d0
   proc_a(0x0) at 0x1000059c
   main(0x2, 0x2ff7fba4) at 0x1000055c
   (dbx)
   ```

5. The **dbx** subcommand **stepi** may also be used to single step through the instructions of a reordered executable program as follows:

   ```
   (dbx) stepi
   ```

   which produces the following output:

   ```
   stopped in proc_d at 0x1000061c
   0x1000061c (???) 9421ffc0       stwu   r1,-64(r1)
   (dbx)
   ```

   In this example, **dbx** indicates that the program stopped in routine `proc_d` at address `0x1000061c` in the reordered text section.

Implementation Specifics

Software Product/Option: *AIX Performance Aide/ Local Performance Analysis & Control Commands*.

Standards Compliance: None.

Files

Item	Description
/usr/bin/fdpr	Contains the **fdpr** command.
program	Name of the unstripped executable to be optimized.
*program***.save**	Saved version of the original executable program.
*program***.nprof**	Name of the profile file.
*program***.instr**	Name of the instrumented version of program.

Item	Description
program.**fdpr**	Default name of optimized executable output file.
program.**instr.dis_text**	Default disassembly file in ASCII format produced by **-disasm** flag after instrumentation phase.
program.**fdpr.dis_text**	Default disassembly file in ASCII format produced by **-disasm** flag after optimization phase.
program.**instr.dis_data**	Default disassembly file in ASCII format produced by **-disasm_data** flag after instrumentation phase.
program.**fdpr.dis_data**	Default disassembly file in ASCII format produced by **-disasm_data** flag after optimization phase.
program.**instr.dis_bss**	Default disassembly file in ASCII format produced by **-disasm_bss** flag after instrumentation phase.
program.**fdpr.dis_bss**	Default disassembly file in ASCII format produced by **-disasm_bss** flag after optimization phase.
program.**instr.mapper**	Default mapping file in ASCII format produced by **-map** flag after instrumentation phase.
program.**fdpr.mapper**	Default mapping file in ASCII format produced by **-map** flag after optimization phase.
program.**ncounts**	Default profile counters file in ASCII format produced by **-profcount** flag.

fencevsd Command

Purpose

Prevents an application running on a node or group of nodes from accessing a virtual shared disk or group of virtual shared disks.

Syntax

`fencevsd` {`-a` | `-v` *vsd_name_list*} `-n` *node_list*

Description

Under some circumstances, the system may believe a node has stopped functioning and begin recovery procedures, when the node is actually operational, but cut off from communication with other nodes running the same application. In this case, the problem node must not be allowed to serve requests for the virtual shared disks it normally serves until recovery is complete and the other nodes running the application recognize the problem node as operational. The `fencevsd` command prevents the problem node from filling requests for its virtual shared disks.

This command can be run from any node in the RSCT peer domain where the recoverable virtual shared disk subsystem is running.

Flags

-a
Specifies all virtual shared disks.

-v *vsd_name_list*
Specifies one or more virtual shared disk names, separated by commas.

-n ***node_list***
Specifies one or more node numbers, separated by commas.

Parameters

logical_volume_name
Is the name of the logical volume you want to specify as a virtual shared disk. This logical volume must reside on the global volume group indicated. The length of the name must be less than or equal to 15 characters.

global_group_name
Is the name of the globally-accessible volume group previously defined by the **vsdvg** command where you want to specify a virtual shared disk. The length of the name must be less than or equal to 31 characters.

vsd_name
Specifies a unique name for the new virtual shared disk. This name must be unique within the RSCT peer domain, and, in order to avoid possible future naming conflicts, should also be unique across the overall cluster. The suggested naming convention is **vsdnn***gvg_name*. The length of the name must be less than or equal to 31 characters.

Note: If you specify a *vsd_name* that is already the name of another device, the **cfgvsd** command will be unsuccessful for that virtual shared disk. This error ensures that the special device files created for the name do not overlay and destroy files of the same name representing some other device type (such as a logical volume).

Security

You must have `root` authority to run this command.

Restrictions

You must issue this command from a node in the peer domain that has an active recoverable virtual shared disk subsystem.

Examples

To fence the virtual shared disks vsd1 and vsd2 from node 5, enter:

```
fencevsd -v vsd1,vsd2 -n 5
```

Location

/opt/rsct/vsd/bin/fencevsd

ff Command

Purpose

Lists the file names and statistics for a file system.

Syntax

ff [**-a** *Number*] [**-c** *Number*] [**-I**] [**-l**] [**-m** *Number*] [**-n** *File*] [**-o** *Options*] [**-p** *Prefix*] [**-s**] [**-u**] [**-V** *VFSName*] [**-i** *I-Number* [**,***I-Number ...*]] [*FileSystem* | *DeviceName*]

Description

The **ff** command reads the i-nodes in the file system specified by the *FileSystem* parameter and then writes information about them to standard output. It assumes the *FileSystem* is a file system, which is referenced in the **/etc/filesystems** file, and saves i-node data for files specified by flags.

The output from the **ff** command consists of the path name for each requested i-node number, in addition to other file information that you can request using the flags. The output is listed in order by i-node number, with tabs between all fields. The default line produced by the **ff** command includes the path name and i-node number fields. With all flags enabled, the output fields include path name, i-node number, size, and UID (user ID).

The *Number* parameter is a decimal number that specifies a number of days. It is prefixed by a + or - (plus or minus sign). Therefore, +3 means more than 3 days, -3 means less than 3 days, and 3 means 3 days, where a day is defined as a 24-hour period.

The **ff** command lists only a single path name out of many possible ones for an i-node with more than one link, unless you specify the **-l** flag. With the **-l** flag, the **ff** command lists all links.

Flags

Item	Description
-a *Number*	Displays the file if it has been accessed within the number of days specified by the *Number* parameter.
-c *Number*	Displays the file if its i-node has been changed within the number of days specified by the *Number* parameter.
-i *I-Number*	Displays the files corresponding to the i-node numbers specified by the *I-Number* parameter. The i-node numbers listed must be separated by a comma.
-I	(This flag is an uppercase i.) Does not display the i-node after each path name.

Item	Description
-l	(This flag is a lowercase L.) Additionally displays a list of pathnames for files with more than one link.
-m *Number*	Displays the file if it has been modified within the number of days specified by the *Number* parameter.
-n *File*	Displays the file if it has been modified more recently than the file specified by the *File* parameter.
-o *Options*	Specifies a comma-separated list of implementation-specific options for a virtual file system. The following options are specific to the enhanced journaled file system (JFS2): **-o snapshot=***snapName* Specifies the name of the internal snapshot subject to the **ff** command. The file system owning the snapshot must be mounted.
-p *Prefix*	Adds the prefix specified by the *Prefix* parameter to each path name. The default prefix is **.** (dot).
-s	Writes the file size, in bytes, after each path name.
-u	Writes the owner's login name after each path name.
-V *VFSName*	Instructs the **ff** command to assume the file system is of type *VFSName*, overriding the value in the **/etc/filesystems** file.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list the path names of all files in a given file system, enter:

   ```
   ff    -I    /dev/hd0
   ```

   This displays the path names of the files on the /dev/hd0 device. If you do not specify the **-I** flag, the **ff** command also displays the i-node number of each file.
2. To list files that have been modified recently, enter:

   ```
   ff    -m    -2    -u    /dev/hd0
   ```

   This displays the path name, i-node number, and owner's user name (the -u flag) of each file on the /dev/hd0 device that has been modified within the last two days (-m -2).
3. To list files that have *not* been used recently, enter:

   ```
   ff    -a    +30    /dev/hd0
   ```

   This displays the path name and i-node of each file that was last accessed more than 30 days ago (-a +30).
4. To find out the paths corresponding to certain i-node numbers, enter:

   ```
   ff    -l    -i    451,76    /dev/hd0
   ```

   This displays all the path names (-l) associated with i-nodes 451 and 76.

Files

Item	Description
/etc/vfs	Contains descriptions of virtual file system types.
/etc/filesystems	Lists the known file systems and defines their characteristics.

fg Command

Purpose

Runs jobs in the foreground.

Syntax

fg [*JobID*]

Description

If job control is enabled, the **fg** command moves a background job in the current environment into the foreground. Use the *JobID* parameter to indicate a specific job to be run in the foreground. If this parameter is not supplied, the **fg** command uses the job most recently suspended, placed in the background, or run as a background job.

The *JobID* parameter can be a process ID number, or you can use one of the following symbol combinations:

Item	Description
%*Number*	Refers to a job by the job number.
%*String*	Refers to a job whose name begins with the specified string.
%?*String*	Refers to a job whose name contains the specified string.
%+ OR **%%**	Refers to the current job.
%-	Refers to the previous job.

Using the **fg** command to place a job into the foreground removes the job's process ID from the list of those known by the current shell environment.

The **/usr/bin/fg** command does not work when operating in its own command execution environment, because that environment does not have applicable jobs to manipulate. For this reason, the **fg** command is implemented as a Korn shell or POSIX shell regular built-in command.

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	An error occurred.

If job control is disabled, the **fg** command exits with an error, and no job is placed in the foreground.

Examples

If the output of the **jobs -l** command shows the following job running in the background:

```
[1] + 16477RunningSleep 100 &
```

use the process ID to run the `sleep 100 &` command in the foreground by entering:

```
fg 16477
```

The screen displays:

```
sleep
```

Files

Item	Description
/usr/bin/ksh	Contains the Korn shell **fg** built-in command.
/usr/bin/fg	Contains the **fg** command.

fgrep Command

Purpose

Searches a file for a literal string.

Syntax

fgrep [**-h**] [**-i**] [**-s**] [**-u**] [**-v**] [**-w**] [**-x**] [**-y**] [[**-b**] [**-n**] | [**-c** | **-l** | **-q**]] [**-p** *Separator*] {*Pattern* | **-e** *Pattern* | **-f** *StringFile*} [*File...*]

Description

The **fgrep** command searches the input files specified by the *File* parameter (standard input by default) for lines that match a pattern. The **fgrep** command searches specifically for *Pattern* parameters that are fixed strings. The **fgrep** command displays the file that contains the matched line if you specify more than one file in the *File* parameter.

The **fgrep** command differs from the **grep** and **egrep** commands because it searches for a string instead of searching for a pattern that matches an expression. The **fgrep** command uses a fast and compact algorithm. The $, *, [, |, (,), and \ characters are interpreted literally by the **fgrep** command. These characters are not interpreted as parts of a regular expression, as they are interpreted in the **grep** and **egrep** command. Since these characters have special meaning to the shell, the entire string must be enclosed in single quotation mark ('...'). If no files are specified, the **fgrep** command assumes standard input. Normally, each line found is copied to the standard output. The file name is printed before each line found if there is more than one input file.

Notes:

1. The **fgrep** command is the same as the **grep** command with the **-F** flag, except that error and usage messages are different and the **-s** flag functions differently.
2. Lines are limited to 2048 bytes.
3. Paragraphs (under the **-p** flag) are currently limited to a length of 5000 characters.
4. Do not run the **grep** command on a special file because it produces unpredictable results.
5. Input lines must not contain the NULL character.
6. Input files must end with the new line character.
7. Although some flags can be specified simultaneously, some flags override others. For example, if you specify **-l** and **-n** together, only file names are written to standard output.

Flags

Flag	Description
-b	Precedes each line by the block number on which it was found. Use this flag to help find disk block numbers by context. The **-b** flag cannot be used with input from stdin or pipes.
-c	Displays only a count of matching lines.
-e *Pattern*	Specifies a pattern. It works like a simple pattern but is useful when the pattern begins with a **-** (minus sign).
-f *StringFile*	Specifies a file that contains strings. **Note:** To enhance the performance of the **fgrep** (or **grep -F**) command with input as a file that contains search patterns, export the **ENABLE_FGREP_AC** environment variable before you run the **fgrep** command. For example, you can run the following command to export this environment variable: `export ENABLE_FGREP_AC=""`
-h	Suppresses file names when multiple files are processed.
-i	Ignores the case of letters when comparing.
-l	Lists just the names of files (once) with matching lines. Each file name is separated by a new line character.

Flag	Description
-n	Precedes each line with its relative line number in the file.
-p *Separator*	Displays the entire paragraph that contains matched lines. Paragraphs are delimited by paragraph separators, as specified by the *Separator* parameter, which are patterns in the same form as the search pattern. Lines containing the paragraph separators are used only as separators; they are never included in the output. The default paragraph separator is a blank line.
-q	Suppresses all writing to standard output, regardless of matching lines. Exits with a 0 status if an input line is selected.
-s	Displays only error messages. It is useful for checking status.
-u	Causes output to be unbuffered.
-v	Displays all lines except those lines that match the specified pattern.
-w	Searches a word.
-x	Displays lines that match the pattern exactly with no additional characters.
-y	Ignores the case of letters when comparing.

Exit Status

This command returns the following exit values:

Item	Description
0	A match was found.
1	No match was found.
>1	A syntax error was found or a file was inaccessible (even if matches were found).

Examples

1. To search several files for a simple string of characters:

   ```
   fgrep strcpy *.c
   ```

   This searches for the string `strcpy` in all files in the current directory with names that end in the `.c` character string.

2. To count the number of lines that match a pattern:

   ```
   fgrep -c "{" pgm.c
   fgrep -c "}" pgm.c
   ```

   It displays the number of lines in `pgm.c` that contain left and right braces.

   If you do not put more than one { (left brace) or one } (right brace) on a line in your C programs, and if the braces are properly balanced, the two numbers displayed are usually the same if the proper conditions are met. If the numbers are not the same, you can display the lines that contain braces in the order that they occur in the file with:

   ```
   egrep {\|} pgm.c
   ```

3. To display the names of files that contain a pattern:

   ```
   fgrep -l strcpy *.c
   ```

   It searches the files in the current directory that end with `.c` and displays the names of those files that contain the `strcpy` string.

Files

File	Description
/usr/bin/fgrep	Contains the **fgrep** command.
/bin/fgrep	Symbolic link to the **fgrep** command.

file Command

Purpose

Determines the file type.

Syntax

To Classify the File Type

file [**-m** *MagicFile*] [**-d**] [**-h**] [**-i**] [**-M** *MagicFile*] [**-f** *FileList*] [*File...*]

To Check the Magic File for Format Errors

file -c [**-m** *MagicFile*]

Description

The **file** command reads the files specified by the *File* parameter or the *FileList* variable, performs a series of tests on each file, and attempts to classify them by type. The command then writes the file types to standard output. The file can be regular file, directory, FIFO(named pipe), block special, character special, symbolic link or sockets type.

- If it is a regular file and of zero length, it is identified as an empty file.
- If the file is a symbolic link, by default, the link is followed by file the symbolic link refers to.

If a file appears to be in ASCII format, the **file** command examines the first 1024 bytes and determines the file type. If a file does not appear to be in ASCII format, the **file** command further attempts to distinguish a binary data file from a text file that contains extended characters.

If the *File* parameter specifies an executable or object module file and the version number is greater than 0, the **file** command displays the version stamp. The **ld** command explains the use of **a.out** files.

If the language environment is the C programming language, the **file** command uses the **/etc/magic** file to identify files that have some sort of a magic number; that is, any file containing a numeric or string constant that indicates type.

However, if the language environment is some language other than the C programming language, the **file** command uses the **/usr/lib/nls/msg/<*language_env.*>/magic.cat** file to identify files with a magic number.

If the file does not exist, cannot be read or its file status could not be determined then, it is not considered as an error that affects the exit status. The output indicates that the file was processed but the type could not be determined.

When the **-i** flag is used, the following format shall be used to identify each operand, *file* specified:

```
"%s: %s\n", file, type
```

The values for *type* are unspecified except that in the POSIX locale, if *file* is identified as one of the types listed in the following table, *type* shall contain (but is not limited to) the corresponding string. Each space shown in the strings shall be exactly one *space*.

Table 5. File Utility Output Strings	
If *file* is a:	***type* shall contain the string:**
Directoy	directory
FIFO	fifo
Socket	socket
Block special	block special
Character special	character special
Executable binary	executable
Empty regular file	empty
Symbolic link	symbolic link to
ar archive library	archive
Extended *cpio* format	cpio archive
Extended *tar* format	tar archive
Shell script	commands text
C-language source	c program text
FORTRAN source	fortran program text

If *file* is identified as a symbolic link, the following alternative output format shall be used:

```
"%s: %s %s\n", file, type, contents of link"
```

If the file named by the *file* operand does not exist or cannot be read, the string `cannot open` shall be included as part of the *type* field, but this shall not be considered an error that affects the exit status. If the type of the file named by the *file* operand cannot be determined, the string `data` shall be included as part of the *type* field, but this shall not be considered an error that affects the exit status.

Flags

Item	Description
-c	Checks the specified magic file (the **/etc/magic** file, by default) for format errors. This validation is not normally done. File typing is not done under this flag.
-d	Applies any default system tests to the file.
-f *FileList*	Reads the specified file list. The file must list one file per line and must not contain leading or trailing spaces.
-h	When a symbolic link is encountered, identifies the file as a symbolic link. If the **-h** flag is not specified and *file* is a symbolic link that refers to a nonexistent file, *file* shall identify the file as a symbolic link, as if the **-h** flag had been specified.
-i	If a file is a regular file, does not attempt to classify the type of the file further, but identifies the file as specified in “Description” on page 1372.
-m *MagicFile*	Specifies the file name of the magic file (the **/etc/magic** file, by default).
-M *MagicFile*	Specifies the name of a file containing tests that shall be applied to a file in order to classify it. No default system tests shall be applied.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To display the type of information a file contains, enter:

   ```
   file myfile
   ```

   This displays the file type of *myfile* (such as directory, data, ASCII text, C-program source, and archive).
2. To display the type of each file named in a list of file names, enter:

   ```
   file -f filenames
   ```

   This displays the type of each file named in the `filenames` list. Each file name must appear alone on a line.

Note: To get customized messages from the **file** command, use a separate magic file with the **-m** option. It is not advisable to edit the read-only **/etc/magic** file.

Files

Item	Description
/usr/bin/file	Contains the **file** command.
/etc/magic	Contains the file type database.

filemon Command

Purpose

Monitors the performance of the file system, and reports the I/O activity on behalf of logical files, virtual memory segments, logical volumes, and physical volumes.

Syntax

filemon [**-d**] [**-i** *Trace_File* **-n** *Gensyms_File*] [**-o** *File*] [**-O** ***Levels***] [**-w**] [**-I** ***count:interval***] [**-P**] [**-T** *n*] [**-u**] [**-v**] [**-@** [*WparList* | **ALL**] [**-r** *RootString*] [**-A** **-x** *User_Command*]

Description

The **filemon** command monitors a trace of file system and I/O system events, and reports on the file and I/O access performance during that period.

In its normal mode, the **filemon** command runs in the background while one or more application programs or system commands are being run and monitored. The **filemon** command automatically starts and monitors a trace of the program file system and I/O events in real time. By default, the trace is started immediately; optionally, tracing might be deferred until you issue a **trcon** command. You can issue the **trcoff** and **trcon** commands while the **filemon** command is running to turn the monitoring off and on, as required. When tracing is stopped by a **trcstop** command, the **filemon** command generates an I/O activity report and exits.

The **filemon** command can also process a trace file that is previously recorded by the trace facility. The file and I/O activity report are based on the events recorded in that file.

To provide a complete understanding of file system performance for an application, the **filemon** command monitors file and I/O activity at four levels:

Item	Description
Logical file system	The **filemon** command monitors logical I/O operations on logical files. The monitored operations include all **read**, **write**, **open**, and **lseek** system calls, which might or might not result in actual physical I/O, depending on whether the files are already buffered in memory. I/O statistics are kept on a per-file basis. Calls to Asynchronous I/O system calls are not monitored by the `filemon` command, so the `filemon` logical file report does not include asynchronous I/O (AIO) requests.
Virtual memory system	The **filemon** command monitors physical I/O operations (that is, paging) between segments and their images on disk. I/O statistics are kept on a per-segment basis.
Logical volumes	The **filemon** command monitors I/O operations on logical volumes. I/O statistics are kept on a per-logical-volume basis.
Physical volumes	The **filemon** command monitors I/O operations on physical volumes. At this level, physical resource utilizations are obtained. I/O statistics are kept on a per-physical-volume basis.

Any combination of the four levels can be monitored, as specified by the command-line flags. By default, the **filemon** command only monitors I/O operations at the virtual memory, logical volume, and physical volume levels. These levels are all concerned with requests for real disk I/O.

The **filemon** command also generates a hotness report on the files, logical volumes, and physical volumes. The hotness report can be generated by using **–O** hot option. This report is supported only in automated offline and manual offline modes. Hotness report contains statistics of I/O operations of files, logical volumes, and physical volumes. This report helps you decide which files or logical volumes to move to any drive, with a different I/O characteristic based on the hotness of the file/logical volume. The hotness is determined based on number of read operations, average number of bytes read per read operation, number of read sequences and the average sequence length.

The **filemon** command writes its report to standard output or to a specified file. By default the report contains a summary of the I/O activity for each of the levels being monitored. Detailed report is printed only if the **-O detailed** flag is enabled. The summary and detailed report contents are described in the Reports section.

Notes:

1. The reports produced by the **filemon** command can be long. Consequently, the **-o** option is to be used to write the report to an output file. When a physical device is opened and accessed directly by an application, only reads and writes of complete 512-byte blocks are reflected in the report. “Short” reads and writes, used by the device driver to issue device commands and read device status, are ignored. CD-ROMs do not have concentric “tracks” or “cylinders,” as in hard files. (There is one spiral track.) Consequently, it is not possible to report distance statistics for CD-ROMs in terms of cylinders.
2. The **-u** flag is used to generate reports on files opened before the start of the **trace** daemon. Some of this data can be useful, but much of it applies to daemons and other unrelated activity. This background information can be overwhelming, especially on large systems. If the **/unix** file and the running kernel are not the same, then the kernel addresses are incorrect, causing the **filemon** command to exit. When using the **filemon** command from within a shell script, view the contents of the **filemon** output file after a slight delay. The **filemon** command might take a few seconds to produce this report.
3. When you specify relative paths in an I/O process program to read or write a file, the **filemon** command interprets this relative path as the directory from where the **filemon** command was run.

In such cases, the I/O activity report might not display the correct volume information (i-node) for that file. To avoid this problem, use the complete path in all the I/O process programs.

4. The **filemon** command does not support the solid state drive (SSD) disks. Hence, the **filemon** command does not report statistics of the SSD disks.

System Trace Facility

The **filemon** command obtains raw I/O performance data using the system trace facility. Currently, the trace facility only supports one output stream. Consequently, only one **filemon** or trace process can be active at a time. If another **filemon** or trace process is already running, the **filemon** command responds with the message:

```
/dev/systrace: Device busy
```

While monitoring the I/O-intensive applications, the **filemon** command might not be able to consume trace events as fast as they are produced in real time. When that happens, the error message:

```
Trace kernel buffers overflowed, N missed entries
```

is displayed on `stderr`, indicating how many trace events were lost while the trace buffers were full. The **filemon** command continues monitoring the I/O activity, but the accuracy of the report is diminished to some unknown degree. One way to prevent overflow is to monitor fewer levels of the file and I/O subsystems: the number of trace events generated is proportional to the number of levels monitored. Additionally, the trace buffer size can be increased using the **-T** option, to accommodate larger bursts of trace events before overflow. Remember that increasing the trace buffer size results in more pinned memory, and therefore might affect I/O and paging behavior.

In memory-constrained environments (where demand for memory exceeds supply), the **-P** option can be used to pin the text and data pages of the real-time **filemon** process in memory so the pages cannot be swapped out. If the **-P** option is not used, letting the **filemon** process to be swapped out, the progress of the **filemon** command might be delayed to the point where it cannot process trace events fast enough. This situation leads to trace buffer overflow as described previously. Consequently, pinning this process takes memory away from the application (although the **filemon** command is not a large program, its process image can consume up to 500KB).

The **-i** *Trace_File* and **-n** *Gensyms_File* flags let offline processing by **filemon** of trace data files created by the **trace** command. Both flags must be supplied if either is present. These flags are useful when it is necessary to postprocess a trace file from a remote machine or perform the trace data collection at one time and postprocess it at another time. The flags are also useful when system load is high and trace hooks are being missed by **filemon**. You can use these flags for automated offline mode.

The **-r** *RootString* flag deprecates the **-i** *Trace_File* flag and the **-n** *Gensyms_File* flag. Apart from using the **-r** *RootString* flag for offline processing, the same can be used along with the **-A** flag which enables automated offline mode.

The **gensyms** file (containing file system information) must be used from the machine that the trace came from. Also, it is wise to run **gensyms** at close to the same time that the system trace file is created, so that the system configuration is the same for both.

Trace hooks relevant to **filemon** must be collected by the **trace** command and are specified by the **trace -j** flag. The relevant trace hooks are listed when **filemon** is started with the **-v** flag. The **gensyms** command with **-F** option is then run, with its output saved in *Gensyms_File* to collect additional information for **filemon**. The **-F** option is used with the gensyms command to collect the device information for physical and logical volumes. It is also used to get the virtual file system information used by offline **filemon**. Then this file and the *Gensyms_File* might be provided to **filemon**.

Reports

Each report generated by the **filemon** command has a header that identifies the date, the machine ID, and the length of the monitoring period, in seconds. The processor utilization during the monitoring period is also reported.

Next, summary reports are generated for each of the file system levels being monitored. By default, the logical file and virtual memory reports are limited to the 20 most active files and segments, as measured by the total amount of data transferred. If the **-v** flag is specified, activity for all files and segments is reported. There is one row for each reported file, segment, or volume. The columns in each row for the four summary reports are described in the following lists:

Most Active Files Report

Column	Description
#MBS	Total number of megabytes transferred to or from file. The rows are sorted by this field, in decreasing order.
#opns	Number of times the file was opened during measurement period.
#rds	Number of read system calls made against file.
#wrs	Number of write system calls made against file.
file	Name of file (full path name is in detailed report).
volume:inode	Name of volume that contains the file, and the i-node number of the file. This field can be used to associate a file with its corresponding persistent segment, shown in the virtual memory I/O reports. This field might be blank; for example, for temporary files created and deleted during execution.

Most Active Segments Report

Item	Description
Column	**Description**
#MBS	Total number of megabytes transferred to/from segment. The rows are sorted by this field, in decreasing order.
#rpgs	Number of 4096-byte pages read into segment from disk (that is, page).
#wpgs	Number of 4096-byte pages written from segment to disk (page out).
segid	Internal ID of segment.
segtype	Type of segment: working segment, persistent segment (local file), client segment (remote file), page table segment, system segment, or special persistent segments containing file system data (log, root directory, .inode, .inodemap, .inodex, .inodexmap, .indirect, .diskmap).
volume:inode	For persistent segments, name of volume that contains the associated file, and the i-node number of the file. This field can be used to associate a persistent segment with its corresponding file, shown in the file I/O reports. This field is blank for non-persistent segments. **Note:** The virtual memory analysis tool, **svmon** can be used to display more information about a segment, given its segment ID (segid), as follows: `svmon -S <segid>`

Most Active Logical Volumes Report

Item	Description
Column	**Description**
util	Utilization of the volume (fraction of time busy). The rows are sorted by this field, in decreasing order.
#rblk	Number of 512-byte blocks read from the volume.
#wblk	Number of 512-byte blocks written to the volume.
KB/sec	Total transfer throughput, in Kilobytes per second.
volume	Name of volume.
description	Contents of volume: either a file system name, or logical volume type (paging, jfslog, boot, or sysdump). Also, indicates whether the file system is fragmented or compressed.

Most Active Physical Volumes Report

Item	Description
Column	**Description**
util	Utilization of the volume (fraction of time busy). The rows are sorted by this field, in decreasing order.
#rblk	Number of 512-byte blocks read from the volume.
#wblk	Number of 512-byte blocks written to the volume.
KB/sec	Total volume throughput, in Kilobytes per second.
volume	Name of volume.
description	Type of volume, for example, 120MB disk, 355MB SCSI, or CDROM SCSI. **Note:** Logical volume I/O requests start before, and end after, physical volume I/O requests. For that reason, total logical volume utilization appears to be higher than total physical volume utilization.

Most Active Files Process-Wise Report

Item	Description
Column	**Description**
#MBS	Total number of megabytes transferred to or from the file. The rows are sorted by this field, in decreasing order.
#opns	Number of times the file was opened during measurement period.
#rds	Number of read system calls made against file.
#wrs	Number of write system calls made against file.
file	Name of file (full path name is in detailed report).
PID	ID of the process which opened the file.
Process	Name of the process which opened the file.
TID	ID of the thread which opened the file.

Most Active Files Thread-Wise Report

Item	Description
Column	**Description**

Item	Description
#MBS	Total number of megabytes transferred to or from the file. The rows are sorted by this field, in decreasing order.
#opns	Number of times the file was opened during measurement period.
#rds	Number of read system calls made against file.
#wrs	Number of write system calls made against file.
file	Name of file (full path name is in detailed report).
TID	ID of the thread which opened the file.
Process	Name of the process which opened the file.
PID	ID of the process which opened the file.

Finally, detailed reports are generated for each of the file system levels being monitored. By default, the logical file and virtual memory reports are limited to the 20 most active files and segments, as measured by the total amount of data transferred. If the **-v** flag is specified, activity for all files and segments is reported. There is one entry for each reported file, segment, or volume.

Some of the fields report a single value, others report statistics that characterize a distribution of many values. For example, response time statistics are kept for all read or write requests that were monitored. The average, minimum, and maximum response times and the standard deviation of the response times are reported. The standard deviation is used to show how much the individual response times deviated from the average. Roughly two-thirds of the sampled response times are between `average - standard deviation` and `average + standard deviation`. If the distribution of response times is scattered over a large range, the standard deviation will be large compared to the average response time. The four detailed reports are described in the following lists:

Detailed File Statistics Report

Item	Description
Column	**Description**
FILE	Name of the file. The full path name is given, if possible.
volume	Name of the logical volume/file system containing the file.
inode	I-node number for the file within its file system.
opens	Number of times the file was opened while monitored.
total bytes xfrd	Total number of bytes read/written to/from the file.
reads	Number of read calls against the file.
read sizes (bytes)	The read transfer-size statistics (avg/min/max/sdev), in bytes.
read times (msec)	The read response-time statistics (avg/min/max/sdev), in milliseconds.
writes	Number of write calls against the file.
write sizes (bytes)	The write transfer-size statistics.
write times (msec)	The write response-time statistics.
seeks	Number of **lseek** subroutine calls.

Detailed VM Segment Statistics Report

Item	Description
Column	**Description**

Item	Description
SEGMENT	Internal segment ID.
segtype	Type of segment contents.
segment flags	Various segment attributes.
volume	For persistent segments, the name of the logical volume containing the corresponding file.
inode	For persistent segments, the i-node number for the corresponding file.
reads	Number of 4096-byte pages read into the segment (that is, paged in).
read times (msec)	The read response-time statistics (avg/min/max/sdev), in milliseconds.
read sequences	Number of read sequences. A sequence is a string of pages that are read (paged in) consecutively. The number of read sequences is an indicator of the amount of sequential access.
read seq. lengths	Statistics describing the lengths of the read sequences, in pages.
writes	Number of pages written from the segment (that is, paged out).
write times (msec)	Write response time statistics.
write sequences	Number of write sequences. A sequence is a string of pages that are written (paged out) consecutively.
write seq.lengths	Statistics describing the lengths of the write sequences, in pages.

Detailed Logical/Physical Volume Statistics Reports

Item	Description
Column	**Description**
VOLUME	Name of the volume.
description	Description of the volume. (Describes contents, if dealing with a logical volume; describes type, if dealing with a physical volume.)
reads	Number of read requests made against the volume.
read sizes (blks)	The read transfer-size statistics (avg/min/max/sdev), in units of 512-byte blocks.
read times (msec)	The read response-time statistics (avg/min/max/sdev), in milliseconds.
read sequences	Number of read sequences. A sequence is a string of 512-byte blocks that are read consecutively and indicate the amount of sequential access.
read seq. lengths	Statistics describing the lengths of the read sequences, in blocks.
writes	Number of write requests made against the volume.
write sizes (blks)	The write transfer-size statistics.
write times (msec)	The write-response time statistics.
write sequences	Number of write sequences. A sequence is a string of 512-byte blocks that are written consecutively.
write seq. lengths	Statistics describing the lengths of the write sequences, in blocks.
seeks	Number of seeks that preceded a read or write request; also expressed as a percentage of the total reads and writes that required seeks.

Item	Description
seek dist (blks)	Seek distance statistics, in units of 512-byte blocks. In addition to the usual statistics (avg/min/max/sdev), the distance of the initial seek operation (assuming block 0 was the starting position) is reported separately. This seek distance is sometimes large, so it is reported separately to avoid skewing the other statistics.
seek dist (cyls)	(Hard files only.) Seek distance statistics, in units of disk cylinders.
time to next req	Statistics (avg/min/max/sdev) describing the length of time, in milliseconds, between consecutive read or write requests to the volume. This column indicates the rate at which the volume is being accessed.
throughput	Total volume throughput, in Kilobytes per second.
utilization	Fraction of time the volume was busy. The entries in this report are sorted by this field, in decreasing order.

Detailed Process-wise Statistics Report

Item	Description
Column	**Description**
Process Id	ID of the process which opened the file.
Name	Name of the file opened including the path.
Thread Id	ID of the thread which opened the file.
Total Bytes	Total number of bytes read or written.
# of seeks	Number of seeks.
# of reads	Number of read operations.
read errors	Number of read errors.
# of writes	Number of write operations.
Bytes Read	Number of bytes read. **min** Minimum number of bytes read at a time. **avr** Average number of bytes read at a time. **max** Maximum number of bytes read at a time.
Bytes Written	Number of bytes written. **min** Minimum number of bytes written at a time. **avr** Average number of bytes written at a time. **max** Maximum number of bytes written at a time.
Read Time	Time spent in read operations.
Write Time	Time spent in write operations.

Detailed Thread-wise Statistics Report

Item	Description
Column	**Description**
Thread Id	ID of the thread which opened the file.
Name	Name of the file opened including the path.
Process Id	ID of the thread which opened the file.
Total Bytes	Total number of bytes read or written.
# of seeks	Number of seeks.
# of reads	Number of read operations.
read errors	Number of read errors.
# of writes	Number of write operations.
Bytes Read	Number of bytes read. **min** Minimum number of bytes read at a time. **avr** Average number of bytes read at a time. **max** Maximum number of bytes read at a time.
Bytes Written	Number of bytes written. **min** Minimum number of bytes written at a time. **avr** Average number of bytes written at a time. **max** Maximum number of bytes written at a time.
Read Time	Time spent in read operations.
Write Time	Time spent in write operations.

Collated Report Format

Item	Description
ID	**process** ID of the process which did read or write operation. **thread** ID of the thread which did read or write operation. **CPU** ID of the CPU in which read or write operation was performed.
transaction type	Type of transaction: SCSI, SSA, and so on.
time	**bstart event** Time at which the **bstart** event was started. **iodone event** Time at which the I/O operation was completed. **duration** Total time duration of the I/O operation.

Item	Description
read/write	Type of operation: read or write.
physical block address	Physical block address.
access pattern	Type of access: pattern, sequential, or random.
physical block size	Physical block size.
volume name or address	**physical** Physical volume name or address. **logical** Logical volume name or address.
Transaction index	Unique ID to identify the transaction.
time	**event** Time at which the event started. **extend** Time at which the event extended.
ID	**process** ID of the process which performed the transaction. **thread** ID of the thread which performed the transaction. **CPU** ID of the CPU in which the transaction was performed.
protocol stage	Displays the breakup of the events.
name	Name of device, buffer, or block; or byte count.
address/count	Address or byte count of device, buffer, or block.
access pattern	Type of access pattern: sequential or random.
label	Volume types or transfer flags.
values	Volume names or flag values.

Hotness Report

The hotness report consists of three sections: information section, summary section, and hotness reports section. The information section contains the system model, the **filemon** command used, and the **trace** command used. The summary section contains: total number of read or write operations, total time taken, total data read or written, and the CPU utilization.

Hot files report

Item	Description
Column	**Description**
Name	Name of the file.
Size	Size of the file. The default unit is MB. The default unit is overridden by the unit specified by **–O** unit option.

Item	Description
CAP_ACC	Capacity accessed. This value is the unique data accessed in the file. The default unit is MB. The default unit is overridden by the unit specified by **–O** unit option.
IOP/#	Number of I/O operations per unit of data accessed. The unit of data is taken from **–O** unit option. The default is MB. Examples of value for this column are 2560/T, 256/G, 0.256/M, 0.000/K. The letters K, M, G and T stand for KB, MB, GB, and TB.
LV	Name of logical volume the file belongs to. If this information cannot be obtained, a **"-"** is reported.
#ROP	Total number of read operations happened on the file.
#WOP	Total number of write operations happened on that file.
B/ROP	<minimum, average, maximum> number of bytes read per read operation.
B/WOP	<minimum, average, maximum> number of bytes read per write operation.
RTIME	<minimum, average, maximum> time taken per read operation in milliseconds.
WTIME	<minimum, average, maximum> time taken per write operation in milliseconds.
Seqlen	<minimum, average, maximum> length of read sequences.
#Seq	Number of read sequences. A sequence is a string of 4K pages that are read (paged in) consecutively. The number of read sequences is an indicator of the amount of sequential access.

Hot Logical Volumes Report

Item	Description
Column	**Description**
Name	Name of the logical file.
Size	Size of the logical volume. The default unit is MB. The default unit is overridden by the unit specified by –O unit option. If this value cannot be obtained, a “-“ is reported.
CAP_ACC	Capacity accessed. This value is the unique data accessed in the file. The default unit is MB. The default unit is overridden by the unit specified by **–O** unit option.
IOP/#	Number of I/O operations per unit of data accessed. The unit of data is taken from **–O** unit option. The default is MB. Examples of value for this column are 2560/T, 256/G, 0.256/M, 0.000/K. The letters K, M, G and T stand for KB, MB, GB, and TB respectively.
#Files	Number of files accessed in this logical volume.
#ROP	Total number of read operations happened on the logical volume.
#WOP	Total number of write operations happened on that logical volume.
B/ROP	<minimum, average, maximum> number of bytes read per read operation.
B/WOP	<minimum, average, maximum> number of bytes read per write operation.
RTIME	<minimum, average, maximum> time taken per read operation in milliseconds.
WTIME	<minimum, average, maximum> time taken per write operation in milliseconds.
Seqlen	<minimum, average, maximum> length of read sequences.
#Seq	Number of read sequences. A sequence is a string of 4K pages that are read (paged in) consecutively. The number of read sequences is an indicator of the amount of sequential access.

Hot Physical Volumes Report

Item	Description
Column	**Description**
Name	Name of the physical volume.
Size	Size of the physical volume. The default unit is MB. The default unit is overridden by the unit specified by **–O** unit option.
CAP_ACC	Capacity accessed. This value is the unique data accessed in the file. The default unit is MB. The default unit is overridden by the unit specified by **–O** unit option.
IOP/#	Number of I/O operations per unit of data accessed. The unit of data is taken from **–O** unit option. The default is MB. Examples of value for this column are 2560/T, 256/G, 0.256/M, 0.000/K. The letters K, M, G and T stand for KB, MB, GB, and TB respectively.
#ROP	Total number of read operations happened on the physical volume.
#WOP	Total number of write operations happened on that physical volume.
B/ROP	<minimum, average, maximum> number of bytes read per read operation.
B/WOP	<minimum, average, maximum> number of bytes read per write operation.
RTIME	<minimum, average, maximum> time taken per read operation in milliseconds.
WTIME	<minimum, average, maximum> time taken per write operation in milliseconds.
Seqlen	<minimum, average, maximum> length of read sequences.
#Seq	Number of read sequences. A sequence is a string of 512-byte blocks that are read consecutively. The number of read sequences is an indicator of the amount of sequential access.

Each of the described hotness reports is repeated multiple times based on the sort field.

The different hotness reports based on different sort fields are:

1. hotness report sorted on key factor
2. hotness report sorted on CAP_ACC
3. hotness report sorted on IOP/#
4. hotness report sorted on #ROP
5. hotness report sorted on #WOP
6. hotness report sorted on RTIME
7. hotness report sorted on WTIME

Each of the reports is sorted in descending order of the corresponding sort field.

If you specify the **–O hot=r** option then only read operations-based reports and report based on key factor are generated, that is, report number 1, 4, and 6 are generated.

If the user specifies **–O hot=w** option then only write operations-based reports and report based on key factor are generated, that is, report number 1, 5, and 7 are generated.

The key factor is determined by the values of following columns: **#ROP**, **B/ROP**, **Seqlen** and **#Seq**.

Flags

Item	Description
-i *Trace_File*	Reads the I/O trace data from the specified *Trace_File*, instead of from the real-time trace process. The **filemon** report summarizes the I/O activity for the system and period represented by the trace file. This option is deprecated. Use the **-r** *RootString* flag instead. For the report to be accurate, the trace file must contain all the hooks required by the **filemon** command. The **-n** option must also be specified.
-n *Gensyms_File*	Specifies a *Gensyms_File* for offline trace processing. This file is created by running the gensyms command with **-f** option and redirecting the output to a file, as follows: `gensyms -F > file` The **-i** option must also be specified. The **-n** flag is depredated. Use the **-r** *RootString* flag instead.
-o *File*	Writes the I/O activity report to the specified *File*, instead of to the **stdout** file.
-d	Starts the **filemon** command, but defers tracing until the **trcon** command is run by the user. By default, tracing is started immediately.
-T *n*	Sets the trace buffer size of the kernel to *n* bytes. The default size is 64 000 bytes per CPU. The buffer size can be increased to accommodate larger bursts of events, if any. (A typical event record size is 30 bytes.) **Note:** The trace driver in the kernel uses double buffering, so in fact there are two buffers allocated of size *n* bytes. Also, note that these buffers are pinned in memory, so they are not subject to paging. Large buffers might affect the performance of paging and other I/O.
-P	Pins monitor process in memory. The **-P** flag causes the **filemon** command text and data pages to be pinned in memory for the duration of the monitoring period. This flag can be used to ensure that the real-time **filemon** process is not paged out when running in a memory-constrained environment.
-v	Prints extra information in the report. The most significant effect of the **-v** flag is that all logical files and all segments that were accessed are included in the I/O activity report, instead of only the 20 most active files and segments.
-A -x *User_Command*	Turns on automated offline mode. You must use the **-x** flag along with the **-A** flag where the trace is collected until the specified user command finishes its execution. The typical example of user command is `sleep 10`.
-r *RootString*	If you combine this flag with the **-A** flag, the **filemon** command stores the trace data in the **RootString.trc** file and generates a **gensyms** file and stores it in the **RootString.syms** file. When this option is enabled in the absence of the **-A** flag, the **filemon** command posts processing the **RootString.trc** file and the **RootString.syms** file to generate offline report. This option deprecates the existing **-n** and **-i** flags. The **filemon** command continues to support the **-i** flag and the **-n** flag for binary compatibility.

Item	Description
-O *Levels*	Monitors only the specified file system levels. Valid comma-separated options are: **abbreviated** Produces a list of transactions in abbreviated format, one line per transaction (replaces old "subpar" tool). This option is supported only in offline mode and cannot be combined with any other **-O** options. **collated** Produces a list of transactions in collated format: events are collected together per transaction. This option is supported only in offline mode and cannot be combined with any other **-O** options. **detailed** Detailed report is generated along with statistical summary mode and cannot be combined with the **abbreviated** option or the **collated** option. **lf=num** Displays only the specified number of logical file entries and cannot be combined with **-O abbreviated** flag or the **-O collated** flag. If the *num* argument is not specified, it displays all the entries. **vm=num** Displays only the specified number of virtual memory entries and cannot be combined with the **-O abbreviated** flag or the **-O collated** flag. If the *num* argument is not specified, it displays all the entries. **lv=num** Displays only the specified number of logical volume entries and cannot be combined with the **-O abbreviated** flag or the **-O collated** flag. If the *num* argument is not specified, it displays all the entries. **pv=num** Displays only the specified number of physical volume entries and cannot be combined with the **-O abbreviated** flag or the **-O collated** flag. If the *num* argument is not specified, it displays all the entries. **hot=r|w** Generates the hotness report. If **hot=r** specified then hotness reports based on read operations only are generated. If **hot=w** specified the hotness reports based on write operations only are generated. **sz=num** Specifies the maximum size of the files accessed to be reported in the hotness report. Unit for this value is specified by **–O** unit option. The default unit is MB. **unit={KB|MB|GB|TB}** Specifies the unit to be used with **sz** option and the unit to be used with **CAP_ACC** and Size fields in hotness report. **th=num** Displays only the specified number of thread statistics entries and cannot be combined with the **-O abbreviated** flag or the **-O collated** flag. If the *num* argument is not specified, it displays all the entries. **pr=num** Displays only the specified number of process statistics entries and cannot be combined with the **-O abbreviated** flag or the **-O collated** flag. If the *num* argument is not specified, it displays all the entries. **all=num** Sets **lf=num**, **vm=num**, **lv=num**, **pv=num**, **ts=num** and overwrites the old values for the options of **lf**, **vm**, **lv**, **pv**, **th**, and **pr**. This option cannot be combined with the **-O abbreviated** flag or the **-O collated** flag. If the *num* argument is not specified, it displays all the entries and this is the default option.

Item	Description
	The **vm**, **lv**, and **pv** levels are implied by default when you run the **filemon -O** command in the global WPAR without the **-@** flag. The **lf** level is implied by default when you run the **filemon -O** command in a WPAR or when you use the **-@** flag. If the *num* argument is not specified, the default is to display all the entries of that section. The *num* argument is not supported in **abbreviated** and **collated** formats and it is supported only in statistical summary. If the **-O detailed** flag is specified, the report is in statistical summary format along with detailed report in both online and offline mode. The default mode of operation for the **filemon** command is changed from Summary and Detailed Statistical report to Summary only Statistical Report. If the **filemon** command is called without any option or just with the **-O** flag with any combination of the **lf**, **vm**, **lv**, **pv**, **pr**, **th**, or **all** option, only summary report is displayed unless otherwise the **-O detailed** flag is specified.
-u	Reports on files that were opened before the start of the **trace** daemon. The process ID (PID) and the file descriptor (FD) are substituted for the file name. **Note:** Since PIDs and FDs are reusable, it is possible to see different files reported with the same name field.

Item	Description
-w	Prints the hotness report in wide format. This option is valid only if the **–O** hot option is specified.
-I *count:interval*	Specifies the count and interval to be used for multi-snapshot tracing. If this option is specified, **count** number of snapshots of trace is collected with a gap of **interval** seconds between two snapshots. This option is valid only in automated offline mode with **–O** hot option specified.
-@ [*WparList* | **ALL**]	Reports are limited to the list of WPARs passed by the argument.

Examples

1. To monitor the physical I/O activity of the virtual memory, logical volume, and physical volume levels of the file system, enter:

   ```
   filemon
   ```

   The **filemon** command automatically starts the system trace and puts itself in the background. After this command, enter the application programs and system commands to be run at this time, then enter:

   ```
   trcstop
   ```

   After the **trcstop** command is issued, the I/O activity report is displayed on standard output (but probably scrolls off the screen). The virtual memory I/O report is limited to the 20 segments that incurred the most I/O.
2. To monitor the activity at all file system levels, and write the report to the `fmon.out` file, enter:

   ```
   filemon -o fmon.out -O all
   ```

   The **filemon** command automatically starts the system trace and puts itself in the background. After this command, enter the application programs and system commands to be run at this time, then enter:

   ```
   trcstop
   ```

   After the **trcstop** command is issued, the I/O activity report is written to the `fmon.out file`. All four levels of the file and I/O system (the logical file, virtual memory, logical volume, and physical volume levels) are monitored. The logical file and virtual memory I/O reports are limited to the 20 files and segments (respectively) that incurred the most I/O.
3. To monitor the activity at all file system levels and write a verbose report to the `fmon.out` file, enter:

   ```
   filemon -v -o fmon.out -O all
   ```

   The **filemon** command automatically starts the system trace and puts itself in the background. After this command, enter the application programs and system commands to be run at this time, then enter:

   ```
   trcstop
   ```

   This example is similar to the previous example, except a verbose report is generated on the `fmon.out` file. The primary difference is that the **filemon** command indicates the steps that it is taking to start the trace, and the summary and detailed reports include all files and segments that incurred any I/O (there might be many), instead of just the top 20.
4. To report on I/O activity captured by a previously recorded trace session, enter:

   ```
   filemon -i trcfile | pg
   ```

In this example, the **filemon** command reads file system trace events from the input file `trcfile`. Since the trace data is already captured on a file, the **filemon** command does not put itself in the background to let application programs run. After the entire file is read, an I/O activity report for the virtual memory, logical volume, and physical volume levels will be displayed on standard output (which, in this example, is piped to pg).

5. To monitor the I/O activity for logical and physical volumes only, while controlling the monitored intervals using the **trcon** and **trcoff** commands, enter:

```
filemon -d -o fmon.out -O pv,lv
```

The **filemon** command automatically starts the system trace and puts itself in the background. After this command, you can enter the unmonitored application programs and system commands to be run at this time, then enter:

```
trcon
```

After this command, you can enter the monitored application programs and system commands to be run at this time, then enter:

```
trcoff
```

After this command, you can enter the unmonitored application programs and system commands to be run at this time, then enter:

```
trcon
```

After this command, you can enter the monitored application programs and system commands to be run at this time, then enter:

```
trcstop
```

In this example, the **-O** flag is used to restrict monitoring to logical and physical volumes only. Only those trace events that are relevant to logical and physical volumes are enabled. Also, as a result of using the **-d** flag, monitoring is initially deferred until the **trcon** command is issued. System tracing can be intermittently disabled and reenabled using the **trcoff** and **trcon** commands, so that only specific intervals are monitored.

6. To run **filemon** in offline mode, run the **trace** and **gensyms** commands separately, then use the output from those commands as input to the **filemon** command, as follows:

```
trace -a -T 768000 -L 10000000 -o trace.out -j
000,000,001,002,003,005,006,139,102,10C,106,00A,107,
101,104,10D,15B,12E,130,163,19C,154,3D3,1BA,1BE,1BC,10B,221,1C9,222,228,232,45B
```

Run the monitored application programs and system commands, then enter:

```
trcstop
```

Create the **gensyms** file:

```
gensyms -F > gensyms.out
```

Then run **filemon** with both **-i** and **-n** flags:

```
filemon -i trace.out -n gensyms.out -O all
```

7. To generate hotness report in automated offline mode, with unit of data as megabytes, use the following command:

```
filemon -O hot,unit=MB -r <rootstring> -A-x "<user command>"
```

8. To generate hotness report with three snapshots of trace in 5-seconds interval, run the following command:

```
filemon -O hot -r <rootstring> -A-x "<user command>" -I 3:5
```

9. To generate hotness report in offline mode:

```
filemon -r <rootstring> -O hot
```

fileplace Command

Purpose

Displays the placement of file blocks within logical or physical volumes.

Syntax

fileplace [{ **-l** | **-p** [**-o** *FragOffset*] [**-n** *FragNumber*] }[**-i**] [**-v**][**-a**]] *File* | [**-m** *LogicalVolumeName*]

Description

The **fileplace** command displays the placement of a specified file within the logical or physical volumes containing the file.

By default, the **fileplace** command lists to standard output the ranges of logical volume fragments allocated to the specified file. The order in which the logical volume fragments are listed corresponds directly to their order in the file. A short header indicates the file size (in bytes), the name of the logical volume in which the file lies, the block size (in bytes) for that volume, the fragment size in bytes, and the compression, indicating if the file system is compressed or not.

Occasionally, portions of a file may not be mapped to any fragments in the volume. These areas, whose size is an integral number of fragments, are implicitly zero-filled by the file system. The **fileplace** command indicates which areas in a file have no allocated fragments.

Optionally, the **fileplace** command also displays:

- Statistics indicating the degree to which the file is spread within the volume.
- The indirect block addresses for the file.
- The file's placement on physical (as opposed to logical) volume, for each of the physical copies of the file.

Notes:

1. The **fileplace** command is not able to display the placement of remote Network File System (NFS) files. If a remote file is specified, the **fileplace** command returns an error message. However, the placement of the remote file can be displayed if the **fileplace** command is run directly on the file server.
2. The **fileplace** command reads the file's list of blocks directly from the logical volume on disk. If the file is newly created, extended, or truncated, the file system information may not yet be on the disk when the **fileplace** command is run. Use the **sync** command to flush the file information to the logical volume.
3. There is no Indirect/Double Indirect blocks concept in JFS2 filesystem. The file is represented in terms of extents. Therefore the size of the maximum extent depends on the aggregate block size. With a 512 byte aggregate block size (the smallest allowable), the maximum extent is 512*(2^ 24-1) bytes long (slightly under 8G). With a 4096 byte aggregate block size (the largest allowable), the maximum extent is 4096*(2^ 24-1) bytes long (slightly under 64G).

   These limits apply only to a single extent; in no way do they have any limiting effects on overall file sizes.

Flags

Item	Description
-i	Displays the indirect blocks for the file, if any. The indirect blocks are displayed in terms of either their logical or physical volume block addresses, depending on whether the **-l** or **-p** flag is specified.
-l	Displays file placement in terms of logical volume fragments, for the logical volume containing the file. The **-l** and **-p** flags are mutually exclusive. **Note:** If neither the **-l** flag nor the**-p** flag is specified, the **-l** flag is implied by default. If both flags are specified, the **-p** flag is used.
-m *LogicalVolumeName*	Displays the logical to physical map for a logical volume.
-n *FragNumber*	Displays the logical or physical file blocks ranging from the first block to the block corresponding to *FragNumber*.
-o *FragOffset*	Displays the logical or physical file blocks ranging from the block corresponding to *fragoffset* + 1 to the last block. The **fileplace** command displays the address of the specific fragment when both the **-n** flag and the **-o** flag is specified.
-p	Displays file placement in terms of underlying physical volume, for the physical volumes that contain the file. If the logical volume containing the file is mirrored, the physical placement is displayed for each mirror copy. The **-l** and **-p** flags are mutually exclusive.

Item	Description
-v	Displays more information about the file and its placement, including statistics on how widely the file is spread across the volume and the degree of fragmentation in the volume. The statistics are expressed in terms of either the logical or physical volume fragment numbers, depending on whether the **-l** or **-p** flag is specified. *File space efficiency* is calculated as the number of nonnull fragments (*N*) divided by the range of fragments (*R*) assigned to the file and multiplied by 100, or (*N* /*R*) x 100. Range is calculated as the highest assigned address minus the lowest assigned address plus 1, or *MaxBlk*-*MinBlk*+1. For example, the logical blocks written for the file are 01550 through 01557, so *N* equals 8. The range, *R*, (01557 - 01550 +1) also equals 8. Space efficiency for this file is 100% or 8/8 x 100. The **-v** flag message prints the results of the (*N*/*R*)+100 equation. According to this method of calculating efficiency, files greater than 32KB are never 100% efficient because of their use of the indirect block. *Sequential efficiency* is defined as 1 minus the number of gaps (*nG*) divided by number of possible gaps (*nPG*) or 1 - (*nG*/*nPG*). The number of possible gaps equals *N* minus 1 (*nPG=N* - 1). If the file is written to 9 blocks (greater than 32KB), and the logical fragment column shows: `01550-01557` `01600` The file is stored in 2 fragments out of a possible 9 fragments. The sequential efficiency calculation for this file is: *nG*=1 *nPG*=9-1=8 (1-1/8) x 100=87.5%
-a	Marks Allocated But Not Recorded (ABNR) blocks with an asterisk (*) at the beginning of the line. ABNR is a feature provided by the J2 filesystem.

Examples

1. To display the placement of a file in its logical volume, enter:

   ```
   fileplace data1
   ```

   This example displays the list of fragments and the logical volume that contains the file `data1`.
2. To display the indirect blocks for a file, enter:

   ```
   fileplace -i data1
   ```

   In addition to the default list of logical volume fragments, the indirect blocks (if any) used to store the file block addresses in the file system are enumerated.
3. To display more placement information for a file, enter:

   ```
   fileplace -v data1
   ```

In addition to the default list of logical volume fragments, statistics about the placement efficiency are displayed.

4. To display all information about the placement of a file on its physical volumes, enter:

```
fileplace -piv data1
```

This example displays the list of file and indirect blocks in terms of the underlying physical volumes, and includes statistics about the efficiency of the placement.

5. To display the locations of the underlying physical volume for the first 18 blocks in the **/usr/lib/boot/unix_mp** file, enter:

```
fileplace -n 18 -p /usr/lib/boot/unix_mp
```

6. To display the locations of the underlying physical volume from the 18th block to the last block in the **/usr/lib/boot/unix_mp** file, enter:

```
fileplace  -p -o 17 /usr/lib/boot/unix_mp
```

7. To display the location of the underlying physical volume of the 18th block in the **/usr/lib/boot/unix_mp** file, enter:

```
fileplace -o 17 -n 1 -p /usr/lib/boot/unix_mp
```

Files

Item	Description
/dev/hd0, /dev/hd1, .../dev/hd*n*	Specifies the logical volume.

find Command

Purpose

Finds files with a matching expression.

Syntax

`find` [**-H** | **-L**] *Path ...* [*Expression*]

Description

The **find** command recursively searches the directory tree for each specified *Path* parameter, seeking files that match a Boolean expression. The Boolean expression is written by using the terms that are provided in the following text. When the **find** command is recursively descending directory structures, it does not descend into directories that are symbolically linked into the current hierarchy. The output from the **find** command depends on the terms that are specified by the *Expression* parameter.

The **find** command does not support the 4.3 BSD fast-find syntax.

Flags

Item	Description
-H	Causes the file information and file type that are evaluated for each symbolic link that is encountered on the command line to be those of the file that is referenced by the link, and not the link itself. If the referenced file does not exist, the file information and type are for the link itself. File information for all symbolic links not on the command line is that of the link itself.

Item	Description
-L	Causes the file information and file type that are evaluated for each symbolic link to be those of the file that is referenced by the link, and not the link itself.

Expression Terms

These Boolean expressions and variables describe the search boundaries of the **find** command as defined in the *Path* and *Expression* parameters.

Note: In the following definitions, the *n* variable specifies a decimal integer that can be expressed as +*n* (more than *n*), -*n* (less than *n*), or *n* (exactly *n*) and the *Number* variable specifies a decimal integer that can be expressed as +*Number* (more than *Number*), -*Number* (less than Number), or *Number* (*Number*-1 to *Number*).

Item	Description
\(*Expression* \)	Evaluates to the value True if the expression in parentheses is true.
-amin *n*	The value of *n* can be one of the following values: ***n*** Evaluates as True if the file access time subtracted from the initialization time, divided by 60 seconds (with any remainder discarded), is *n*. ***-n*** Evaluates as True if the file access time subtracted from the initialization time, divided by 60 seconds (with any remainder discarded), is less than *n*. ***+n*** Evaluates as True if the file access time subtracted from the initialization time, divided by 60 seconds (with any remainder discarded), is greater than *n* (in case of UNIX03, greater than *n*+1). For example, `-amin 2` is true if the file is accessed within 1 to 2 minutes. **Note:** Files that are accessed after the **find** command start time are not taken into account. However, when the **find** command is used within the unary NOT operator for non-UNIX03 behavior, the files that are modified after the command start time are displayed until the value of *n*.

Item	Description
-atime *n*	The value of *n* can be one of the following values: ***n*** Evaluates as True if the file access time subtracted from the initialization time, divided by 86400 seconds (with any remainder discarded), is *n*. ***-n*** Evaluates as True if the file access time subtracted from the initialization time, divided by 86400 seconds (with any remainder discarded), is less than *n*. ***+n*** Evaluates as True if the file access time subtracted from the initialization time, divided by 86400 seconds (with any remainder discarded), is greater than *n* (in case of UNIX03, greater than *n*+1). **Note:** The definition of `-atime` is changed to comply with the Single UNIX Specification, Version 3. The previous behavior of `-atime` evaluated as True if the file was accessed in *n*-1 to *n* multiples of 24 hours. By default, `find -atime` works like it did before UNIX03. The UNIX03 behavior can be obtained by setting the environment variables `XPG_SUS_ENV` to `ON` and `XPG_UNIX98` to `OFF`. The previous behavior for this option can be obtained by setting the `XPG_UNIX98` variable to `ON`. Files that are accessed after the **find** command start time is not taken into account. However, when the **find** command is used within the unary `NOT` operator for non-UNIX03 behavior, the files that are modified after the command start time are displayed until the value of *n*.
-cmin *n*	The value of *n* can be one of the following values: ***n*** Evaluates as True if the file i-node modification time subtracted from the initialization time, divided by 60 seconds (with any remainder discarded), is *n*. ***-n*** Evaluates as True if the file i-node modification time subtracted from the initialization time, divided by 60 seconds (with any remainder discarded), is less than *n*. ***+n*** Evaluates as True if the file i-node modification time subtracted from the initialization time, divided by 60 seconds (with any remainder discarded), is greater than *n* (in case of UNIX03, greater than *n*+1). **Note:** Files with i-nodes that are modified after the **find** command start time are not taken into account. However, when the **find** command is used within the unary `NOT` operator for non-UNIX03 behavior, files with i-nodes modified after the command start time are displayed until the value of *n*.
-cpio *Device*	Writes the current file to the specified device in the **cpio** command format.

Item	Description
-ctime *n*	The value of *n* can be one of the following values: ***n*** Evaluates as True if the file i-node modification time subtracted from the initialization time, divided by 86400 seconds (with any remainder discarded), is *n*. ***-n*** Evaluates as True if the file i-node modification time subtracted from the initialization time, divided by 86400 seconds (with any remainder discarded), is less than *n*. ***+n*** Evaluates as True if the file i-node modification time subtracted from the initialization time, divided by 86400 seconds (with any remainder discarded), is greater than *n* (in case of UNIX03, greater than *n*+1). **Note:** The definition of **-ctime** is changed to comply with the Single UNIX Specification, Version 3. The previous behavior of **-ctime** evaluated as True if the file was accessed in *n*-1 to *n* multiples of 24 hours. By default, `find -ctime` works like it did before UNIX03. The UNIX03 behavior can be obtained by setting the environment variables `XPG_SUS_ENV` to `ON` and `XPG_UNIX98` to `OFF`. The previous behavior for this option can be obtained by setting the `XPG_UNIX98` variable to `ON`. Files with i-nodes modified after the **find** command start time is not taken into account. However, when the **find** command is used within the unary `NOT` operator for non-UNIX03 behavior, files with i-nodes modified after the command start time is displayed until the value of *n*.
-depth	Always evaluates to the value True. Causes the descent of the directory hierarchy to be done so that all entries in a directory are affected before the directory itself is affected. It can be useful when the **find** command is used with the **cpio** command to transfer files that are contained in directories without write permission.
-ea	Evaluates to the value True if file has either access control information (ACL) or Extended attributes (EA) set.
-exec *Command*	Evaluates to the value True if the specified command runs and returns a 0 value as exit status. The end of the specified command must be punctuated by a semicolon in quotation marks, an escaped semicolon, or a plus sign. An argument that contains the two characters `{}` (braces) must be followed by a plus sign that punctuates the end of the specified command. A command parameter `{}` (braces) is replaced by the current path name.
-follow	Causes symbolic and hard links to be followed.
-fstype *Type*	Evaluates to the value True if the file system to which the file belongs is of the specified type. The *Type* variable has a value of `jfs` (journaled file system), `nfs` (network file system), `jfs2` (enhanced journaled file system), `procfs` (proc file system), or `namefs` (name file system).
-group *Group*	Evaluates to the value True if the file belongs to the specified group. If the value of the *Group* variable is numeric and does not appear in the `/etc/group` file, it is interpreted as a group ID.

Item	Description
-inum *n*	Evaluates to the value True if file has an i-node matching the value of the *n* variable.
-links *n*	Evaluates to the value True if the file has the specified number of links. See the **ln** command for a description of links.
-iregex *regular_expression*	Evaluates to the value True if the entire path name of the file matches the regular expression. This option is similar to the **-regex** option, except that the match is case-insensitive.
-long	Prints all available characters of each user/group name instead of truncating to the first 8 when used in combination with `-ls`.
-ls	Always evaluates to the value True. Causes the current path name to be printed together with its associated statistics. These statistics include the following values: • I-node number • Size in KB (1024 bytes) • Protection mode • Number of hard links • User • Group • Size in bytes • Modification time If the file is a special file, the size field contains the major and minor device numbers. If the file is a symbolic link, the path name of the linked-to file is printed preceded by the -> (hyphen, greater than) symbols. Formatting is similar to that of the **ls -filds** command, however formatting is done internally without running the **ls** command. Therefore, differences in output with the **ls** command might exist, such as with the protection mode.
-mmin *n*	The value of *n* can be one of the following values: ***n*** Evaluates as True if the file modification time subtracted from the initialization time, divided by 60 seconds (with any remainder discarded), is *n*. ***-n*** Evaluates as True if the file modification time subtracted from the initialization time, divided by 60 seconds (with any remainder discarded), is less than *n*. ***+n*** Evaluates as True if the file modification time subtracted from the initialization time, divided by 60 seconds (with any remainder discarded), is greater than *n* (in case of UNIX03, greater than *n*+1). **Note:** Files that are modified after the **find** command start time are not taken into account. However, when the **find** command is used within the unary NOT operator for non-UNIX03 behavior, the files that are modified after the command start time are displayed until the value of *n*.

Item	Description
-mtime *n*	The value of *n* can be one of the following values: ***n*** Evaluates as True if the file modification time subtracted from the initialization time, divided by 86400 seconds (with any remainder discarded), is *n*. 86400 seconds is 24 hours. ***-n*** Evaluates as True if the file modification time subtracted from the initialization time, divided by 86400 seconds (with any remainder discarded), is less than *n*. ***+n*** Evaluates as True if the file modification time subtracted from the initialization time, divided by 86400 seconds (with any remainder discarded), is greater than *n* (in case of UNIX03, greater than *n*+1). **Note:** The definition of **-mtime** is changed to comply with the Single UNIX Specification, Version 3. The previous behavior of **-mtime** evaluated as True if the file is modified in *n*-1 to *n* multiples of 24 hours. By default, `find -mtime` works like it did before UNIX03. The UNIX03 behavior can be obtained by setting the environment variables `XPG_SUS_ENV` to `ON` and `XPG_UNIX98` to `OFF`. The previous behavior for this option can be obtained by setting the `XPG_UNIX98` variable to `ON`. Files that are modified after the **find** command start time are not taken into account. However, when the **find** command is used within the unary `NOT` operator for non-UNIX03 behavior, the files modified after the command start time are displayed until the value of *n*.
-name *File*	Evaluates to the value True if the value of the *File* variable matches the file name. The typical shell file name generation characters (see the **sh** command) can be used. The pattern must be enclosed in either quotation marks or the escape characters. The escape character is used when the **find** command is used from the shell. A backslash (\) is used as an escape character within the pattern. You can use wildcard (pattern-matching) characters, provided they are in quotation marks. In an expression such as [a-z], the hyphen means through according to the current collating sequence. A collating sequence might define equivalence classes for use in character ranges. For more information about collating sequences and equivalence classes, see "National Language Support Overview" in the *Globalization Guide and Reference*.
-newer *File*	Evaluates to the value True if the current file is modified more recently than the file indicated by the *File* variable.
-nogroup	Evaluates to the value True if the file belongs to a group not in the `/etc/group` database.
-nouser	Evaluates to the value True if the file belongs to a user not in the `/etc/passwd` database.

Item	Description
-ok *Command*	The same as the **-exec** expression, except that the **find** command verifies whether it must start the specified command. An affirmative response starts the command. The end of the specified command must be punctuated by a semicolon that is enclosed in quotation marks or the \; (backslash-escape semicolon).
-perm [**-**] *OctalNumber*	Evaluates to the value True if the permission code of the file exactly matches the *OctalNumber* parameter. For details about file permissions, refer to the **chmod** command. If the optional **-** (hyphen) is present, this expression evaluates to the value true if at least these permissions are set. The *OctalNumber* parameter can be up to 9 octal digits. **Note:** For files that are a part of TCB environment, additional security bits are added to the permission of the files. These files have the S_ITCB bit set and the security bit set is defined as 0x010000000. Therefore, the octal permissions value of a TCB enabled file must include the bit setting of 100000000 along with its other permission bits. Example: To list a file, which is a part of the TCB environment, find -perm 100000600 -print. It lists the names of the files that have only owner-read and owner-write permission and are a part of the TCB environment. See the **chmod** command for an explanation of permission codes.
-perm [**-**] *Mode*	The mode argument is used to represent file mode bits. It is identical in format to the <symbolicmode> operand described in **chmod**, and is interpreted as follows: Initially, a template is assumed with all file mode bits cleared. Op (-) symbols have the following function: **+** Sets the appropriate mode bits in the template **-** Clears the appropriate bits **=** Sets the appropriate mode bits, without regard to the contents of the process' file mode creation mask The op symbol - cannot be the first character of mode. It avoids ambiguity with the optional leading hyphen. Because the initial mode is all bits off, there are no symbolic modes that must use - as the first character. If the hyphen is omitted, the primary evaluates as True when the file permission bits exactly match the value of the resulting template. Otherwise, if mode is prefixed by a hyphen, the primary evaluates as True if at least all bits in the resulting template are set in the file permission bits. The *Mode* parameter is identical to the **chmod** command syntax. This expression evaluates to the value True if the file has exactly these permissions. If the optional **-** (hyphen) is present, this expression evaluates to the value True if at least these permissions are set.

Item	Description
-print	Always evaluates to the value True. Displays the current path name. The **find** command assumes a **-print** expression, unless the **-exec, - ls**, or **-ok** expressions are present.
-prune	Always evaluates to the value True. Stops the descent of the current path name if it is a directory. If the **-depth** flag is specified, the **-prune** flag is ignored.
-size *n*	Evaluates to the value True if the file is the specified *n* of blocks long (512 bytes per block). The file size is rounded up to the nearest block for comparison.
-regex *regular_expression*	Evaluates to the value True if the entire path name of the file matches the regular expression. This option does not search for the regular expression but matches the regular expression with the complete path name of the file. For example, to match a file named `./test`, you can use the regular expression `.*test.*` or `.*t.*t`, but not `t.*t`.
-regextype *Type*	Always evaluates to the value True. This option specifies the type of regular expression syntax for the **-regex** and **-iregex** options. It also affects regular expressions that occur later in the command line. The *Type* variable can have one of the following values: **Basic** For basic regular expression syntax. **Extended** For extended regular expression syntax. **Note:** If the **-regextype** option is not used, the regular expressions are interpreted as `basic`.
-size *n***c**	Evaluates to the value True if the file is exactly the specified *n* of bytes long. Adding **c** to the end of the *n* variable indicates that the size of the file is measured in individual bytes not blocks.
-type *Type*	Evaluates to the value True if the *Type* variable specifies one of the following values: **b** Block special file **c** Character special file **d** Directory **f** Plain file **l** Symbolic link **p** FIFO (a named pipe) **s** Socket

Item	Description
-user *User*	Evaluates to the value True if the file belongs to the specified user. If the value of the *User* variable is numeric and does not appear as a login name in the `/etc/passwd` file, it is interpreted as a user ID.
-xdev	Always evaluates to the value True. Prevents the **find** command from traversing a file system different from the one specified by the *Path* parameter.

These expressions can be combined by using the following operators in the order of decreasing precedence:

1. **(** *Expression* **)** - A parenthetic group of expressions and operators (parentheses are special to the shell and require the backslash-escape sequence).
2. **!** *Expression* - The negation of an expression (**'!'** is the unary NOT operator).
3. *Expression* [**-a**] *Expression* - Concatenation of expressions (the AND operation is implied by the juxtaposition of two primaries or might be explicitly stated as **-a**).
4. *Expression* **-o** *Expression* - Alternation of primaries; **-o** is the OR operator. The second expression is not evaluated if the first expression is true.

Note: When you use the **find** and **cpio** commands together, you must use the `-follow` option and the `-L` option with the **cpio** command. Not using these two options together produces undesirable results. If expression is not present, `-print` as used in the default expression. For example, if the specified expression does not contain any of the primaries `-exec`, `-ok`, or `-print`, the expression is replaced by (*given_expression*) `-print`. The `-user`, `-group`, and `-newer` primaries each evaluate their respective arguments only once. Using a command that is specified by `-exec` or `-ok` does not affect subsequent primaries on the same file.

Exit Status

This command returns the following exit values:

Item	Description
0	All *Path* parameters were traversed successfully.
>0	An error occurred.

Examples

1. To list all files in the file system with a specified base file name, type:

   ```
   find /  -name .profile -print
   ```

   This command searches the entire file system and writes the complete path names of all files named `.profile`. The / (slash) instructs the **find** command to search the root directory and all of its subdirectories. In order not to waste time, it is best to limit the search by specifying the directories where you think the files might be.

2. To list files that have a specific permission code in the current directory tree, type:

   ```
   find . -perm 0600  -print
   ```

   This command lists the names of the files that have only owner-read and owner-write permission. The . (dot) instructs the **find** command to search the current directory and its subdirectories. See the **chmod** command for an explanation of permission codes.

3. To search several directories for files with certain permission codes, type:

```
find manual clients proposals -perm -0600  -print
```

This command lists the names of the files that have owner-read and owner-write permissions and possibly other permissions. The `manual`, `clients`, and `proposals` directories and their subdirectories are searched. In the previous example, `-perm 0600` selects only files with permission codes that match `0600` exactly. In this example, `-perm -0600` selects files with permission codes that allow the accesses that are indicated by `0600` and other accesses above the `0600` level. It also matches the permission codes 0622 and 0744.

4. To list all files in the current directory that are changed during the current 24-hour period, type:

```
find .  -ctime 1  -print
```

5. To search for regular files with multiple links, type:

```
find .  -type f  -links +1  -print
```

This command lists the names of the ordinary files (`-type f`) that have more than one link (`-links +1`).

Note: Every directory has at least two links: the entry in its parent directory and its own `.` (dot) entry. The **ln** command explains multiple file links.

6. To find all accessible files whose path name contains **find**, type:

```
find .  -name '*find*' -print
```

7. To remove all files named `a.out` or `*.o` that are not accessed for a week and that are not mounted by using `nfs`, type:

```
find / \( -name a.out -o -name '*.o' \) -atime +7 ! -fstype nfs -exec rm {} \;
```

Note: The number that is used within the **-atime** expression is +7. It is the correct entry if you want the command to act on files that are not accessed for more than a week (seven 24-hour periods).

8. To print the path names of all files in or below the current directory, except the directories named SCCS or files in the SCCS directories, type:

```
find .  -name SCCS  -prune -o -print
```

To print the path names of all files in or below the current directory, including the names of SCCS directories, type:

```
find . -print -name SCCS -prune
```

9. To search for all files that are exactly 414 bytes long, type:

```
find . -size 414c -print
```

10. To find and remove every file in your home directory with the `.c` suffix, type:

```
find /u/arnold -name "*.c" -exec rm {} \;
```

Every time the **find** command identifies a file with the `.c` suffix, the **rm** command deletes that file. The **rm** command is the only parameter that is specified for the **-exec** expression. The {} (braces) represent the current path name.

11. In this example, `dirlink` is a symbolic link to the directory `dir`. To list the files in `dir` by referring to the symbolic link `dirlink` on the command line, type:

```
find -H dirlink -print
```

12. In this example, `dirlink` is a symbolic link to the directory `dir`. To list the files in `dirlink`, traversing the file hierarchy under `dir` including any symbolic links, type:

```
find -L dirlink -print
```

13. To determine whether the file `dir1` referred by the symbolic link `dirlink` is newer than `dir2`, type:

```
find -H dirlink -newer dir2
```

Note: Because the **-H** flag is used, time data is collected not from `dirlink` but instead from `dir1`, which is found by traversing the symbolic link.

14. To produce a listing of files in the current directory in `ls` format with expanded user and group name, type:

```
find . -ls -long
```

15. To list the files with ACL/EA set in current directory, type:

```
find . -ea
```

16. To list the files that are modified within 60 minutes, type:

```
find . -mmin -60
```

17. To find all path names in the `/home` directory that contain a pattern `afile` in the path name, type the following command:

```
find /home -regextype basic -regex ".*afile.*"
```

18. To find all path names in the `/home` directory that contain a pattern `afile` or `cap` in the path name, type the following command:

```
find /home -regextype extended -regex ".*afile.*|.*cap.*"
```

19. To find all path names in the `/home` directory that contain a pattern `afile`, `AFILE`, `cap`, or `CAp` in the path name, type the following command:

```
find /home -regextype extended -iregex ".*afile.*|.*cap.*"
```

Files

Item	Description
`/usr/bin/find`	Contains the **find** command.
`/bin/find`	Symbolic link to the **find** command.
`/etc/group`	Contains a list of all known groups.
`/etc/passwd`	Contains a list of all known users.

finger Command

Purpose

Shows user information. This command is the same as the **f** command.

Syntax

{ **finger** | **f** }[[**-b**][**-h**] [**-l**][**-p**]]|[**-i**][**-q**][**-s**][**-w**]]

[**-f**][**-m**][*User*| *User* **@***Host*| **@***Host*]

Description

The **/usr/bin/finger** command displays information about the users currently logged in to a host. The format of the output varies with the options for the information presented.

Default Format

The default format includes the following items:

- Login name
- Full user name
- Terminal name
- Write status (an * (asterisk) before the terminal name indicates that write permission is denied)

For each user on the host, the default information list also includes, if known, the following items:

- Idle time (Idle time is minutes if it is a single integer, hours and minutes if a : (colon) is present, or days and hours if a "d" is present.)
- Login time
- Site-specific information

The site-specific information is retrieved from the gecos field in the **/etc/passwd** file. The gecos field may contain the Full user name followed by a comma or / (slash character). All information that follows the comma or slash character is displayed by the finger command with the Site-specific information.

Longer Format

A longer format is used by the **finger** command whenever a list of user's names is given. (Account names as well as first and last names of users are accepted.) This format is multiline, and includes all the information described above along with the following:

- User's **$HOME** directory
- User's login shell
- Contents of the **.plan** file in the user's **$HOME** directory
- Contents of the **.project** file in the user's **$HOME** directory

The **finger** command may also be used to look up users on a remote system. The format is to specify the user as *User@Host*. If you omit the user name, the **finger** command provides the standard format listing on the remote system.

Create the **.plan** and **.project** files using your favorite text editor and place the files in your **$HOME** directory. The **finger** command uses the **toascii** subroutine to convert characters outside the normal ASCII character range when displaying the contents of the **.plan** and **.project** files. The **finger** command displays a `M-` before each converted character.

When you specify users with the *User* parameter, you can specify either the user's first name, last name, or account name. When you specify users, the **finger** command, at the specified host, returns information about those users only in long format.

For other information about the **finger** command, see "Installation of TCP/IP" in *Networks and communication management*.

Flags

Item	Description
-b	Gives a brief, long-form listing.
-f	Suppresses printing of header line on output (the first line that defines the fields that are being displayed).
-h	Suppresses printing of **.project** files on long and brief long formats.

Item	Description
-i	Gives a quick listing with idle times.
-l	Gives a long-form listing.
-m	Assumes that the *User* parameter specifies a user ID (used for discretionary access control), *not* a user login name.
-p	Suppresses printing of **.plan** files on long-form and brief long-form formats.
-q	Gives a quick listing.
-s	Gives a short format list.
-w	Gives a narrow, short-format list.

Parameters

Item	Description
@Host	Specifies all logged-in users on the remote host.
User	Specifies a local user ID (used for discretionary access control) or local user login name, as specified in the **/etc/passwd** file.
User@Host	Specifies a user ID on the remote host, displayed in long format.

Examples

1. To get information about all users logged in to host `alcatraz`, enter:

   ```
   finger @alcatraz
   ```

   Information similar to the following is displayed:

   ```
   [alcatraz.austin.ibm.com]
   Login     Name         TTY Idle          When      Site Info
   brown    Bob Brown   console   2d   Mar 15 13:19
   smith    Susan Smith    pts0  11:   Mar 15 13:01
   jones    Joe Jones      tty0    3   Mar 15 13:01
   ```

   User `brown` is logged in at the `console`, user `smith` is logged in from pseudo teletype line `pts0`, and user `jones` is logged in from `tty0`.

2. To get information about user `brown` at `alcatraz`, enter:

   ```
   finger brown@alcatraz
   ```

   Information similar to the following is displayed:

   ```
   Login name: brown
   Directory: /home/brown    Shell: /home/bin/xinit -L -n Startup
   On since May 8 07:13:49 on console
   No Plan.
   ```

3. To get information about user `brown` at a local host in short form, enter:

   ```
   finger -q brown
   ```

   Information similar to the following is displayed:

   ```
   Login           TTY             When
   brown           pts/6           Mon Dec1710:58
   ```

Files

Item	Description
/usr/bin/finger	Contains the **finger** command.
/etc/utmp	Contains list of users currently logged in.
/etc/passwd	Defines user accounts, names, and home directories.
/etc/security/passwd	Defines user passwords.
/var/adm/lastlog	Contains last login times.
$HOME/.plan	Optional file that contains a one-line description of a user's plan.
$HOME/.project	Optional file that contains a user's project assignment.

fingerd Daemon

Purpose

Provides server function for the **finger** command.

Syntax

Note: The **fingerd** daemon is usually started by the **inetd** daemon. It can also be controlled from the command line, using System Resource Controller (SRC) commands.

/usr/sbin/fingerd [**-s**] [**-f**]

Description

The **/usr/sbin/fingerd** daemon is a simple protocol that provides an interface to the **finger** command at several network sites. The **finger** command returns a status report on either the current system or a user. The **fingerd** daemon listens for Transmission Control Protocol (TCP) requests at port 79 as listed in the **/etc/services** file and the /**etc**/**inetd.conf** file.

For individual site security concern the **fingerd** daemon, by default, will not forward any **finger** request to any other system. If it receives a **finger** forward request, the **fingerd** daemon replies with the message `Finger forwarding service denied` to the **finger** command. The system administractor has the option to turn on finger forwarding as the default when running the **fingerd** daemon by using the **-f** flag.

Changes to the **fingerd** daemon can be made using the System Management Interface Tool (SMIT) or SRC or by editing the **/etc/inetd.conf** file or /**etc**/**services** file. Entering `fingerd` at the command line is not recommended. The **fingerd** daemon is started by default when it is uncommented in the **/etc/inetd.conf** file.

The **inetd** daemon get its information from the /**etc**/**inetd.conf** file and the /**etc**/**services** file.

After changing the /**etc**/**inetd.conf** or /**etc**/**services** file, run the **refresh -s inetd** or **kill-1***InetdPID* command to inform the **inetd** daemon of the changes to its configuration file.

The **fingerd** daemon should have a user ID with the least privileges possible. The **nobody** ID allows the least permissions. Giving the **fingerd** daemon the **nobody** user ID allows the daemon to be used on your host. Change the /**etc**/**services** file to the reflect the user ID you want to use.

Manipulating the fingerd Daemon with the System Resource Controller

The **fingerd** daemon is a subserver of the **inetd** daemon. The **fingerd** daemon is a member of the **tcpip** SRC subsystem group. This daemon is enabled when it is uncommented in the **/etc/inetd.conf** file and can be manipulated by the following SRC commands:

Item	Description
startsrc	Starts a subsystem, group of subsystems, or a subserver.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
lssrc	Gets the status or a subsystem, group or subsystems, or a subserver.

Flags

Item	Description
-s	Turns on socket-level debugging.

Item	Description
-f	Turns on finger forwarding service for this **fingerd** daemon.

Examples

Note: The arguments for the **fingerd** daemon can be specified by using SMIT or by editing the **/etc/inetd.conf** file.

1. To start the **fingerd** daemon type:

   ```
   startsrc -t finger
   ```

   This command starts the **fingerd** subserver.
2. To stop the **fingerd** daemon usually, type:

   ```
   stopsrc -t finger
   ```

   This command allows all pending connections to start and existing connections to complete but prevents new connections from starting.
3. To force stop the **fingerd** daemon and all **fingerd** connections type:

   ```
   stopsrc -f -t finger
   ```

   This command terminates all pending connections and existing connections immediately.
4. To display a short status report about the **fingerd** daemon type:

   ```
   lssrc -t finger
   ```

   This command returns the daemon's name, process ID, and state (active or inactive).

fish Command

Purpose

Plays the go fish card game.

Syntax

fish

Description

The object of the go fish game is to accumulate books of four cards with the same face value. You and the program (your opponent) take turns asking for cards from one another's hand. If your opponent has one or more cards of the value requested, your opponent must hand them over. If not, your opponent prompts `GO FISH!`, and you draw a card from the pool of undealt cards. If you draw the card you asked for, you draw again. As books are made, they are laid down on the table. Play continues until there are no cards left. The player with the most books wins the game. The **fish** command tells you the winner and exits.

The **fish** command prompts with `instructions?` before play begins. To see the instructions, enter Y (yes).

Entering a p as your first move gives you the professional-level game. The default is an amateur-level game.

When playing go fish, you enter the card you want when your opponent prompts:

```
you ask me for:
```

If you press only the Enter key when prompted, you receive information about the number of cards in your opponent's hand and in the pool.

The game displays:

- your current hand, including the books you have accumulated
- `GO FISH!` when either you or your opponent ask for a card the other does not have
- the card drawn after the `GO FISH!` prompt
- the card your opponent asks you for
- completed books (yours or your opponent's)
- the requested card when you or your opponent get another guess.

Examples

The following is a sample of a **fish** screen display:

```
your hand is: A 5 5 7 10 J Q
you ask me for: 5
I say "GO FISH!"
You draw A
I ask you for: 5
Made a book of 5's
I get another guess
I ask you for 6
You say "GO FISH!"
your hand is: A A 7 10 J Q
you ask me for:
```

To exit the game before play is completed, press the Interrupt (Ctrl-C) key sequence.

Files

Item	Description
/usr/games	Location of the system's games.

flcopy Command

Purpose

Copies to and from diskettes.

Syntax

flcopy [**-f** *Device*] [**-h** | **-r**] [**-t** *Number*]

Description

The **flcopy** command copies a diskette (opened as **/dev/rfd0**) to a file named **floppy** created in the current directory, then prints the message: `Change floppy, hit return when done`. The **flcopy** command then copies the **floppy** file to the diskette. You can specify the **-f**, **-h**, **-r**, or **-t***Number* flag to modify the behavior of the **flcopy** command.

Note: You cannot use the **flcopy** command to copy data from one diskette to another diskette of different size.

Flags

Item	Description
-f *Device*	Allows you to specify a drive other than **/dev/rfd0**.
-h	Causes the **flcopy** command to open the **floppy** file in the current directory and copy it to **/dev/rfd0**.
-r	Tells the **flcopy** command to exit after copying the diskette to the **floppy** file in the current directory.
-t *Number*	Causes only the specified *Number* of tracks to be copied. The tracks copied always begin with the first tracks on the diskette.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To copy **/dev/rfd1** to the **floppy** file in the current directory, enter:

   ```
   flcopy -f/dev/rfd1 -r
   ```

2. To copy the first 100 tracks of the diskette, enter:

   ```
   flcopy -f/dev/rfd1 -t100
   ```

Files

Item	Description
/usr/sbin/flcopy	Contains the **flcopy** command.

flush-secldapclntd Command

Purpose

The **flush-secldapclntd** command flushes the cache for the **secldapclntd** daemon process.

Syntax

/**/usr/sbin/flush-secldapclntd**

Description

The **flush-secldapclntd** command clears the cache for the **secldapclntd** daemon process.

Example

1. To flush the **secldapclntd** daemon cache, type:

```
/usr/sbin/flush-secldapclntd
```

Files

Item	Description
/etc/security/ldap/ldap.cfg	Contains information needed by the **secldapclntd** daemon to connect to the server.

fmt Command

Purpose

Formats mail messages prior to sending.

Syntax

/usr/bin/fmt [**-***Width*] [*File* ...]

Description

The **fmt** command starts a text formatter that reads the concatenation of input *Files* (or standard input if no *Files* are specified), then produces on standard output a version of the input with the line lengths set to the value of **-***Width*. If no value is specified with the **-***Width* flag, the default value of 72 characters is used. The spacing at the beginning of the input lines is preserved in the output, as are blank lines and spacing between words.

The **fmt** command is generally used to format mail messages to improve their appearance before they are sent. However, the **fmt** command may also be useful for simple formatting tasks. For example, within visual mode of a text editing program such as the vi editor, the command **!}fmt** formats a paragraph so that all lines are set to the value specified with the **-***Width* flag. If no value is specified with the -*Width* flag, the default value of 72 characters is used. Standard text editing programs are more appropriate than **fmt** for complex formatting operations.

Note: Do not use the **fmt** command if the message contains embedded messages or preformatted information from other files. This command formats the heading information in embedded messages and may change the format of preformatted information.

Flags

Item	Description
File	Specifies the name of the file to be formatted.
-*Width*	Specifies the line length. The default value for *Width* is 72 characters.

Examples

1. To format a message you have created with the mail editor, enter:

   ```
   ~| fmt
   ```

   The ~| is entered at the left margin of the message. After you issue the ~| `fmt` command, the message is formatted. The word `(continue)` is displayed to indicate that you can enter more information or send the message.
2. To format a file and display the output on your screen, enter:

   ```
   fmt file1
   ```

   In this example, the file `file1` is formatted and displayed on your screen.

Files

Item	Description
/usr/bin/fmt	Contains the **fmt** command.

fold Command

Purpose

Folds long lines for fixed-width output devices.

Syntax

fold [**-b**] [**-s**] [**-w** *Width*] [*File*...]

Description

The **fold** command is a filter that folds long lines for a finite-width output device. By default, the command folds the contents of standard input, breaking the lines to a line width of 80 (eighty). You can also specify one or more files as input to the command. The standard input is used if you do not specify any file parameters or if you specify the '-' parameter.

The **fold** command inserts a new-line character in the input lines so that each output line is as wide as possible without exceeding the value specified by the *Width* parameter. If the **-b** flag is specified, line width is counted in bytes. If the **-b** flag is not specified:

- *Width* is counted in columns as determined by the **LC_CTYPE** environment variable.
- A backspace character decreases the length of an output line by 1.
- A tab character advances to the next column where the column position is 1 plus a multiple of 8.

The **fold** command accepts **-w** *Width* values in multiples of 8 if the file contains tabs. To use other width values when the file contains tabs, use the **expand** command before using the **fold** command.

Notes:

1. The **fold** command may affect any underlining that is present.
2. The **fold** command does not insert new-line characters in the middle of multibyte characters even when the **-b** flag is used.

Flags

Item	Description
-b	Counts *Width* in bytes. The default is to count in columns.
-s	Breaks the line after the rightmost blank within the *Width* limit, if an output line segment contains any blank characters. The default is to break lines so each output line segment is as wide as possible.
-w *Width*	Specifies the maximum line width as the value of the *Width* variable. The default is 80.

Parameters

The **fold** command supports the following parameters:

Item	Description
File	The path name of a text file that must be folded. If you do not specify any file parameters, the standard input is used.
INPUT FILES	If you specify the **-b** flag, the input files are text files and the width of the lines of the text files can exceed {LINE_MAX} bytes in length. **Note:** If the **-b** flag is not specified, the input files are text files and the width of the lines of the text files can not exceed {LINE_MAX} bytes in length.

Exit Status

This command returns the following exit values:

Item	Description
0	All input files processed successfully.
>0	An error occurred.

Examples

To fold the lines of a file named `longlines` into width 72 (seventy-two), enter:

```
fold -w 72 longlines
```

Files

Item	Description
/usr/bin/fold	Contains the **fold** command.

folder Command

Purpose

Selects and lists folders and messages.

Syntax

folder [**+** *Folder*] [*Message*] [**-all**] [**-nopack** | **-pack**] [**-nofast** | **-fast**] [**-norecurse** | **-recurse**] [**-print** | **-noprint**] [**-header** | **-noheader**] [**-nototal** | **-total**] [**-push** | **-pop**] [**-list** | **-nolist**]

Description

The **folder** command sets the current folder and the current message for that folder, and lists information about your folders. By default, the **folder** command lists the current folder name, the number of messages, the range of the message numbers, and the current message.

The folder specified by the **+***Folder* flag becomes the current folder. The message specified by the *Message* parameter becomes the current message for the folder. Use the **-pack** flag to renumber the messages in a folder.

Flags

Item	Description
-all	Displays a line of information about each folder in your mail directory.
-fast	Displays only the names of the folders.
+*Folder*	Specifies the folder information to display.
-header	Displays column headings for the folder information.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.
-list	Displays the current folder followed by the contents of the folder stack.
Message	Sets the specified message as the current message. Unless you specify the **+***Folder* flag, the command sets the specified message for the current folder. Use the following references to specify a message: ***Number*** Number of the message. **cur or . (period)** Current message. This is the default. **first** First message in a folder. **last** Last message in a folder. **next** Message following the current message. **new** The new message that is created. **prev** Message preceding the current message.
-nofast	Displays information about each folder. This flag is the default.
-noheader	Suppresses column headings for the folder information. This flag is the default.
-nolist	Suppresses the display of the folder-stack contents. This flag is the default.
-nopack	Prevents renumbering of the messages in the folder. This flag is the default.
-noprint	Prevents display of folder information. If the **-push**, **-pop**, or **-list** flag is specified, the **-noprint** flag is the default.
-norecurse	Displays information about the top-level folders in your current folder only. Information about subfolders is not displayed. This flag is the default.

Item	Description
-nototal	Prevents display of the total of all messages and folders in your mail directory structure. When the **-all** flag is specified, the default is the **-total** flag; otherwise, the **-nototal** flag is the default.
-pack	Renumbers the messages in the specified folder. Renumbering eliminates gaps in the message numbering after messages have been deleted.
-pop	Removes the folder from the top of the folder stack and makes it the current folder. The **+***Folder* flag cannot be specified with the **-pop** flag.
-print	Displays information about the folders. If the **-push**, **-pop**, or **-list** flag is specified, the **-noprint** flag is the default; otherwise, the **-print** flag is the default.
-push	Moves the current folder to the top of the folder stack and sets the specified folder as the current folder. If no folder is specified, the **-push** flag swaps the current folder for the folder on top of the folder stack.
-recurse	Displays information about all folders and subfolders in your current folder.
-total	Displays all messages and folders in your mail directory structure. The **-total** flag does not display information for subfolders unless you specify the **-recurse** flag. The **-total** flag is the default if the **-all** flag is specified.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Current-Folder:`	Sets the default current folder.
`Folder-Protect:`	Sets the protection level for the new folder directories.
`Folder-Stack:`	Specifies the folder stack.
`lsproc:`	Specifies the program used to list the contents of a folder.
`Path:`	Specifies the user's MH directory.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display information about the current folder, enter:

   ```
   folder
   ```

   The system responds with a message similar to the following:

   ```
   inbox+ has 80 messages (1-82); cur = 7; (others).
   ```

   In this example, the current folder is `inbox`.The folder contains `80 messages`, ranging from message 1 to message 82. The current message number is 7.

2. To display information about all folders, enter:

   ```
   folder -all
   ```

The system responds with a message similar to the following:

```
Folder    #  of  messages  (range);  cur  msg    (other  files)
inbox+  has  80  messages  (1-82);  cur=  7;  (others).
test    has    5  messages  (1-5);    cur=  5;  (others).

        Total=  85  messages  in  2  folders
```

In this example, there are 2 folders containing a total of 85 messages. The current folder is `inbox`, indicated by the + (plus sign) that follows it.

3. To make the `test` folder the current folder and display information about `test`, enter:

   ```
   folder   +test
   ```

   The system responds with a message similar to the following:

   ```
   test+  has  5  messages  (1-5);  cur  =  5;  (others)
   ```

4. To make message 2 the current message in the current folder, enter:

   ```
   folder  2
   ```

   The system responds with a message similar to the following:

   ```
   test+  has  5  messages  (1-5);  cur  =  2;  (others)
   ```

5. To create a folder called `group` and make it the current folder, enter:

   ```
   folder   +group
   ```

   The system responds with a message similar to the following:

   ```
   Create  folder  "/home/dawn/Mail/group"?  _
   ```

   Enter:

   ```
   yes
   ```

   The system responds with a message similar to the following:

   ```
   group+  has  no  messages.
   ```

6. To renumber the messages in the current folder, enter:

   ```
   folder   -pack
   ```

   The system responds with a message similar to the following:

   ```
   inbox+  has  80  messages  (1-80);  cur=  7;  (others).
   ```

   In this example, the messages are renumbered to eliminate gaps in the message numbering after messages have been deleted.

Files

Item	Description
$HOME/.mh_profile	Contains the MH user profile.
/usr/bin/folder	Contains the **folder** command.

folders Command

Purpose

Lists all folders and messages in mail directory.

Syntax

folders [**+***Folder*] [*Message*] [**-all**] [**-pack** | **-nopack**] [**-fast** | **-nofast**] [**-recurse** | **-norecurse**] [**-print** | **-noprint**] [**-header** | **-noheader**] [**-total** | **-nototal**] [**-push** | **-pop**] [**-list** | **-nolist**]

Description

The **folders** command lists all folders and messages in your mail directory. This command is equivalent to the **folder** command specified with the **-all** flag.

Flags

Item	Description
-all	Displays a line of information about each folder in your mail directory.
-fast	Displays only the names of the folders.
+*Folder*	Specifies the folder information to display.
-header	Displays column headings for the folder information. This flag is the default.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.
-list	Displays the current folder followed by the contents of the folder stack.
Message	Sets the specified message as the current message. Unless you specify the **+***Folder* flag, the command sets the specified message for the current folder. Use the following references to specify a message: ***Number*** Number of the message. **cur or . (period)** Current message. This is the default. **first** First message in a folder. **last** Last message in a folder. **next** Message following the current message. **new** The new message that is created. **prev** Message preceding the current message.
-nofast	Displays information about each folder. This flag is the default.
-noheader	Suppresses column headings for the folder information.
-nolist	Suppresses the display of the folder-stack contents. This flag is the default.
-nopack	Prevents renumbering of the messages in the folder. This flag is the default.

Item	Description
-noprint	Prevents display of folder information. If the **-push**, **-pop**, or **-list** flag is specified, the **-noprint** flag is the default.
-norecurse	Displays information about the folders in your mail directory. Information about subfolders is not displayed. This flag is the default.
-nototal	Prevents display all messages and folders in your mail directory structure.
-pack	Renumbers the messages in the folders. Renumbering eliminates gaps in message numbering after messages have been deleted.
-pop	Removes the folder from the top of the folder stack and makes it the current folder.
-print	Displays the number of messages in each folder, the current message for each folder, and the current folder. If the **-push**, **-pop**, or **-list** flag is specified, the **-noprint** flag is the default; otherwise, the **-print** flag is the default.
-push	Moves the current folder to the top of the folder stack and sets the specified folder as the current folder. If no folder is specified, the **-push** flag swaps the current folder for the folder on top of the folder stack.
-recurse	Displays information about all folders and subfolders in your mail directory structure.
-total	Displays all messages and folders in your mail directory structure. The **-total** flag does not display information for subfolders unless you specify the **-recurse** flag. The **-total** flag is the default.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Current-Folder:`	Sets the default current folder.
`Folder-Protect:`	Sets the protection level for the new folder directories.
`Folder-Stack:`	Specifies the folder stack.
`lsproc:`	Specifies the program used to list the contents of a folder.
`Path:`	Specifies the user's MH directory.

Examples

1. To display information about all folders, enter:

   ```
   folders
   ```

   The system responds with a message similar to the following:

   ```
   Folder  # of messages (range); cur msg  (other files)
   inbox+ has 80 messages (1-82); cur= 7; (others).
   test has 5 messages (1-6);   cur=  5; (others).

      Total= 85 messages in 2 folders.
   ```

   In this example, there are 2 folders containing a total of 85 messages. The current folder is `inbox`, indicated by the + (plus sign) following it.

2. To list only the names of all folders, enter:

   ```
   folders    -fast
   ```

   The system responds with a message similar to the following:

```
inbox
test
```

3. To renumber the messages in all folders, enter:

   ```
   folders    -pack
   ```

   The system responds with a message similar to the following:

   ```
   inbox+ has 80 messages (1-80); cur= 7; (others).
   test has 5 messages (1-5);  cur= 5; (others).
   ```

   In this example, the messages in the `inbox` folder and in the `test` folder have been renumbered to eliminate gaps in message numbering after messages were deleted.

Files

Item	Description
$HOME/.mh_profile	Contains the MH user profile.
/usr/bin/folders	Contains the **folders** command.

forcerpoffline Command

Purpose

Forces a peer domain to be offline.

Syntax

`forcerpoffline` [`-h`] *domain_name*

Description

Attention: Use this command with extreme caution.

The `forcerpoffline` command must be used only if a node is in a pending online state and you are unable to bring it online using the `startrpdomain` command. This scenario can occur if you try to bring the node online while the domain is operating under quorum. If you are not sure why the node is stuck in the pending online state, run the `ctsnap` command before using the `forcerpoffline` command. As a result of running the `forcerpoffline` command, the configuration resource manager subsystem (`IBM.ConfigRM`) and the RMC subsystem (`ctrmc`) are recycled.

Parameters

domain_name
Specifies the name of a previously defined peer domain that is to be forced offline.

Flags

-h
Writes the command usage statement to standard output.

Files

The `/var/ct/cfg/current_cluster` file and the `/var/ct/cfg/default_cluster` file are modified.

Standard output

When the `-h` flag is specified, this command usage statement is written to standard output.

Exit status

0
The command ran successfully.

1
The command terminated due to an underlying RMC error.

2
The command terminated due to an underlying error in the command script.

3
The command terminated because the user specified a non-valid flag.

4
The command terminated because the user specified a non-valid parameter.

5
The command terminated due to a user error (specifying a domain name that does not exist, for example).

Security

You must have `root` authority to run this command.

Implementation specifics

This command is part of the `rsct.basic.rte` fileset for AIX and `rsct.basic-3.1.0.0-0.`*platform*`.rpm` package for Linux, Solaris, and Windows, where *platform* is `i386`, `ppc`, `ppc64`, `s390`, or `x86_64`.

Location

/opt/rsct/bin/forcerpoffline

format Command

Purpose

Formats either diskettes or read/write optical media disks.

Syntax

format [**-d** *Device*] [**-f**] [**-l**]

Description

Attention: Formatting a diskette or read/write optical disk destroys any existing data on it.

The **format** command formats diskettes in the diskette drive specified by the *Device* parameter. The **format** command determines the device type, which may be one of the following:

- 5.25-inch low-density diskette (360KB) containing 40x2 tracks, each with 9 sectors
- 5.25-inch high-capacity diskette (1.2MB) containing 80x2 tracks, each with 15 sectors
- 3.5-inch low-density diskette (720KB) containing 80x2 tracks, each with 9 sectors
- 3.5-inch high-capacity diskette (1.44MB) containing 80x2 tracks, each with 18 sectors

- 3.5-inch high-capacity diskette (2.88MB) containing 80x2 tracks, each with 36 sectors

The sector size is 512 bytes for all diskette types.

The **format** command formats a diskette with the highest capacity supported by the diskette drive, unless the *Device* parameter specifies a different density.

The **format** command formats a read/write optical disk, provided that the drive supports setting the Format Options Valid (FOV) bit of the defect list header to 0. To format a read/write optical disk, use the name of the read/write optical drive (such as **/dev/romd0**) after the **-d** flag.

Before formatting a diskette or read/write optical disk, the **format** command prompts for verification. This allows you to end the operation cleanly.

Flags

Item	Description
-d *Device*	Specifies the device used to format the diskette. If the device name ends with the letter **h**, the drive formats the diskette for high density. If the device name ends with the letter **l**, the drive formats the diskette for low density. Refer to the **fd** special file for information about valid device types. This flag is used only with the **format** command. **Attention:** If the diskette drive supports a higher capacity than the highest capacity for which the diskette was manufactured, the capacity of the diskette should be explicitly stated in the *Device* parameter (**-d** *Device* flag) of the **format** command. For example, to format a 1MB diskette on a 4MB diskette drive, specify the diskette capacity in the **-d** flag as follows: `-d /dev/fd0.9 for a 1MB diskette` Failure to do this may cause read and write errors.
-f	Formats the diskette without checking for bad tracks, thus formatting the diskette more quickly. This flag applies to diskettes only, not to read/write optical disks. It is used only with the **format** command.
-l	(Lowercase L) Formats a 360KB diskette in a 5.25-inch, 1.2MB diskette drive. Formats a 720KB diskette in a 3.5-inch 1.4MB diskette drive. This flag applies to diskettes only, not to read/write optical disks. It is used only with the **format** command. **Attention:** A 360KB diskette drive may not be able to read a 360KB diskette that has been formatted in a 1.2MB drive.

Parameters

Item	Description
Device	Specifies the device containing the diskette to be formatted. The default is the **/dev/rfd0** device for drive 0.

Examples

1. To format a diskette in the **/dev/rfd0** device, enter:

```
format   -d   /dev/rfd0
```

2. To format a diskette without checking for bad tracks, enter:

```
format   -f
```

3. To format a 360KB diskette in a 5.25-inch, 1.2MB diskette drive in the **/dev/rfd1** device, enter:

```
format   -l   -d   /dev/rfd1
```

4. To format a 3.5-inch, low-density (720KB) diskette, enter:

```
format -d /dev/fd0.9
```

5. To format a 3.5-inch, high-capacity (1.44MB) diskette, enter:

```
format -d /dev/fd0.18
```

6. To format a read/write optical disk in the **/dev/romd0** device, enter:

```
format -d /dev/romd0
```

Files

Item	Description
/usr/sbin/format	Contains the **format** command.
/dev/rfd*	Specifies the device parameters.
/dev/fd*	Specifies the device parameters.
/dev/romd*	Specifies the device parameters.
/dev/omd*	Specifies the device parameters.

fortune Command

Purpose

Displays a random fortune from a database of fortunes.

Syntax

fortune [**-**] [**-s** | **-l** | **-a** [**-w**]] [*File*]

Description

The **fortune** command displays a fortune from either the **fortunes.dat** file or the file specified by the *File* parameter. After displaying the fortune, the **fortune** command exits.

Flags

Item	Description
-	Displays the usage summary.
-a	Displays either type of fortune.
-l	Displays long fortunes only.
-s	Displays short fortunes only.
-w	Waits after displaying a fortune to allow the user time to read the fortune.

Files

Item	Description
/usr/games	Location of the system's games.
/usr/games/lib/fortune/fortunes.dat	Location of the default **fortune** database.

forw Command

Purpose

Forwards messages.

Syntax

forw [**+** *Folder*] [**-draftfolder** **+***Folder* | **-nodraftfolder**] [*Message*] [**-draftmessage** *Message*] [**-digest** *Name* [**-issue** *Number*] [**-volume** *Number*]] [**-form** *FormFile*] [**-editor** *Editor* | **-noedit**] [**-whatnowproc** *Program* | **-nowhatnowproc**] [**-filter***File*] [**-annotate** [**-inplace** | **-noinplace**] | **-noannotate**] [**-format** | **-noformat**] [**-help**]

Description

The **forw** command starts an interface for forwarding messages. By default, the **forw** command interface:

- Opens for editing a *UserMhDirectory***/draft** file.
- Prompts the user to enter forwarding information based on the template defined in the **/etc/mh/mhl.forward** file.
- Prompts the user to enter any additional text that should accompany the forwarded message.

To complete editing of the *UserMhDirectory***/draft** file, press the Ctrl-D sequence. The **forw** command appends the current message from the current folder to the **draft** file. If you want to append more than one message, use the *Messages* parameter.

Note: A line of dashes or a blank line must be left between the header and the body of the message for the message to be identified when it is sent.

Upon exiting the editor, the **forw** command starts the `What Now?` prompt. Press the Enter key to see a list of the available **whatnow** subcommands. These subcommands enable you to continue to edit the message, list the message, direct the disposition of the message, or end the processing of the **forw** command.

The **forw** command allows you to change the format of the forwarded message with the **-form** flag. By default, the command uses the default message format located in your *UserMhDirectory***/forwcomps** file. If you have not defined your own **forwcomps** file, the **/etc/mh/forwcomps** file is used.

Use the **-annotate** flag to annotate the original message with forwarding information. To ensure annotation, send the forwarded note before exiting the **forw** command interface.

Note: The **-annotate** flag is not preserved over multiple executions of the **forw** command on the same draft.

Flags

Item	Description
-annotate	Annotates the forwarded messages with the lines: `Forwarded: Date` `Forwarded: Addresses` Use the **-inplace** flag to force annotation in place. This preserves links to the annotated message.
-digest *Name*	Uses the digest facility to create a new issue for the digest specified by the *Name* variable. The **forw** command expands the format strings in the **components** file (using the same format string mechanism used by the **repl** command) and composes the draft using the standard digest encapsulation algorithm. After the draft has been composed, the **forw** command writes out the volume and issue entries for the digest and starts the editor. Unless you specify the **-form** flag, the **forw** command uses the format in the *UserMhDirectory***/digestcomps** file. If this file does not exist, the command uses the default specified in the **/etc/mh/digestcomps** file.
-draftfolder +*Folder*	Places the draft message in the specified folder. If you do not specify this flag, the **forw** command selects a default draft folder according to the information supplied in the Message Handler (MH) profiles. If **+***Folder* is not specified, the `Current-Folder` is assumed. You can define a default draft folder in the **$HOME/.mh_profile** file. **Note:** If **-draftfolder +***Folder* is followed by a *Message* parameter, it is the same as specifying the **-draftmessage** flag.
-draftmessage *Message*	Identifies a draft message. If you specify **-draftfolder** without the **-draftmessage** flag, then the default message is `new`.
-editor *Editor*	Specifies the initial editor for preparing the message.
-filter *File*	Reformats each message being forwarded and places the reformatted message in the draft message. The **-filter** flag accepts formats used by the **mhl** command.
+*Folder*	Specifies the folder that contains the messages you want to forward. If a folder is not specified, `Current-Folder` is assumed.
-form *FormFile*	Displays the **forw** command output in the format specified by the *FormFile* variable. The **forw** command treats each line in the specified file as a format string. If the **-digest** flag is also specified, the **forw** command uses the form specified by the *File* variable as the format of the digest. If the **-form** flag is not specified when the **-digest** flag is used, the digest filter file becomes the form default.
-format	Using the **mhl** command and a default format file, reformats each message being forwarded and places the reformatted message in the draft message. If the *UserMhDirectory***/mhl.forward** file exists, it contains the default format. Otherwise, the **/etc/mh/mhl.forward** file contains the default format.

Item	Description
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-inplace	Forces annotation to be done in place to preserve links to the annotated message.
-issue *Number*	Specifies the issue number of the digest. The default issue number is one greater than the current value of the `DigestName-issue-list` entry in the *UserMhDirectory***/context** file.
Message	Specifies a message. You can specify several messages, a range of messages, or a single message. Use the following references when specifying messages: ***Number*** Number of the message. ***Sequence*** A group of messages specified by the user. Recognized values include: **all** All messages in the folder. **cur or . (period)** Current message. This is the default. **first** First message in a folder. **last** Last message in a folder. **new** New message that is created. **next** Message following the current message. **prev** Message preceding the current message The default message is the current message in the current folder. When you specify several messages, the first message forwarded becomes the current message. When you specify a folder, that folder becomes the current folder.
-noannotate	Prevents annotation of the original message. This flag is the default.
-nodraftfolder	Places the draft in the *UserMhDirectory***/draft** file.
-noedit	Suppresses the initial edit.
-noformat	Prevents reformatting of the messages being forwarded. This flag is the default.
-noinplace	Prevents annotation in place. This flag is the default.
-nowhatnowproc	Prevents interactive processing of the **forw** command. With this flag, no editing occurs.

Item	Description
-volume *Number*	Specifies the volume number of the digest. The default volume number is the current value of the `DigestName-volume-list` entry in the *UserMhDirectory***/context** file.
-whatnowproc *Program*	Starts the specified program to guide you through the forwarding tasks. **Note:** If you specify the **whatnow** command for *Program*, the **forw** command starts an internal **whatnow** procedure instead of a program with the file name **whatnow**.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Current-Folder:`	Sets the default current folder.
`Draft-Folder:`	Sets the default folder for drafts.
`Editor:`	Sets the default editor.
`fileproc:`	Specifies the program used to refile messages.
`mhlproc:`	Specifies the program used to filter messages being forwarded.
`Msg-Protect:`	Sets the protection level for the new message files.
`Path:`	Specifies the *UserMhDirectory*.
`whatnowproc:`	Specifies the program used to prompt `What now?` questions.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To forward the current message to another person, enter:

   ```
   forw
   ```

   The system prompts you to enter information in the header fields. To skip a field, press the Enter key. You must enter information in the `To:` field. The system responds with:

   ```
   ---------Enter initial text
   ```

   Enter the text you want displayed before the text of the forwarded message, and press the Ctrl-D key sequence. The text of the forwarded message is displayed, and you are prompted with `What now?` Enter `send` after the `What now?` prompt to forward the message.

2. To forward message 5 from the `inbox` folder, enter:

   ```
   forw    +inbox   5
   ```

Files

Item	Description
/etc/mh/digestcomps	Defines the MH default message form when the **-digest** flag is specified.
/etc/mh/mhl.forward	Contains the default MH message filter.
UserMhDirectory/**digestcomps**	Specifies a user's default message form when the **-digest** flag is specified. (If it exists, it overrides the MH default message filter.)
UserMhDirectory/**forwcomps**	Contains a user's default message form.
UserMhDirectory/**mhl.forward**	Contains a user's default message filter. (If it exists, it overrides the MH default message filter.)
/usr/bin/forw	Contains the executable form of the **forw** command.
$HOME/.mh_profile	Contains the file that customizes MH for an individual user.
UserMhDirectory/**draft**	Contains the draft created for editing messages.
/etc/mh/forwcomps	Defines components for the messages created by the **forw** command.

fpm Command

Purpose

Manages the permissions on commands and daemons owned by privileged users with setuid or setgid permissions.

Syntax

fpm [**-l** *level* [**-f** *file*] [[**-c**] [**-p**]] [**-v**]] | [**-s**] | [**-q**] | [**-?**]

Description

The **fpm** command allows administrators to harden their system by disabling the setuid and setgid bits on many commands in the operating system. This command is intended to remove the setuid permissions from commands and daemons owned by privileged users, but you can also customize it to address the specific needs of unique computer environments.

The setuid programs on the base AIX operating system have been grouped to allow for levels of hardening. This grouping allows administrators to choose the level of hardening according to their system environment. Additionally, you can use the **fpm** command to customize the list of programs that need to be disabled in your environment. You must review the levels of disablement and choose the right level for your environment.

Changing execution permissions of commands and daemons with the **fpm** command affects non-privileged users, denying their access to these commands and daemons or functions of the commands and daemons. Additionally, other commands that call or depend on these commands and daemons can be affected. Any user-created scripts that depend on commands and daemons with permissions that were altered by the **fpm** command cannot operate as expected when run by non-privileged users. Give full consideration to the effect and potential impact of modifying default permissions of commands and daemons.

You must perform appropriate testing before using this command to change the execution permissions of commands and daemons in any critical computer environment. If you encounter problems in an environment where execution permissions have been modified, restore the default permissions and

recreate the problem in this default environment to ensure the issue is not due to lack of appropriate execution permissions.

The **fpm** command provides the capability to restore the original AIX installation default permissions using the **-l default** flag.

Additionally, the **fpm** command logs the permission state of the files prior to changing them. The **fpm** log files are created in the **/var/security/fpm/log/*date_time*** file. If necessary, you can use these log files to restore the system's file permissions recorded in a previously saved log file.

When the **fpm** command is used on files that have extended permissions, it disables the extended permissions, though any extended permission data that existed prior to the **fpm** invocation is retained in the extended ACL.

Customized configuration files can be created and enacted as part of the high, medium, low, and default settings. File lists can be specified in the **/usr/lib/security/fpm/custom/high/*** directory, the **/usr/lib/security/fpm/custom/medium/*** directory, and the **/usr/lib/security/fpm/custom/default/*** directory. To take advantage of this feature, create a file containing a list of files that you want to be automatically processed in addition to the **fpm** commands internal list. When the **fpm** command is run, it also processes the lists in the corresponding customized directories. To see an example of the format for a customized file, view the **/usr/lib/security/fpm/data/high_fpm_list** file. The default format can be viewed in the **/usr/lib/security/fpm/data/default_fpm_list.example** file. For the customization of the **-l low** flag, the **fpm** command reads the same files in the **/usr/lib/security/fpm/custom/medium** directory, but only removes the setgid permissions, whereas the **-l medium** flag removes both the setuid and setgid permissions.

The **fpm** command cannot run on TCB-enabled hosts.

Flags

Item	Description
-l *level*	Specifies that the file permissions are changed according to the level specified. **-l high** High-level security. This flag removes the setuid and setgid permissions for computer systems that fall into the category of high-level security. This flag uses the list of files in the **/usr/lib/security/fpm/data/high_fpm_list** file and the **/usr/lib/security/fpm/custom/high/*.*** file as input by default, but an alternate input file can be selected with the **-f** flag. **-l medium** Medium-level security. This flag removes the setuid and setgid permissions for computer systems that fall into the category of medium-level security. This flag uses the list of files in the **/usr/lib/security/fpm/data/med_fpm_list** file and the **/usr/lib/security/fpm/custom/med/*.*** file as input by default. An alternate input file can be selected with the **-f** flag. **-l low** Low-level security. This flag removes only the setuid permission for computer systems that fall into the category of low-level security. This flag uses the list of files in the **/usr/lib/security/fpm/data/med_fpm_list** file and the **/usr/lib/security/fpm/custom/med/*.*** file as input by default. An alternate input file can be selected with the **-f** flag. **-l default** Returns the system commands previously modified by the **fpm** command to their default out-of-the-box permissions, if the commands were previously altered using the level of high, medium or low. This option reads the **/usr/lib/security/fpm/custom/default/*.*** file and sets the permissions defined in the file.
-s	Displays the status of the changes last made by the **fpm** command. The status is written in the **/usr/lib/security/fpm/data/status_fpm** file. The security level is represented as a whole integer from 1-5 (inclusive).
-f *file*	Allows the specification of a file list to override the default input file, where the *file* parameter is a file name containing the list of files to be used as input. This flag must be used along with the **-l high|medium|low|default** or the **-c** flag. When using a level of high, medium or low, the input file format is as follows: *full_path*/*filename* For example, `/usr/sbin/foo`. When used with the **-l default** flag, the input file format is as follows: *octet_permissions full_path*/*filename* There must be a space between the *octet_permissions* variable and the *full_path* variable. For example, `0750 /usr/sbin/foo`. The **-f** format allows for the specific control of the list of files being affected.

Item	Description
-c	Checks the files permissions, but takes no action. The **fpm** command returns 0 if no files were found out of compliance. If one or more files contain non-compliant permissions, this option lists the non-compliant file(s) and returns 1. This flag must be used with the **-l *level*** option. For example, if the **-c** and the **-l high** flags are used together, the **fpm** command checks the files listed in the **/usr/lib/security/fpm/data/high_fpm_list** file and removes their setuid and setgid permissions. The **-f *file*** flag can also be used with the **-c** option.
-v	Verbose output.
-p	Previews the changes the **fpm** command is to make, but takes no action. This flag must be used in conjunction with the **-l *level*** flag.
-q	Quit mode, which minimizes output and suppresses warnings.
-?	Prints the usage statement.

Exit Status

Item	Description
0	Success.
Non-zero	Failure or partial failure. Use the **-v** flag for more details.

Security

The **fpm** command reduces the number of commands with setuid and setgid permissions.

Examples

1. To apply the **fpm** command's low level security settings, enter:

   ```
   fpm –l low
   ```

   This command also processes any file list in the **/usr/lib/security/fpm/custom/med/** directory.
2. To check if the system commands are presently set to **fpm** low-level permissions, enter:

   ```
   fpm –c –l low
   ```

   This command reports any file with permissions out of conformance.
3. To restore the traditional out-of-the-box default permissions, enter:

   ```
   fpm –l default
   ```

   This command also processes any file list in the **/usr/lib/security/fpm/custom/default/** directory.
4. To list, or give a preview of what permission changes are to be done to make the system compliant with the **fpm** command's high-level security without changing any file permissions, enter:

   ```
   fpm -l high –p
   ```

   This command also previews any file list in the **/usr/lib/security/fpm/custom/high/** directory.
5. To apply the **fpm** command's high level security settings, enter:

   ```
   fpm –l high
   ```

   This command also processes any file list in the **/usr/lib/security/fpm/custom/high/** directory.

6. To list the current status of the system as changed through the **fpm** command, enter:

```
fpm -s
```

7. If the **fpm -l *level*** command was run on 7 January 2007 at 8:00 a.m., then the permission state of the affected files was captured by the **fpm** command before it made any changes. To restore the file permissions to their state of 7 January 2007 at 8:00 a.m., enter:

```
fpm -l default -f /var/security/fpm/log/01072007_08:00:00
```

Files

Item	Description
/usr/lib/security/fpm/data/default_list_example	Contains the default out-of-the-box permissions and files.
/usr/lib/security/fpm/data/high_fpm_list	Contains the list of files whose permissions can be changed with the **-l high** flag.
/usr/lib/security/fpm/data/med_fpm_list	Contains the list of files whose permissions can be changed with the **-l medium** or **-l low** flag.
/usr/lib/security/fpm/custom/high/*	Files in this directory can be used as user-configured input when the **-l high** level is selected. These files must contain a list of files, from which the **fpm** command removes setuid and setgid permissions.
/usr/lib/security/fpm/custom/medium/*	Files in this directory serve the same function as the high-level directory, but are used with the **-l medium** flag and the **-l low** flag.
/usr/lib/security/fpm/custom/default/*	Files in this directory serve the same function as the high-level directory, but are used with the **-l default** flag. **Note:** These files must be in the same format as the **/usr/lib/security/fpm/data/default_list_example** file.
/usr/lib/security/fpm/data/status_fpm	Contains the status of the file permissions changed from the last run of the **fpm** command.
/var/security/fpm/log/*date_time*	Contains the list of files changed by the **fpm** command corresponding to the data and the time at which the command was run. This file can be used as the input file of the **-f** flag to restore permissions to this instance.

frcactrl Command

Purpose

Controls and configures FRCA.

Syntax

frcactrl { **load** | **unload** }**frcactrl open** *Ip_Address Port* [*Virtual_Host*] *Server_Name Virtual_Root Log_File***frcactrl close** *Ip_Address Port* [*Virtual_Host*]**frcactrl loadfile** *Ip_Address Port* [*Virtual_Host*] *Document_Root File ...***frcactrl stats** [reset] [*Interval*]**frcactrl logging** *Ip_Address Port* [*Virtual_Host*] { on | off } [*Format*] [*CPU_Id*] **frcactrl** { **start** | **stop** } *Ip_Address Port* [*Virtual_Host*] **frcactrl revaltimeout** *Ip_Address Port* [*Virtual_Host*] [*Seconds*] **frcactrl pctonintr** [*Percentage*]**frcactrl set** { option=*value* }**frcactrl getfrcactrl default** [*option*]

Description

The **frcactrl** command controls and configures the FRCA kernel extension. The kernel extension must be loaded before starting any Web servers that want to use FRCA.

Subcommands

load
Loads the FRCA kernel extension if not loaded.

unload
Unloads the FRCA kernel extension if loaded.

open *Ip_Address Port* [*Virtual_Host*] *Server_Name Virtual_Root Log_File*
Opens and configures an FRCA instance under the name *Server_Name* for IP address *Ip_Address* on port *Port*. The *Virtual_Root* parameter specifies the directory where the Web data starts. The requests will be logged in the file specified by *Log_File*. This filename must be fully qualified.

Note: FRCA only supports one log file. When running more than one Web server on a system with FRCA, all requests will be logged to the same file.

close *Ip_Address Port* [*Virtual_Host*]
Closes the FRCA instance associated with the specified IP address and port.

loadfile *Ip_Address Port* [*Virtual_Host*] *Document_Root File ...*
Loads the specified file(s) into the FRCA / Network Buffer Cache. The IP and Port number at which the FRCA instance has been opened earlier must be specified here along with the document root and the file(s) to be loaded.

stats [reset] [*Interval*]
Displays FRCA statistics. The optional **reset** subcommand clears (zeros) the statistics. You can display the statistics at a regular interval by specifying the duration of the interval in seconds with the *Interval* parameter.

logging *Ip_Address Port* [*Virtual_Host*] { on | off } [*Format*] [*CPU_Id*]
Turns logging of request served by an FRCA instance bound to the specified *Ip_Address* and *Port* on or off. The format can be one of CLF, V-CLF, or ECLF (Common Log Format, Virtual Host & CLF, Extended CLF). The FRCA logging thread can also be bound to a particular CPU by specifying the optional *CPU_Id* parameter on multiprocessor machines.

start *Ip_Address Port* [*Virtual_Host*]
Enables the kernel get engine to serve requests sent to the specified IP and port.

stop *Ip_Address Port* [*Virtual_Host*]
Disables the kernel get engine for the specified IP and port.

revaltimeout *Ip_Address Port* [*Virtual_Host*] [*Seconds*]
Changes the revalidation timeout value for an FRCA instance at the specified address and port. The timeout value must be specified in seconds.

pctonintr [*Percentage*]
Controls the percentage of CPU time that can be spent in interrupt context. If this value is too low then FRCA will send requests up to Web server more often since it always executes in interrupt context. Any value >= 100 will result in FRCA serving every request that is cached in the FRCA cache.

set {option=*value*}
Sets the specified FRCA option to the value. The only option currently available is **frca_hashsz** which sets the number of slots in the FRCA hash table to the specified value. The default value of **frca_hashsz** is 12841. If changed, the value used must be prime as this results in a more even distribution of hash table entries.

get
Displays all FRCA options available along with their current values. Only one option called **frca_hashsz** currently exists.

default [*option*]
Sets the value of all options to their default values when used without specifying an option name. If an option name is specified it sets only the value of the specified option to its default.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. The following are examples of using the **open** subcommand:

   ```
   frcactrl open 9.1.1.1 80 ici imgcache01 /htdocs /logs/frca.log bin
   ```

   ```
   frcactrl open 9.1.1.2 80 ici imgcache02 /htdocs /logs/frca.log bin
   ```

   In the above examples "ici" is the virtual host name which could be used to access one of the mirrors imgcache01 or imgcache02. The IP address may be 0.0.0.0 if the Web server is not bound to a specific IP address.
2. To close the FRCA instance associated with IP address 9.1.1.1 and port 80, type:

   ```
   frcactrl close 9.1.1.1 80
   ```
3. To load the content of files /a/b/c/d and /a/b/c/e with URLs /d and /e, type:

   ```
   frcactrl loadfile /a/b/c /a/b/c/d e
   ```
4. To display the FRCA statistics, type:

   ```
   frcactrl stats
   ```

   This will cause the FRCA statistics to be displayed. They will look similar to this:

   ```
          Total    Deferred       Cache       Cache    Resource
       Requests    Requests        Hits      Misses      Errors
    ------------------------------------------------------------
     1024065396         227  1024065168           1           0
   ```
5. This examples shows how to use the **start** subcommand for virtual host "ici":

   ```
   frcactrl start 9.1.1.1 80 ici
   ```

   Note: The virtual host parameter is optional.
6. To disable the kernel get engine for port 80 on IP address 9.1.1.1 on virtual host "ici", type:

   ```
   frcactrl stop 9.1.1.1 80 ici
   ```
7. The following example sets the revalidation timeout value for the FRCA instance at port 80 of IP address 9.1.1.1 to 100 seconds:

```
frcactrl revaltimeout 9.1.1.1 80 100
```

8. To allow the CPU to spend 98 percent of its time in interrupt context, type:

```
frcactrl pctonintr 98
```

9. To set the value of the **frca_hashsz** option to 24499, type:

```
frcactrl set frca_hashsz=24499
```

10. To set the value of **frca_hashsz** to its default, type:

```
frcactrl default frca_hashsz
```

Files

/usr/bin/frcactrl

from Command

Purpose

To determine whom mail is from.

Syntax

from [**-d** *Directory*] [**-s** *Sender*] [*user*]

Description

The **from** command displays the message headings in your mailbox file to show you whom mail is from. If you specify *user*, the *user* mailbox is examined instead of your own (provided that you have read permission to user's mailbox).

Flags

Item	Description
-d *Directory*	Specifies the system mailbox directory.
-s *Sender*	Prints message headers only for mail sent by *Sender*.

Parameters

Item	Description
user	Specifies the *user* mailbox that is examined instead of your own (provided that you have read permission to the user's mailbox).

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the message headings in your mailbox, enter:

```
from
```

The names of the senders and message dates are displayed.

2. To display the message headings for mail sent by a specific user, enter:

   ```
   from -s dale
   ```

   In this example, only the message headings of the messages sent from user `dale` are displayed.

3. To display the message headings in a specific user's mailbox, enter:

   ```
   from   dawn
   ```

   In this example, the message headings from user `dawn ' s` mailbox are displayed (provided that you have read permission to `dawn ' s` mailbox).

4. To view all messages bob received from `jane`, enter:

   ```
   from -d /var/spool/mail -s jane bob
   ```

   This allows you to see all messages that bob received from `jane`, provided you have the permissions (such as root).

Files

Item	Description
/var/spool/mail/*	System mailboxes for all users.
/usr/bin/from	User mailbox files.

fsck Command

Purpose

Checks file system consistency and interactively repairs the file system.

Syntax

fsck [**-n**] [**-p**] [**-y**] [**-d***BlockNumber*] [**-f**] [**-i***i-NodeNumber*] [**-o** *Options*] [**-t***File*] [**-V** *VfsName*] [*FileSystem1 - FileSystem2 ...*]

Description

Attention: Always run the **fsck** command on file systems after a system malfunction. Corrective actions may result in some loss of data. The default action for each consistency correction is to wait for the operator to enter `yes` or `no`. If you do not have write permission for an affected file system, the **fsck** command defaults to a no response in spite of your actual response.

Notes:

1. The **fsck** command does not make corrections to a mounted file system.
2. The **fsck** command can be run on a mounted file system for reasons other than repairs. However, inaccurate error messages may be returned when the file system is mounted.

The **fsck** command checks and interactively repairs inconsistent file systems. You should run this command before mounting any file system. You must be able to read the device file on which the file system resides (for example, the **/dev/hd0** device). Normally, the file system is consistent, and the **fsck** command merely reports on the number of files, used blocks, and free blocks in the file system. If the file system is inconsistent, the **fsck** command displays information about the inconsistencies found and prompts you for permission to repair them.

The **fsck** command is conservative in its repair efforts and tries to avoid actions that might result in the loss of valid data. In certain cases, however, the **fsck** command recommends the destruction of a damaged file. If you do not allow the **fsck** command to perform the necessary repairs, an inconsistent file system may result. Mounting an inconsistent file system may result in a system crash.

If a JFS2 file system has snapshots, the `fsck` command will attempt to preserve them. If this action fails, the snapshots cannot be guaranteed to contain all of the before-images from the snapped file system. The `fsck` command will delete the snapshots and the snapshot logical volumes. Internal snapshots are deleted if the **fsck** command modifies the file system.

If you do not specify a file system with the *FileSystem* parameter, the **fsck** command checks all file systems listed in the **/etc/filesystems** file for which the **check** attribute is set to True. You can enable this type of checking by adding a line in the stanza, as follows:

```
check=true
```

You can also perform checks on multiple file systems by grouping the file systems in the **/etc/filesystems** file. To do so, change the check attribute in the **/etc/filesystems** file as follows:

```
check=Number
```

The *Number* parameter tells the **fsck** command which group contains a particular file system. File systems that use a common log device should be placed in the same group. File systems are checked, one at a time, in group order, and then in the order that they are listed in the **/etc/filesystems** file. All `check=true` file systems are in group 1. The **fsck** command attempts to check the root file system before any other file system regardless of the order specified on the command line or in the **/etc/filesystems** file.

The **fsck** command checks for the following inconsistencies:

- Blocks or fragments allocated to multiple files.
- i-nodes containing block or fragment numbers that overlap.
- i-nodes containing block or fragment numbers out of range.
- Discrepancies between the number of directory references to a file and the link count of the file.
- Illegally allocated blocks or fragments.
- i-nodes containing block or fragment numbers that are marked free in the disk map.
- i-nodes containing corrupt block or fragment numbers.
- A fragment that is not the last disk address in an i-node. This check does not apply to compressed file systems.
- Files larger than 32KB containing a fragment. This check does not apply to compressed file systems.
- Size checks:
  - Incorrect number of blocks.
  - Directory size not a multiple of 512 bytes.

  These checks do not apply to compressed file systems.
- Directory checks:
  - Directory entry containing an i-node number marked free in the i-node map.
  - i-node number out of range.
  - Dot (.) link missing or not pointing to itself.
  - Dot dot (..) link missing or not pointing to the parent directory.
  - Files that are not referenced or directories that are not reachable.
- Inconsistent disk map.
- Inconsistent i-node map.

Orphaned files and directories (those that cannot be reached) are, if you allow it, reconnected by placing them in the **lost+found** subdirectory in the root directory of the file system. The name assigned is the i-node number. If you do not allow the **fsck** command to reattach an orphaned file, it requests permission to destroy the file.

In addition to its messages, the **fsck** command records the outcome of its checks and repairs through its exit value. This exit value can be any sum of the following conditions:

Item	Description
0	All checked file systems are now okay.
2	The **fsck** command was interrupted before it could complete checks or repairs.
4	The **fsck** command changed the file system; the user must restart the system immediately.
8	The file system contains unrepaired damage.

The **fsck** command requires exclusive access to the underlying logical volume device of the file system. If fsck fails because the underlying device is unavailable, then fsck should be retried after the device is free to be opened.

When the system is booted from a disk, the boot process explicitly runs the **fsck** command, specified with the **-f** and **-p** flags on the **/**, **/usr**, **/var**, and **/tmp** file systems. If the **fsck** command is unsuccessful on any of these file systems, the system does not boot. Booting from removable media and performing maintenance work will then be required before such a system will boot.

If the **fsck** command successfully runs on **/**, **/usr**, **/var**, and **/tmp**, normal system initialization continues. During normal system initialization, the **fsck** command specified with the **-f** and **-p** flags runs from the **/etc/rc** file. This command sequence checks all file systems in which the **check** attribute is set to True (`check=true`). If the **fsck** command executed from the **/etc/rc** file is unable to guarantee the consistency of any file system, system initialization continues. However, the mount of any inconsistent file systems may fail. A mount failure may cause incomplete system initialization.

Note: By default, the **/**, **/usr**, **/var**, and **/tmp** file systems have the **check** attribute set to False (`check=false`) in their **/etc/filesystem** stanzas. The attribute is set to False for the following reasons:

1. The boot process explicitly runs the **fsck** command on the **/**, **/usr**, **/var**, and **/tmp** file systems.
2. The **/**, **/usr**, **/var**, and **/tmp** file systems are mounted when the **/etc/rc** file is executed. The **fsck** command will not modify a mounted file system. Furthermore, the **fsck** command run on a mounted file system produces unreliable results.

You can use the System Management Interface Tool (SMIT) **`smit fsck`** fast path to run this command.

Flags

Item	Description
-d*BlockNumber*	Searches for references to a specified disk block. Whenever the **fsck** command encounters a file that contains a specified block, it displays the i-node number and all path names that refer to it. For JFS2 filesystems, the i-node numbers referencing the specified block will be displayed but not their path names."

Item	Description
-f	Performs a fast check. Under normal circumstances, the only file systems likely to be affected by halting the system without shutting down properly are those that are mounted when the system stops. The **-f** flag prompts the **fsck** command not to check file systems that were unmounted successfully. The **fsck** command determines this by inspecting the **s_fmod** flag in the file system superblock. This flag is set whenever a file system is mounted and cleared when it is unmounted successfully. If a file system is unmounted successfully, it is unlikely to have any problems. Because most file systems are unmounted successfully, not checking those file systems can reduce the checking time.
-i*i-NodeNumber*	Searches for references to a specified i-node. Whenever the **fsck** command encounters a directory reference to a specified i-node, it displays the full path name of the reference.
-n	Assumes a no response to all questions asked by the **fsck** command; does not open the specified file system for writing.
-o *Options*	Passes comma-separated options to the **fsck** command. The following options are currently supported for JFS (these options are obsolete for newer file systems and can be ignored): **mountable** Causes the **fsck** command to exit with success, returning a value of 0, if the file system in question is mountable (clean). If the file system is not mountable, the **fsck** command exits returning with a value of 8. **mytype** Causes the **fsck** command to exit with success (0) if the file system in question is of the same type as either specified in the **/etc/filesystems** file or by the **-V** flag on the command line. Otherwise, 8 is returned. For example, `fsck -o mytype -V jfs /` exits with a value of 0 if **/** (the root file system) is a journaled file system.
-p	Does not display messages about minor problems but fixes them automatically. This flag does not grant the wholesale license that the **-y** flag does and is useful for performing automatic checks when the system is started normally. You should use this flag as part of the system startup procedures, whenever the system is being run automatically. If the primary superblock is corrupt, the secondary superblock is verified and copied to the primary superblock.
-t*File*	Specifies a *File* parameter as a scratch file on a file system other than the one being checked, if the **fsck** command cannot obtain enough memory to keep its tables. If you do not specify the **-t** flag and the **fsck** command needs a scratch file, it prompts you for the name of the scratch file. However, if you have specified the **-p** flag, the **fsck** command is unsuccessful. If the scratch file is not a special file, it is removed when the **fsck** command ends.
-V *VfsName*	Uses the description of the virtual file system specified by the *VFSName* variable for the file system instead of using the **/etc/filesystems** file to determine the description. If the **-V** *VfsName* flag is not specified on the command line, the **/etc/filesystems** file is checked and the **vfs=***Attribute* of the matching stanza is assumed to be the correct file system type.
-y	Assumes a yes response to all questions asked by the **fsck** command. This flag lets the **fsck** command take any action it considers necessary. Use this flag only on severely damaged file systems.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To check all the default file systems, enter:

   ```
   fsck
   ```

   This command checks all the file systems marked `check=true` in the **/etc/filesystems** file. This form of the **fsck** command asks you for permission before making any changes to a file system.
2. To fix minor problems with the default file systems automatically, enter:

   ```
   fsck -p
   ```

3. To check a specific file system, enter:

   ```
   fsck /dev/hd1
   ```

   This command checks the unmounted file system located on the **/dev/hd1** device.

Files

Item	Description
/usr/sbin/fsck	Contains the **fsck** command.
/etc/filesystems	Lists the known file systems and defines their characteristics.
/etc/vfs	Contains descriptions of virtual file system types.
/etc/rc	Contains commands (including the **fsck** command) that are run when the system is started.

fsck_cachefs Command

Purpose

Checks the integrity of data cached with CacheFS.

Syntax

fsck_cachefs [**-m**] [**-o noclean**] *cache_directory*

Description

The CacheFS version of the **fsck** command checks the integrity of a cache directory. By default it corrects any CacheFS problems it finds. There is no interactive mode. The most likely invocation of **fsck_cachefs** for CacheFS filesystems is at boot time from an entry in **/etc/rc.nfs**.

Flags

Item	Description
-m	Check, but do not repair.

Item	Description
-o noclean	Force a check on the cache even if there is no reason to suspect there is a problem.

Examples

To force a check on the cache directory, enter:

```
fsck_cachefs -o noclean /cache3
```

fsdb Command

Purpose

Debugs file systems.

Syntax

fsdb *FileSystem* [**-**]

Description

The **fsdb** command enables you to examine, alter, and debug a file system, specified by the *FileSystem* parameter. The command provides access to file system objects, such as blocks, i-nodes, or directories. You can use the **fsdb** command to examine and patch damaged file systems. Key components of a file system can be referenced symbolically. This feature simplifies the procedures for correcting control-block entries and for descending the file system tree.

To examine a file system, specify it by a block device name, a raw device name, or a mounted file system name. In the last case, the **fsdb** command determines the associated file system name by reading the **/etc/filesystems** file. Mounted file systems cannot be modified.

The **fsdb** command has a different interface for a JFS file system and a JFS2 file system. The following explains how to use **fsdb** with a JFS file system. See JFS2 Subcommands for information about JFS2 subcommands.

If the file system specified is a JFS2 snapshot, the **fsdb** command enables examination and modification of the snapshot superblock, snapshot map, block map xtree copy, and segment headers. See JFS2 Snapshot Subcommands for information about JFS2 snapshot subcommands.

The subcommands for the **fsbd** command allow you to access, view, or change the information in a file system. Any number you enter in the subcommand is considered decimal by default, unless you prefix it with either 0 to indicate an octal number or 0x to indicate a hexadecimal number. All addresses are printed in hexadecimal.

Because the **fsdb** command reads and writes one block at a time, it works with raw as well as with block I/O.

Flag

Item	Description
-	Disables the error checking routines used to verify i-nodes and block addresses. The **O** subcommand switches these routines on and off. When these routines are running, the **fsdb** command reads critical file system data from the superblock. The obtained information allows the **fsdb** command to access the various file system objects successfully and to perform various error checks.

Subcommands

The **fsdb** subcommands are requests to locate and display or modify information in the file system. The main categories of subcommands are:

Item	Description
Category	**Function**
Location	Access the information in the file system.
Display	View the information in the file system.
Modification	Change the information in the file system.

In addition, there are a few miscellaneous subcommands.

Location Subcommands

There are two types of location subcommands:

```
Number[ I | M | i | b ]
```

```
OR
```

```
dDirectorySlot
```

The first type consists of a number, optionally followed by an address specification. The address specification defines how the preceding number is to be interpreted. There are four address specifications corresponding to four different interpretations of the *Number* variable:

Item	Description
I	I-node map block number
M	Disk map block number
i	I-node number
b	Fragment number

Depending on the address specification (or absence of it), this type of location subcommand accesses information as follows:

Item	Description
Number	Accesses data at the absolute byte offset specified by the *Number* variable.
*MapBlockNumber***I**	Accesses the i-node map block indicated by the *MapBlockNumber* variable.
*MapBlockNumber***M**	Accesses the disk map block indicated by the *MapBlockNumber* variable.
*InodeNumber***i**	Accesses the i-node indicated by the *InodeNumber* variable.
*FragmentNumber***b**	Accesses the file system block indicated by the *FragmentNumber* variable. A fragment number consists of a block address and an encoded length. A complete fragment address is 32 bits in length. The low-order 28 bits are the beginning fragment address. The fragment length is encoded in the remaining 4 bits; it is encoded as the number of fragments less than a full block. For example, on a file system consisting of 1024-byte fragments, the address `0x2000010f` references a block that begins at 1KB block number `0x10f` and is 2KB in length. In contrast, on a file system of 512-byte fragments, the address `0x2000010f` references a block that begins at 512-byte block `0x10f` and is 3072 (512 * 6) bytes in length.

The second type of location subcommand is used to access directory entries. The subcommand consists of the character **d** followed by a directory-slot number. Directory-slot numbers start at 0 for each block of the associated i-node.

This type of location subcommand accesses information as follows:

Item	Description
d*DirectorySlot*	Accesses the directory entry indexed by the *DirectorySlot* variable for the current i-node. Only allocated directory entries can be manipulated using this location subcommand.

Display Subcommands

To view information relative to the address specification, use a display subcommand comprised of one of the display facilities in conjunction with one of the display formats, as follows:

p[*Number*]{ **i** | **d** | **o** | **e** | **c** | **b** | **y** | **M** | **I** | **x** | **s** | **D** }

OR

f[*Number*]{ **i** | **d** | **o** | **e** | **c** | **b** | **y** | **M** | **I** | **x** | **s** | **D** }

The display facilities are:

Item	Description
p	Indicates a general facility. Use the general display subcommand to display data relative to the current address. If you enter a number after the **p** symbol, the **fsdb** command displays that number of entries. A check is made to detect block boundary overflows. If you enter 0 or * (asterisk), the **fsdb** command displays all entries to the end of the current fragment.
f	Indicates a file facility. Use the file display subcommand to display data blocks associated with the current i-node. If you enter a number after the **f** symbol, the **fsdb** command displays that block of the file. Block numbering begins at 0. The display format follows the block number. If you enter **f** without a block number, the **fsdb** command defaults to displaying block 0 of the current i-node.

The display formats for either facility are:

Item	Description
i	Displays as i-nodes.
d	Displays as directories.
o	Displays as octal words.
e	Displays as decimal words.
c	Displays as characters.
b	Displays as octal bytes.
y	Displays as hexadecimal bytes.
M	Displays as disk map entries.
I	Displays as i-node map entries.
x	Displays as hexadecimal words.
S	Displays as single indirect blocks.
D	Displays as double indirect blocks.

The chosen display facility and display format remain in effect during the processing of the **fsdb** command until explicitly changed. You may receive an error message indicating improper alignment if the address you specify does not fall on an appropriate boundary.

If you use the *Number*, *MapBlockNumber***I**, or *FragmentNumber***b** location subcommands to access i-node information, you can step through the data, examining each byte, word, or double word. Select the desired display mode by entering one of the following subcommands:

Item	Description
B	Begins displaying in byte mode.
D	Begins displaying in double-word mode.
W	Begins displaying in word mode.

You can move forward or backward through the information. The boundary advances with the display screen and is left at the address of the last item displayed. The output can be ended at any time by pressing the INTERRUPT key. The following symbols allow movement through the information:

Item	Description
+ *Number*	Moves forward the specified number of units currently in effect.
-*Number*	Moves backward the specified number of units currently in effect.

The following symbols allow you to store the current address and return to it conveniently:

Item	Description
>	Stores the current address.
<	Returns to the previously stored address.

You can use dots, tabs, and spaces as subcommand delimiters, but they are only necessary to delimit a hexadecimal number from a subcommand that could be interpreted as a hexadecimal digit. Pressing the Enter key (entering a blank line) increments the current address by the size of the data type last displayed. That is, the address is set to the next byte, word, double word, directory entry, or i-node, allowing you to step through a region of a file system.

The **fsdb** command displays information in a format appropriate to the data type. Bytes, words, and double words are displayed as a hexadecimal address followed by the hexadecimal representation of the data at that address and the decimal equivalent enclosed in parentheses. The **fsdb** command adds a **.B** or **.D** suffix to the end of the address to indicate a display of byte or double word values. It displays directories as a directory slot offset followed by the decimal i-node number and the character representation of the entry name. It displays i-nodes with labeled fields describing each element. The environment variables control the formats of the date and time fields.

Modification Subcommands

You can modify information relative to the address specification by using a field specification (for fields in the i-node and fields in the directory). The general form for assigning new values is: *mnemonic operator new-value*, where the *mnemonic* parameter represents one of the fields described in the following list:

The following mnemonics are used for the names of the fields of an i-node and refer to the current working i-node:

Item	Description
`md`	Permission mode
`ln`	Link count
`uid`	User number

Item	Description
`gid`	Group number
`sz`	File size
a*Number*	Data block numbers (0 to 8) where the *Number* parameter can be a location subcommand
`at`	Access time
`mt`	Modification time
`maj`	Major device number
`min`	Minor device number

The following mnemonics refer to the i-node and disk maps:

Item	Description
`mf`	Map free count
`ms`	Map size
`mp`	Permanent allocation bit map
`mw`	Working allocation bit map

The following mnemonics are used for the names of the fields in directories:

Item	Description
`rl`	Length of directory entry record
`nl`	Length of directory name
`nm`	Directory name

Valid values of the *Operator* parameter include:

Note: A file system must be unmounted before attempting to modify it.

Item	Description
=	Assigns the *New-Value* parameter to the specified *Mnemonic* parameter.
=+	Increment the *Mnemonic* parameter by the specified *New-Value* parameter. The default *New-Value* parameter is a value of one.
=-	Decrease the *Mnemonic* by the specified *New-Value*. The default *New-Value* is a value of one.
="	Assigns the character string specified by the *New-Value* parameter to the specified *Mnemonic* parameter. If the current display format is the **d** address specification for directory and a mnemonic is not specified, the directory name is changed. The new directory name cannot be longer than the previous directory name.

Miscellaneous Subcommands

Miscellaneous subcommands are:

Item	Description
q	Quits.

Item	Description
x*n*	Expands a directory by *n* bytes where *n* plus the current size of the directory is not greater than the current directory's fragment in bytes.
!	Escapes to the shell.
O	Toggles error checking.

JFS2 Subcommands

These subcommands can be entered by their entire name or by using a subset of the name. At least the bold letters must be entered.

Item	Description
a[lter] <block> <offset> <hex string>	Alters disk data.
b[map] [<block number>]	Displays block allocation map.
dir[ectory] <inode number> [<fileset>] [R]	Displays directory entries.
d[isplay] [<block> [<offset> [<format> [<count>]]]]	Displays data.
dt[ree] {<block number> |<inode number>{a | f } }	Displays dtree nodes.
h[elp] [<command>]	Provides help on subcommands.
ia[g] [<IAG number>] [a | <fileset>]	Displays IAG pages.
im[ap] [a | <fileset>]	Displays inode allocation map.
i[node] [<inode number>] [a | <fileset>]	Displays inodes.
q[uit]	Exits fsdb.
su[perblock] [p | s]	Displays superblock.
x[tree] {<block number> | <inode number>{a | f}}	Displays xtree nodes.

a[lter] <block> <offset> <hex string>

where:

Item	Description
<block>	block number (decimal)
<offset>	offset within block (hex)
<hex string>	string of hex digits

Alters disk data. <hex string> should contain an even number of digits.

b[map] [<block numbers>]

Display Block Allocation Map.

<block number> Display the **dmap** page which describes this block number

Subcommands:

Item	Description
m	modify current node
u	visit upper level **bmap** page
l	visit left sibling
r	visit right sibling
w	display wmap
p	display pmap
s	display stree
x	exit subcommand mode

dir[ectory] <inode number> [<fileset>][R]

Item	Description
<inode number>	inode number of directory (decimal)
<fileset>	number, currently must be zero
R	recursively lists all subdirectories

Displays directory entries.

d[isplay] [<block> [<offset> [<format>[<count>]]]]

Item	Description
<block>	block number (decimal)
<offset>	offset within block (hex)
<format>	format in which to display data (see below)
<count>	number of objects to display (decimal)

Displays data in a variety of formats.

Format may be one of the following:

Item	Description
a	ascii

Item	Description	
i	inode	struct dinode
I	inode allocation map	iag_t
s	superblock	struct superblock
x	hexadecimal	

dt[ree] {<block number> | <inode number>{a | f}}

Item	Description
<block number>	block number containing a dtree page
<inode number>	inode number of directory (decimal)
{a | f}	'a' indicates inode number is an aggregate inode. 'f' indicates inode number is a fileset inode.

Displays root of the directory btree and enters a subcommand mode in which to navigate the btree.

Subcommands:

Item	Description
m	Modifies current node
f	Walks freelist entries
s	Displays specified slot entry
[0-9]+	Displays specified stbl entry
t	Displays formatted stbl
u	Visits parent node (not parent directory)
d	Visits child node
x	Exits subcommand mode

h[elp] [<command>]

Item	Description
<command>	command name

Prints help text. Lists all commands if no parameter.

ia[g] [<IAG number>] [a | <fileset>]

Item	Description
<IAG number>	IAG number (decimal)
a	use aggregate inode table
<fileset>	fileset number (currently must be zero)

Displays iag information and enters subcommand mode.

Subcommands:

Item	Description
e	Displays/modifies inode extents map
m	Modifies iag
p	Displays/modifies persistent map
w	Displays/modifies working map

im[ap] [a | <fileset>]

Item	Description
a	use aggregate inode table
<fileset>	fileset number (currently must be zero)

Display specified inode map and enters subcommand mode.

Subcommands:

Item	Description
e	Displays/modifies inode extents map
m	Modifies iag
p	Displays/modifies persistent map

i[node] [<inode number>] [a | <fileset>]

Item	Description
<inode number>	Inode number (decimal)
a	Use aggregate inode table
fileset	Fileset number (currently must be zero)

Displays inode information and enters subcommand mode.

Subcommands:

Item	Description
m	Modifies inode
t	Displays/modifies inode's b-tree
e	display/modify inode's EAs

Note: The **fsdb** command understands both the **v1** and the **v2** extended attribute formats. The behavior when viewing EAs is dependent on the format for the inode being viewed.

For **v1**, after displaying the inode's EAs you can modify its `pxdTable` or `eaDirectory` entries. Specify modify option and then the `pxdTable` or `eaDirectory` indicator and the offset into the table.

For **v2** the EAs are displayed using the **dtree** subcommand format. All of the **dtree** subcommands are then available for further action on the EAs.

q[uit]
Exits fsdb.

su[perblock] [p | s]

Item	Description
p	Displays primary superblock
s	Displays secondary superblock

Displays superblock data.

x[tree] {<block number> | <inode number>{a | f} }

Item	Description
<block number>	block number (decimal)
<inode number>	inode number
{a | f}	'a' indicates inode number is an aggregate inode. 'f' indicates inode number is a fileset inode.

Displays one node of a xtree and enters a subcommand mode in which to navigate the xtree.

Subcommands:

Item	Description
m	Modifies current node
u	Visits parent node
d	Visits child node
n	Visits right sibling
p	Visits left sibling
s	Selects xad entry to view
x	Exits subcommand mode

JFS2 Snapshot Subcommands

These subcommands can be entered by their entire name or by using a subset of the name. At least the bold letters must be entered.

Item	Description
a[lter] <block> <offset> <hex string>	Alters disk data.
b[map]	Displays block map xtree copy.
d[isplay] [<block> [<offset> [<format> [<count>]]]]	Displays data.
h[elp] [<command>]	Provides help on subcommands.
q[uit]	Exits fsdb.
st[able] [<block number>]	Displays summary snapshot table.
s[map] <block number>	Displays snapshot bit map.
su[perblock]	Displays superblock.

a[lter] <block> <offset> <hex string>

where:

Item	Description
<block>	block number (decimal)
<offset>	offset within block (hex)
<hex string>	string of hex digits

Alters disk data. <hex string> should contain an even number of digits.

b[map]

Displays block map xtree copy.

d[isplay] [<block> [<offset> [<format>[<count>]]]]

Item	Description
<block>	block number (decimal)
<offset>	offset within block (hex)
<format>	format in which to display data (see below)
<count>	number of objects to display (decimal)

Displays data in a variety of formats.

Format may be one of the following:

Item	Description
a	ascii
s	snapshot segment header
t	snapshot table page
x	xtree page

h[elp] [<command>]

Item	Description
<command>	command name

Provides help on subcommands.

q[uit]
Exits fsdb.

st[able] [<block number>]
where:

Item	Description
<block number>	block number (decimal)

Displays summary snapshot table.

s[map] [<block number>]
where:

Item	Description
<block number>	block number (decimal)

Displays snapshot bit map.

su[perblock]

Displays superblock.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

The following examples show subcommands you can use after starting the **fsdb** command on a JFS file system.

1. To display an i-node, enter:

   ```
   386i
   ```

   This command displays i-node 386 in i-node format. It now becomes the current i-node.
2. To change the link count for the current i-node to a value of 4, enter:

   ```
   ln=4
   ```
3. To increase the link count of the current i-node by a value of 1, enter:

   ```
   ln=+1
   ```
4. To display part of the file associated with the current i-node, enter:

   ```
   fc
   ```

   This command displays block 0 of the file associated with the current i-node in ASCII bytes.
5. To display entries of a directory, enter:

   ```
   2i.fd
   ```

   This changes the current i-node to the root i-node (i-node 2) and then displays the directory entries in the first block associated with that i-node. One or more of the last entries displayed may have an i-node number of 0 (zero). These are unused directory blocks; such entries cannot be manipulated as in the next example.
6. To go down a level of the directory tree, enter:

   ```
   d5i.fc
   ```

This command changes the current i-node to the one associated with directory entry 5. Then it displays the first block of the file as ASCII text (`fc`). Directory entries are numbered starting from 0.

7. To display a block when you know its block number, enter:

```
1b.p0o
```

This command displays the superblock (block 1) of file system in octal.

8. To change the i-node of a directory entry, enter:

```
2i.a0b.d7=3
```

This command changes the i-node of directory entry 7 in the root directory (`2i`) to 3. This example also shows how several operations can be combined on one line.

9. To change the file name of a directory entry, enter:

```
d7.nm="chap1.rec"
```

This command changes the name field of directory entry 7 to `chap1.rec`.

10. To display a given block of the file associated with the current i-node, enter:

```
a2b.p0d
```

This command displays block 2 of the current i-node as directory entries.

11. To display the content of a single indirect block at block 7, enter:

```
7b. p0S
```

This command displays the block numbers allocated to the i-node that has a single indirect block at block 7.

12. To display the first page of the disk map, enter:

```
OM
```

13. To display the first 10 words of permanent block allocation map in hexadecimal, enter:

```
mp1.p10x
```

This command shows the allocation bit map at the current address; for example, at 0M.

The following examples show some subcommands you can use on a JFS2 file system.

Attention: Do not use JFS2 subcommands to modify a file system.

1. To display an i-node, enter:

```
inode 2
```

This command displays i-node 2 in i-node format.

2. To display entries of a directory, enter:

```
dir 2
```

This command displays the directory entries associated with i-node 2.

3. To display a block whose block number is 0x1000, enter:

```
display 0x1000
```

This command displays the block at file system in hexadecimal format.

Files

Item	Description
/usr/sbin	Contains the **fsdb** command.
/etc/filesystems	Contains information on the file systems.

fsplit Command

Purpose

Splits FORTRAN source code into separate routine files.

Syntax

fsplit [**-e** *SubprogramUnit*] ... [*File*]

Description

The **fsplit** command takes as input either a file or standard input containing FORTRAN source code and splits the input into separate routine files of the form *name***.f**, where *name* is the name of the program unit (for example, function, subroutine, block data or program).

The name for unnamed block data subprograms has the form *blkdtaNNN***.f**, where NNN is three digits and a file of this name does not already exist. For unnamed main programs the name has the form *mainNNN***.f**. If there is an error in classifying a program unit, or if *name***.f** already exists, the program unit is put in a file of the form *zzzNNN***.f**, where *zzzNNN***.f** does not already exist.

Note: The **fsplit** command assumes that the subprogram name is on the first non-comment line of the subprogram unit. Non-standard source formats can confuse the command and produce unpredictable results.

Flags

Item	Description
-e *SubprogramUnit*	Causes only the specified subprogram units to be split into separate files. Normally each subprogram unit is split into a separate file. The **-e** flag can be used only for named main programs and block data subprograms. If names specified via the **-e** option are not found, a diagnostic is written to standard error.

Example

The following **fsplit** command splits the subprograms `readit` and `doit` into separate files:

```
fsplit -e readit -e doit prog.f
```

Files

Item	Description
/usr/bin/fsplit	Contains the **fsplit** command.

ftp Command

Purpose

Transfers files between a local and a remote host.

Syntax

ftp [**-d**] [**-D** *DataConnTimeOut*] [**-g**] [**-i**] [**-n**] [**-v**] [**-f**] [**-K**] [**-k** *realm*] [**-q**] [**-C**] [**-s**] [**-M**] [*HostName* [*Port*]] [**-H**]

Description

The **ftp** command uses the File Transfer Protocol (FTP) to transfer files between the local host and a remote host or between two remote hosts. Remote execution of the **ftp** command is not recommended.

The FTP protocol allows data transfer between hosts that use dissimilar file systems. Although the protocol provides a high degree of flexibility in transferring data, it does not attempt to preserve file attributes (such as the protection mode or modification times of a file) that are specific to a particular file system. Moreover, the FTP protocol makes few assumptions about the overall structure of a file system and does not provide or allow such functions as recursively copying subdirectories.

Note: If you are transferring files between systems and need to preserve file attributes or recursively copy subdirectories, use the **rcp** command.

Issuing Subcommands

At the `ftp>` prompt, you can enter subcommands to perform tasks such as listing remote directories, changing the current local and remote directory, transferring multiple files in a single request, creating and removing directories, and escaping to the local shell to perform shell commands. See the Subcommands section for a description of each subcommand.

If you execute the **ftp** command and do not specify the *HostName* parameter for a remote host, the **ftp** command immediately displays the `ftp>` prompt and waits for an **ftp** subcommand. To connect to a remote host, execute the **open** subcommand**.** When the **ftp** command connects to the remote host, the **ftp** command then prompts for the login name and password before displaying the `ftp>` prompt again. The **ftp** command is unsuccessful if no password is defined at the remote host for the login name.

The **ftp** command interpreter, which handles all subcommands entered at the `ftp>` prompt, provides facilities that are not available with most file-transfer programs, such as:

- Handling file-name parameters to **ftp** subcommands
- Collecting a group of subcommands into a single subcommand macro
- Loading macros from a **$HOME/.netrc** file

These facilities help simplify repetitive tasks and allow you to use the **ftp** command in unattended mode.

The command interpreter handles file-name parameters according to the following rules:

- If a - (hyphen) is specified for the parameter, standard input (stdin) is used for read operations and standard output (stdout) is used for write operations.
- If the preceding check does not apply and file-name expansion is enabled (see the **-g** flag or the **glob** subcommand), the interpreter expands the file name according to the rules of the C shell. When globbing is enabled and a pattern-matching character is used in a subcommand that expects a single file name, results may be different than expected.

  For example, the **append** and **put** subcommands perform file-name expansion and then use only the first file name generated. Other **ftp** subcommands, such as **cd**, **delete**, **get**, **mkdir**, **rename**, and **rmdir**, do not perform file-name expansion and take the pattern-matching characters literally.
- For the **get**, **put**, **mget**, and **mput** subcommands, the interpreter has the ability to translate and map between different local and remote file-name syntax styles (see the **case**, **ntrans**, and **nmap**

subcommands) and the ability to modify a local file name if it is not unique (see the **runique** subcommand). Additionally, the **ftp** command can send instructions to a remote **ftpd** server to modify a remote file name if it is not unique (see the **sunique** subcommand).

- Use double quotes (" ") to specify parameters that include blank characters.

  Note: The **ftp** command interpreter does not support pipes. It also does not necessarily support all multibyte-character file names.

To end an **ftp** session when you are running interactively, use the **quit** or **bye** subcommand or the End of File (Ctrl-D) key sequence at the `ftp>` prompt. To end a file transfer before it has completed, press the Interrupt key sequence. The default Interrupt key sequence is Ctrl-C. The **stty** command can be used to redefine this key sequence.

The **ftp** command normally halts transfers being sent (from the local host to the remote host) immediately. The **ftp** command halts transfers being received (from the remote host to the local host) by sending an FTP ABOR instruction to the remote FTP server and discarding all incoming file transfer packets until the remote server stops sending them. If the remote server does not support the ABOR instruction, the **ftp** command does not display the `ftp>` prompt until the remote server has sent all of the requested file. Additionally, if the remote server does something unexpected, you may need to end the local **ftp** process.

Security and Automatic Login

If Standard is the current authentication method:

The **ftp** command also handles security by sending passwords to the remote host and permits automatic login, file transfers, and logoff.

If you execute the **ftp** command and specify the host name (*HostName*) of a remote host, the **ftp** command tries to establish a connection to the specified host. If the **ftp** command connects successfully, the **ftp** command searches for a local **$HOME/.netrc** file in your current directory or home directory. If the file exists, the **ftp** command searches the file for an entry initiating the login process and command macro definitions for the remote host. If the **$HOME/.netrc** file or automatic login entry does not exist or if your system has been secured with the **securetcpip** command, the **ftp** command prompts the user for a user name and password. The command displays the prompt whether or not the *HostName* parameter is specified on the command line.

Note: The queuing system does not support multibyte host names.

If the **ftp** command finds a **$HOME/.netrc** automatic login entry for the specified host, the **ftp** command attempts to use the information in that entry to log in to the remote host. The **ftp** command also loads any command macros defined in the entry. In some cases (for example, when the required password is not listed in an automatic login entry), the **ftp** command prompts for the password before displaying the `ftp>` prompt.

Once the **ftp** command completes the automatic login, the **ftp** command executes the **init** macro if the macro is defined in the automatic login entry. If the **init** macro does not exist or does not contain a **quit** or **bye** subcommand, the **ftp** command then displays the `ftp>` prompt and waits for a subcommand.

Note: The remote user name specified either at the prompt or in a **$HOME/.netrc** file must exist and have a password defined at the remote host. Otherwise, the **ftp** command fails.

If Kerberos 5 is the current authentication method

The **ftp** command will use the extensions to ftp specifications as defined in IETF draft document "draft-ietf-cat-ftpsec-09.txt". The FTP security extensions will be implemented using the Generic Security Service API (GSSAPI) security mechanism. The GSSAPI provides services independent to the underlying security and communication mechanism. The GSSAPI is defined in rfc 1508 and 1509.

The **ftp** command will use the AUTH and ADAT commands to authenticate with the **ftpd** daemon. If both support Kerberos authentication, then they will use the local users DCE credentials to authenticate the user on the remote system. If this fails and Standard authentication is configured on both systems, the process described above will be used.

The *HostName* parameter is the name of the host machine to which files are transferred. The optional *Port* parameter specifies the ID of the port through which to transmit. (The **/etc/services** file specifies the default port.)

Note: If the value of the registry is correctly set to the current authentication scheme, the FTP authentication works with the active directory password. If the value of registry is set to null, then the default value of files (local user authentication) is used.

Transport Layer Security support

The **ftp** command supports Transport Layer Security (TLS) as defined in RFC 4217. TLS is a cryptographic protocol that provides secure communications between clients and servers.

The **ftp** command uses the **AUTH TLS** and **PROT P** commands to secure the communication with the **ftpd** daemon. If both the **AUTH TLS** and **PROT P** commands support the TLS protocol, then a secure channel is established. Only the Standard Authentication method is supported.

If the **-s** flag is specified when you run the **ftp** command, then the **ftp** command searches for a local **$HOME/.ftpcnf** file in the your home directory. If the file is found, the **ftp** command uses the following configuration parameters to set up a TLS session with the server. If the file is not found or the configuration parameters are missing, the **ftp** command attempts to connect to the server without using the configuration parameters.

CRL_PATH
The CRL_PATH parameter provides the path to the certificate revocation list file, which must be in privacy enhanced mail (PEM) format. If specified, the digital certificate that is provided by the server is verified against the certificate revocation list. If the certificate was revoked, the TLS session fails. If not specified, the digital certificate is not verified against a certificate revocation list.

CA_PATH
The CA_PATH parameter provides the path to the certificate authority file, which must be in PEM format. If specified, the server certificate is verified against the certificate authority. If the digital certificate that is provided by the server was not signed by the security authority, the TLS session fails. If not specified, the digital certificate that is provided by the server is not verified against a certificate revocation list.

CIPHER_LIST
If the CIPHER_LIST parameter is specified, the list is used during the TLS session. If not, a default cipher list is used.

DEPTH
If the CA_PATH configuration parameter is specified, the DEPTH value is used to verify the certificate that is provided by the **ftpd** server in the digital certificate hierarchy. If not provided, a default value of 9 is used.

CERTIFICATE
The CERTIFICATE parameter provides a path to the chain file of a valid digital certificate in PEM format. This file is used in the TLS session.

CERTIFICATE_PRIVATE_KEY
The CERTIFICATE_PRIVATE_KEY parameter contains the path to the certificate private key, in PEM format, which is used during the TLS session. To support TLS, you must install the latest version of the OpenSSL tool from the AIX Web Download Pack Programs website.

For Trusted AIX system

The user is assigned a default login Sensitivity Label (SL) and Integrity Label (TL), which is the effective SL and effective TL of the user's process after a successful login. If the user does not want to log in at the default login SL, the user can choose to supply a different SL at the login time by using the **-e** option. The SL supplied by the user must be dominated by the user's clearance and contained in the system accreditation range. The TL cannot be specified by the user at login time. The default login SL and TL are defined in the **/etc/security/user** file along with the user name and the clearance for each user. To use the **-e** option, the server side's kernel trusted network bit must be turned off.

Note: Any user with user ID less than or equal to 128 cannot log in to the remote Trusted AIX system.

Flags

Item	Description
-C	Allows the user to specify that the outgoing file sent using the **send_file** command must be cached in the Network Buffer Cache (NBC). This flag cannot be used unless the **-q** flag is specified. This flag is only applicable when a file is being sent out in the binary mode with no protection.
-d	Sends debugging information about **ftp** command operations to the **syslogd** daemon. If you specify the **-d** flag, you must edit the **/etc/syslog.conf** file and add one of the following entries: `user.info FileName` OR `user.debug FileName` **Note:** The **syslogd** daemon debug level includes info level messages. If you do not edit the **/etc/syslog.conf** file, no messages are produced. After changing the **/etc/syslog.conf** file, run the **refresh -s syslogd** or **kill -1** *SyslogdPID* command to inform the **syslogd** daemon of the changes to its configuration file. For more information about debug levels, refer to the **/etc/syslog.conf** file. Also, refer to the **debug** subcommand**.**
-D *DataConnTimeOut*	Specifies the maximum number of seconds that the **ftp** command holds a data connection. The default value is 300 seconds and can range from 300 seconds to 3600 seconds.
-f	Causes the credentials to be forwarded. This flag will be ignored if Kerberos 5 is not the current authentication method.
-g	Disables the expansion of metacharacters in file names. Interpreting metacharacters can be referred to as expanding (sometimes called globbing) a file name. See the **glob** subcommand.
-H	Turns on audit logging for the `FILE_Unlink` event if the event is enabled for the user.
-i	Turns off interactive prompting during multiple file transfers. See the **prompt**, **mget**, **mput**, and **mdelete** subcommands for descriptions of prompting during multiple file transfers.
-K	Disables the SO_KEEPALIVE option defined in the `sys/socket.h` file on both the control and data connection.
-k *realm*	Allows the user to specify the realm of the remote station if it is different from the local systems realm. For these purposes, a realm is synonymous with a DCE cell. This flag will be ignored if Kerberos 5 is not the current authentication method.
-M	Prevents the **ftp** command from being blocked after a file is transferred between a local and a remote host.
-n	Prevents an automatic login on the initial connection. Otherwise, the **ftp** command searches for a **$HOME/.netrc** entry that describes the login and initialization process for the remote host. See the **user** subcommand.
-q	Allows the user to specify that the **send_file** subroutine must be used for sending the file on the network. This flag is only applicable when a file is being sent out in the binary mode with no protection.

Item	Description
-v	Displays all the responses from the remote server and provides data transfer statistics. This display mode is the default when the output of the **ftp** command is to a terminal, such as the console or a display. If stdin is not a terminal, the **ftp** command disables verbose mode unless the user invoked the **ftp** command with the **-v** flag or issued the **verbose** subcommand.
-s	Starts a TLS session with the server by sending an **AUTH TLS** command and a **PROT P** command to the **ftpd** daemon. If the TLS session is established, and you are authenticated by using the Standard Authentication method, the transfer of the data and commands is encrypted.

Subcommands

The following **ftp** subcommands can be entered at the `ftp>` prompt. Use double quotes (" ") to specify parameters that include blank characters.

Item	Description
![*Command* [*Parameters*]]	Invokes an interactive shell on the local host. An optional command, with one or more optional parameters, can be given with the shell command.
$*Macro* [*Parameters*]	Executes the specified macro, previously defined with the **macdef** subcommand. Parameters are not expanded.
?[*Subcommand*]	Displays a help message describing the subcommand. If you do not specify a *Subcommand* parameter, the **ftp** command displays a list of known subcommands.
account [*Password*]	Sends a supplemental password that a remote host may require before granting access to its resources. If the password is not supplied with the command, the user is prompted for the password. The password is not displayed on the screen.
append *LocalFile* [*RemoteFile*]	Appends a local file to a file on the remote host. If the remote file name is not specified, the local file name is used, altered by any setting made with the **ntrans** subcommand or the **nmap** subcommand. The **append** subcommand uses the current values for **form**, **mode**, **struct**, and **type** subcommands while appending the file.
ascii	Synonym for the **type ascii** subcommand.
bell	Sounds a bell after the completion of each file transfer.
binary	Synonym for the **type binary** subcommand.
block	Synonym for the **mode block** subcommand.
bye	Ends the file-transfer session and exits the **ftp** command. Same as the **quit** subcommand.
carriage-control	Synonym for the **form carriage-control** subcommand.
case	Sets a toggle for the case of file names. When the **case** subcommand is On, the **ftp** command changes remote file names displayed in all capital letters from uppercase to lowercase when writing them in the local directory. The default is Off (so the **ftp** command writes uppercase remote file names in uppercase in the local directory).
cd *RemoteDirectory*	Changes the working directory on the remote host to the specified directory.
cdup	Changes the working directory on the remote host to the parent of the current directory.

Item	Description
close	Ends the file-transfer session, but does not exit the **ftp** command. Defined macros are erased. Same as the **disconnect** subcommand.
copylocal	Toggles local copy. **copylocal** defaults to off. An effort is made by ftp to make sure you do not zero out a file by ftp'ing it to itself (eg. same hostname, same pathname). Turning **copylocal** ON bypasses this check.
cr	Strips the carriage return character from a carriage return and line-feed sequence when receiving records during ASCII-type file transfers. (The **ftp** command terminates each ASCII-type record with a carriage return and line feed during file transfers.) Records on remote hosts with operating systems other than the one you are running can have single line feeds embedded in records. To distinguish these embedded line feeds from record delimiters, set the **cr** subcommand to Off. The **cr** subcommand toggles between On and Off.
debug [**0** | **1**]	Toggles debug record keeping On and Off. Specify **debug** or **debug 1** to print each command sent to the remote host and save the restart control file. Specify **debug** again, or **debug 0**, to stop the debug record keeping. The Ctrl-C key sequence also saves the restart control file. Specifying the **debug** subcommand sends debugging information about **ftp** command operations to the **syslogd** daemon. If you specify the **debug** subcommand, you must edit the **/etc/syslog.conf** file and add one of the following entries: `user.info FileName` OR `user.debug FileName` **Note:** The **syslogd** daemon debug level includes info level messages. If you do not edit the **/etc/syslog.conf** file, no messages are produced. After changing the **/etc/syslog.conf** file, run the **refresh -s syslogd** or **kill -1** *SyslogdPID* command to inform the **syslogd** daemon of the changes to its configuration file. For more information about debug levels, refer to the **/etc/syslog.conf** file. Also, refer to the **ftp -d** flag.
delete *RemoteFile*	Deletes the specified remote file.
dir [*RemoteDirectory*] [*LocalFile*]	Writes a listing of the contents of the specified remote directory (*RemoteDirectory*) to the specified local file (*LocalFile*). If the *RemoteDirectory* parameter is not specified, the **dir** subcommand lists the contents of the current remote directory. If the *LocalFile* parameter is not specified or is a - (hyphen), the **dir** subcommand displays the listing on the local terminal.
disconnect	Ends the file-transfer session but does not exit the **ftp** command. Defined macros are erased. Same as the **close** subcommand.
ebcdic	Synonym for the **type ebcdic** subcommand.
exp_cmd	Toggles between conventional and experimental protocol commands. The default is off.
file	Synonym for the **struct file** subcommand.

Item	Description
form [**carriage-control** | **non-print** | **telnet**]	Specifies the form of the file transfer. The **form** subcommand modifies the **type** subcommand to send the file transfer in the indicated form. Valid arguments are **carriage-control**, **non-print**, and **telnet**. **carriage-control** Sets the form of the file transfer to carriage-control. **non-print** Sets the form of the file transfer to non-print. **telnet** Sets the form of the file transfer to Telnet. Telnet is a Transmission Control Protocol/Internet Protocol (TCP/IP) protocol that opens connections to a system.
get *RemoteFile* [*LocalFile*]	Copies the remote file to the local host. If the *LocalFile* parameter is not specified, the remote file name is used locally and is altered by any settings made by the **case**, **ntrans**, and **nmap** subcommands**.** The **ftp** command uses the current settings for the **type**, **form**, **mode**, and **struct** subcommands while transferring the file.
glob	Toggles file-name expansion (globbing) for the **mdelete**, **mget**, and **mput** subcommands. If globbing is disabled, file-name parameters for these subcommands are not expanded. When globbing is enabled and a pattern-matching character is used in a subcommand that expects a single file name, results may be different than expected. For example, the **append** and **put** subcommands perform file-name expansion and then use only the first file name generated. Other **ftp** subcommands, such as **cd**, **delete**, **get**, **mkdir**, **rename**, and **rmdir**, do not perform file-name expansion and take the pattern-matching characters literally. Globbing for the **mput** subcommand is done locally in the same way as for the **csh** command. For the **mdelete** and **mget** subcommands, each file name is expanded separately at the remote machine and the lists are not merged. The expansion of a directory name can be different from the expansion of a file name, depending on the remote host and the **ftp** server. To preview the expansion of a directory name, use the **mls** subcommand: `mls RemoteFile` To transfer an entire directory subtree of files, transfer a **tar** archive of the subtree in binary form, rather than using the **mget** or **mput** subcommand.
hash	Toggles hash sign (#) printing. When the **hash** subcommand is on, the **ftp** command displays one hash sign for each data block (1024 bytes) transferred.
help [*Subcommand*]	Displays help information. See the **?** subcommand.
image	Synonym for the **type image** subcommand.
lcd [*Directory*]	Changes the working directory on the local host. If you do not specify a directory, the **ftp** command uses your home directory.
local *M*	Synonym for the **type local** *M* subcommand.
ls [*RemoteDirectory*] [*LocalFile*]	Writes an abbreviated file listing of a remote directory to a local file. If the *RemoteDirectory* parameter is not specified, the **ftp** command lists the current remote directory. If the *LocalFile* parameter is not specified or is a - (hyphen), the **ftp** command displays the listing on the local terminal.

Item	Description
macdef *Macro*	Defines a subcommand macro. Subsequent lines up to a null line (two consecutive line feeds) are saved as the text of the macro. Up to 16 macros, containing at most 4096 characters for all macros, can be defined. Macros remain defined until either redefined or a **close** subcommand is executed. The $ (dollar sign) and \ (backslash) are special characters in **ftp** macros. A $ symbol followed by one or more numbers is replaced by the corresponding macro parameter on the invocation line (see the **$** subcommand). A $ symbol followed by the letter i indicates that the macro is to loop, with the $i character combination being replaced by consecutive parameters on each pass. The first macro parameter is used on the first pass, the second parameter is used on the second pass, and so on. A \ symbol prevents special treatment of the next character. Use the \ symbol to turn off the special meanings of the $ and \. (backslash period) symbols.
mdelete *RemoteFiles*	Expands the files specified by the *RemoteFiles* parameter at the remote host and deletes the remote files.
mdir [*RemoteDirectories LocalFile*]	Expands the directories specified by the *RemoteDirectories* parameter at the remote host and writes a listing of the contents of those directories to the file specified in the *LocalFile* parameter. If the *RemoteDirectories* parameter contains a pattern-matching character, the **mdir** subcommand prompts for a local file if none is specified. If the *RemoteDirectories* parameter is a list of remote directories separated by blanks, the last argument in the list must be either a local file name or a - (hyphen). If the *LocalFile* parameter is - (hyphen), the **mdir** subcommand displays the listing on the local terminal. If interactive prompting is on (see the **prompt** subcommand), the **ftp** command prompts the user to verify that the last parameter is a local file and not a remote directory.
mget *RemoteFiles*	Expands the *RemoteFiles* parameter at the remote host and copies the indicated remote files to the current directory on the local host. See the **glob** subcommand for more information on file-name expansion. The remote file names are used locally and are altered by any settings made by the **case**, **ntrans**, and **nmap** subcommands. The **ftp** command uses the current settings for the **form**, **mode**, **struct**, and **type** subcommands while transferring the files.
mkdir [*RemoteDirectory*]	Creates the directory specified in the *RemoteDirectory* parameter on the remote host.
mls [*RemoteDirectories LocalFile*]	Expands the directories specified in the *RemoteDirectories* parameter at the remote host and writes an abbreviated file listing of the indicated remote directories to a local file. If the *RemoteDirectories* parameter contains a pattern-matching character, the **mls** subcommand prompts for a local file if none is specified. If the *RemoteDirectories* parameter is a list of remote directories separated by blanks, the last argument in the list must be either a local file name or a - (hyphen). If the *LocalFile* parameter is - (hyphen), the **mls** subcommand displays the listing on the local terminal. If interactive prompting is on (see the **prompt** subcommand), the **ftp** command prompts the user to verify that the last parameter is a local file and not a remote directory.

Item	Description
mode [**stream** | **block**]	Sets file-transfer mode. If an argument is not supplied, the default is **stream**. **block** Sets the file-transfer mode to block. **stream** Sets the file-transfer mode to stream.

Item	Description
modtime	Shows the last modification time of the specified file on the remote machine. If the **ftp** command is not connected to a host prior to execution, the **modtime** subcommand terminates with an error message. The **ftp** command ignores parameter beyond the first parameter. If the *FileName* parameter is not specified, the **ftp** command prompts for a file name. If no file name is given, the **ftp** command sends a usage message to standard output and terminates the subcommand. If the name specified by the *FileName* parameter exists on the remote host, and the name specifies a file, then the **ftp** command sends a message containing the last modification time of the file to standard output and terminates the subcommand. If *FileName* specifies a directory, the **ftp** command sends an error message to standard output and terminates the subcommand. **Note:** The **modtime** subcommand interprets metacharacters when allowed.
mput [*LocalFiles*]	Expands the files specified in the *LocalFiles* parameter at the local host and copies the indicated local files to the remote host. See the **glob** subcommand for more information on file-name expansion. The local file names are used at the remote host and are altered by any settings made by the **ntrans** and **nmap** subcommands. The **ftp** command uses the current settings for the **type**, **form**, **mode**, and **struct** subcommands while transferring the files.
nlist [*RemoteDirectory*] [*LocalFile*]	Writes a listing of the contents of the specified remote directory (*RemoteDirectory*) to the specified local file (*LocalFile*). If the *RemoteDirectory* parameter is not specified, the **nlist** subcommand lists the contents of the current remote directory. If the *LocalFile* parameter is not specified or is a **-** (hyphen), the **nlist** subcommand displays the listing on the local terminal.

Item	Description
nmap [*InPattern OutPattern*]	Turns the file-name mapping mechanism On or Off. If no parameters are specified, file-name mapping is turned off. If parameters are specified, source file names are mapped for the **mget** and **mput** subcommands and for the **get** and **put** subcommands when the destination file name is not specified. This subcommand is useful when the local and remote hosts use different file-naming conventions or practices. Mapping follows the pattern set by the *InPattern* and *OutPattern* parameters. The *InPattern* parameter specifies the template for incoming file names, which may have already been processed according to the **case** and **ntrans** settings. The template variables $1 through $9 can be included in the *InPattern* parameter. All characters in the *InPattern* parameter, other than the $ (dollar sign) and the \$ (backslash, dollar sign), are treated literally and are used as delimiters between *InPattern* variables. For example, if the *InPattern* parameter is `$1.$2` and the remote file name is `mydata.dat`, the value of $1 is `mydata` and the value of $2 is `dat`. The *OutPattern* parameter determines the resulting file name. The variables $1 through $9 are replaced by their values as derived from the *InPattern* parameter, and the variable $0 is replaced by the original file name. Additionally, the sequence [*Sequence1*,*Sequence2*] is replaced by the value of *Sequence1*, if *Sequence1* is not null; otherwise, it is replaced by the value of *Sequence2*. For example, the subcommand: `nmap $1.$2.$3 [$1,$2].[$2,file]` would yield `myfile.data` from `myfile.data` or `myfile.data.old`, `myfile.file` from `myfile`, and `myfile.myfile` from `.myfile`. Use the \ (backslash) symbol to prevent the special meanings of the $ (dollar sign), [(left bracket),] (right bracket), and , (comma) in the *OutPattern* parameter.
non-print	Synonym for the **form non-print** subcommand.
ntrans [*InCharacters* [*OutCharacters*]]	Turns the file-name character translation mechanism On and Off. If no parameters are specified, character translation is turned off. If parameters are specified, characters in source file names are translated for **mget** and **mput** subcommands and for **get** and **put** subcommands when the destination file name is not specified. This subcommand is useful when the local and remote hosts use different file-naming conventions or practices. Character translation follows the pattern set by the *InCharacters* and *OutCharacters* parameter. Characters in a source file name matching characters in the *InCharacters* parameter are replaced by the corresponding characters in the *OutCharacters* parameter. If the string specified by the *InCharacters* parameter is longer than the string specified by the *OutCharacters* parameter, the characters in the *InCharacters* parameter are deleted if they have no corresponding character in the *OutCharacters* parameter.
open *HostName* [*Port*]	Establishes a connection to the FTP server at the host specified by the *HostName* parameter. If the optional port number is specified, the **ftp** command attempts to connect to a server at that port. If the automatic login feature is set (that is, the **-n** flag was not specified on the command line), the **ftp** command attempts to log in the user to the FTP server. You must also have a **$HOME/.netrc** file with the correct information in it and the correct permissions set. The **.netrc** file must be in your home directory.

Item	Description
passive	Toggles passive mode for file transfers. When a file transfer command (such as **get**, **mget**, **put**, or **mput**) is invoked with passive mode off, the **ftp** server opens a data connection back to the client. In passive mode, the client opens data connections to the server when sending or receiving data.
private	Sets the protection level to private only when the authentication method is set. At this level, data integrity and confidentiality are protected.
prompt	Toggles interactive prompting. If interactive prompting is on (the default), the **ftp** command prompts for verification before retrieving, sending, or deleting multiple files during the **mget**, **mput**, and **mdelete** subcommands. Otherwise, the **ftp** command acts accordingly on all files specified.
protect	This command returns the current level of protection.
proxy [*Subcommand*]	Executes an **ftp** command on a secondary control connection. This subcommand allows the **ftp** command to connect simultaneously to two remote FTP servers for transferring files between the two servers. The first **proxy** subcommand should be an **open** subcommand to establish the secondary control connection. Enter the **proxy ?** subcommand to see the other **ftp** subcommands that are executable on the secondary connection. The following subcommands behave differently when prefaced by the **proxy** subcommand: • The **open** subcommand does not define new macros during the automatic login process. • The **close** subcommand does not erase existing macro definitions. • The **get** and **mget** subcommands transfer files from the host on the primary connection to the host on the secondary connection. • The **put**, **mput**, and **append** subcommands transfer files from the host on the secondary connection to the host on the primary connection. • The **restart** subcommand can be handled by the **proxy** command. • The **status** subcommand displays accurate information. File transfers require that the FTP server on the secondary connection must support the PASV (passive) instruction.
put *LocalFile* [*RemoteFile*]	Stores a local file on the remote host. If you do not specify the *RemoteFile* parameter, the **ftp** command uses the local file name to name the remote file, and the remote file name is altered by any settings made by the **ntrans** and **nmap** subcommands. The **ftp** command uses the current settings for the **type**, **form**, **mode**, and **struct** subcommands while transferring the files.
pwd	Displays the name of the current directory on the remote host.
quit	Closes the connection and exits the **ftp** command. Same as the **bye** subcommand.
quote *String*	Sends the string specified by the *String* parameter verbatim to the remote host. Execute the **remotehelp** or **quote help** subcommand to display a list of valid values for the *String* parameter. **Note:** "Quoting" commands that involve data transfers can produce unpredictable results.
record	Synonym for the **struct record** subcommand.
recv *RemoteFile* [*LocalFile*]	Copies the remote file to the local host. Same as the **get** subcommand.

Item	Description
reinitialize	Reinitializes an FTP session by flushing all I/O and allowing transfers to complete. Resets all defaults as if a user had just started an FTP session without logging in to a remote host.
remotehelp [*Subcommand*]	Requests help from the remote FTP server.
rename *FromName ToName*	Renames a file on the remote host.
reset	Clears the reply queue. This subcommand resynchronizes the command parsing.
restart get | **put** | **append**	Restarts a file transfer at the point where the last checkpoint was made. To run successfully, the subcommand must be the same as the aborted subcommand, including structure, type, and form. Valid arguments are **get**, **put**, and **append**.
rmdir *RemoteDirectory*	Removes the remote directory specified by the *RemoteDirectory* parameter at the remote host.
runique	(ReceiveUnique) Toggles the facility for creating unique file names for local destination files during **get** and **mget** subcommands. If this facility is Off (the default), the **ftp** command overwrites local files. Otherwise, if a local file has the same name as that specified for a local destination file, the **ftp** command modifies the specified name of the local destination file with `.1`. If a local file is already using the new name, the **ftp** command appends the postfix .2 to the specified name. If a local file is already using this second name, the **ftp** command continues incrementing the postfix until it either finds a unique file name or reaches .99 without finding a unique file name. If the **ftp** command cannot find a unique file name, the **ftp** command reports an error and the transfer does not take place. Note that the **runique** subcommand does not affect local file names generated from a shell command.
safe	Sets the protection level to "safe." At this level, data is integrity protected.
send *LocalFile* [*RemoteFile*]	Stores a local file on the remote host. Same as the **put** subcommand.
sendport	Toggles the use of FTP PORT instructions. By default, the **ftp** command uses a PORT instruction when establishing a connection for each data transfer. When the use of PORT instructions is disabled, the **ftp** command does not use PORT instructions for data transfers. The PORT instruction is useful when dealing with FTP servers that ignore PORT instructions while incorrectly indicating the instructions have been accepted.
site *Args*	Displays or sets the idle time-out period, displays or sets the file-creation umask, or changes the permissions of a file, using the **chmod** command. Possible values for the *Args* parameter are **umask** and **chmod**.
size *RemoteFile*	Displays the size in bytes of the remote file specified by the *RemoteFile* parameter.
status	Displays the current status of the **ftp** command as well as the status of the subcommands.
stream	Synonym for the **mode stream** subcommand.

Item	Description
struct [**file** | **record**]	Sets the data transfer structure type. Valid arguments are **file** and **record**. **file** Sets the data-transfer structure type to file. **record** Sets the data-transfer structure type to record.
sunique	(Send/Store Unique) Toggles the facility for creating unique file names for remote destination files during **put** and **mput** subcommands. If this facility is off (the default), the **ftp** command overwrites remote files. Otherwise, if a remote file has the same name as that specified for a remote destination file, the remote FTP server modifies the name of the remote destination file. Note that the remote server must support the STOU instruction.
system	Shows the type of operating system running on the remote machine.
telnet	Synonym for the **form telnet** subcommand.
tenex	Synonym for the **type tenex** subcommand.
trace	Toggles packet tracing.
type [**ascii** | **binary** | **ebcdic** | **image** | **local** *M* | **tenex**]	Sets the file-transfer type. Valid arguments are **ascii**, **binary**, **ebcdic**, **image**, **local** *M*, and **tenex**. If an argument is not specified, the current type is printed. The default type is **ascii**; the **binary** type can be more efficient than **ascii**. **ascii** Sets the file-transfer type to network ASCII. This type is the default. File transfer may be more efficient with binary-image transfer. See the **binary** argument for further information. **binary** Sets the file-transfer type to binary image. This type can be more efficient than an ASCII transfer. **ebcdic** Sets the file-transfer type to EBCDIC. **image** Sets the file-transfer type to binary image. This type can be more efficient than an ASCII transfer. **local *M*** Sets the file-transfer type to local. The *M* parameter defines the decimal number of bits per machine word. This parameter does not have a default. **tenex** Sets the file-transfer type to that needed for TENEX machines.
user *User* [*Password*] [*Account*]	Identifies the local user (*User*) to the remote FTP server. If the *Password* or *Account* parameter is not specified and the remote server requires it, the **ftp** command prompts for the password or account locally. If the *Account* parameter is required, the **ftp** command sends it to the remote server after the remote login process completes. **Note:** Unless automatic login is disabled by specifying the **-n** flag on the command line, the **ftp** command sends the *User*, *Password*, and *Account* parameters automatically for the initial connection to the remote server. You also need a **.netrc** file in your home directory in order to issue an automatic login.

Item	Description
verbose	Toggles verbose mode. When the verbose mode is on (the default), the **ftp** command displays all responses from the remote FTP server. Additionally, the **ftp** command displays statistics on all file transfers when the transfers complete.

Examples

1. To invoke the **ftp** command, log in to the system canopus, display local help information, display remote help information, display status, toggle the **bell**, **prompt**, **runique**, **trace**, and **verbose** subcommands, and then quit, enter:

```
$ ftp canopus
Connected to canopus.austin.century.com.
220 canopus.austin.century.com FTP server (Version 4.1 Sat Nov 23 12:52:09 CST 1991) ready.
Name (canopus:eric): dee
331 Password required for dee.
Password:
230 User dee logged in.
ftp> help
Commands may be abbreviated. Commands are:
!          delete          mdelete          proxy          runique
$          debug           mdir             sendport       send
account    dir             mget             put            size
append     disconnect      mkdir            pwd            status
ascii      form            mls              quit           struct
bell       get             mode             quote          sunique
binary     glob            modtime          recv           system
bye        hash            mput             remotehelp     tenex
case       help            nmap             rstatus        trace
cd         image           nlist            rhelp          type
cdup       lcd             ntrans           rename         user
close      ls              open             reset          verbose
cr         macdef          prompt           rmdir          ?
clear      private         protect          safe
ftp> remotehelp
214-The following commands are recognized(* =>'s unimplemented).
 USER   PORT   RETR   MSND*  ALLO   DELE   SITE*  XMKD   CDUP
 PASS   PASV   STOR   MSOM*  REST*  CWD    STAT*  RMD    XCUP
 ACCT*  TYPE   APPE   MSAM*  RNFR   XCWD   HELP   XRMD   STOU
 REIN*  STRU   MLFL*  MRSQ*  RNTO   LIST   NOOP   PWD
 QUIT   MODE   MAIL*  MRCP*  ABOR   NLST   MKD    XPWD
 AUTH   ADAT   PROT   PBSZ   MIC    ENC    CCC
214 Direct comments to ftp-bugs@canopus.austin.century.com.
ftp> status
Connected to canopus.austin.century.com.
No proxy connection.
Mode: stream; Type: ascii; Form: non-print; Structure: file
Verbose: on; Bell: off; Prompting: on; Globbing: on
Store unique: off; Receive unique: off
Case: off; CR stripping: on
Ntrans: off
Nmap: off
Hash mark printing: off; Use of PORT cmds: on
ftp> bell
Bell mode on.
ftp> prompt
Interactive mode off.
ftp> runique
Receive unique on.
ftp> trace
Packet tracing on.
ftp> verbose
Verbose mode off.
ftp> quit
$
```

2. To invoke the **ftp** command, log in to the system canopus, print the working directory, change the working directory, set the file transfer type to ASCII, send a local file to the remote host, change the working directory to the parent directory, and then quit, enter:

```
$ ftp canopus
Connected to canopus.austin.century.com.
```

```
220 canopus.austin.century.com FTP server (Version 4.1 Sat Nov 23 12:52:09 CST 1991) ready.
Name (canopus:eric): dee
331 Password required for dee.
Password:
230 User dee logged in.
ftp> pwd
257 "/home/dee" is current directory.
ftp> cd desktop
250 CWD command successful.
ftp> type ascii
200 Type set to A.
ftp> send typescript
200 PORT command successful.
150 Opening data connection for typescript (128.114.4.99,1412).
226 Transfer complete.
ftp> cdup
250 CWD command successful.
ftp> bye
221 Goodbye.
$
```

3. To invoke the **ftp** command with automatic logon (using the **.netrc** file), open a session with the system canopus, log in, change the working directory to the parent directory, print the working directory, list the contents of the current directory, delete a file, write a listing of the contents of the current directory to a local file, close the session, and then quit, enter:

```
$ ftp canopus
Connected to canopus.austin.century.com.
220 canopus.austin.century.com FTP server (Version 4.1 Sat Nov 23 12:52:09 CST 1991) ready.
331 Password required for dee.
230 User dee logged in.
ftp> cdup
250 CWD command successful.
ftp> pwd
257 "/home" is current directory.
ftp> dir
200 PORT command successful.
150 Opening data connection for /usr/bin/ls (128.114.4.99,1407)
(0 bytes).
total 104
drwxr-xr-x   2 system        32 Feb 23 17:55 bin
Drwxr-xr-x  26 rios        4000 May 30 17:18 bin1
drwxr-xr-x   2 system        32 Feb 23 17:55 books
drwxrwxrwx  18 rios        1152 Jun  5 13:41 dee
-r--r--r--   1 system      9452 May 17 12:21 filesystems
drwxr-xr-x   2 system        32 Feb 23 17:55 jim
drwxr-xr-x   5 system        80 Feb 23 17:55 krs
drwxrwxrwx   2 rios       16432 Feb 23 17:36 lost+found
-rwxr-xr-x   1 rios        3651 May 24 16:45 oldmail
drwxr-xr-x   2 system       256 Feb 23 17:55 pubserv
drwxrwxrwx   2 system       144 Feb 23 17:55 rein989
drwxr-xr-x   2 system       112 Feb 23 17:55 reinstall
226 Transfer complete.
ftp> delete oldmail
250 DELE command successful.
ftp> mdir /home/dee/bin binlist
output to local-file: binlist? y
200 PORT command successful.
150 Opening data connection for /usr/bin/ls (128.114.4.99,1408) (0 bytes).
226 Transfer complete.
ftp> close
221 Goodbye.
ftp> quit
$
```

Files

Item	Description
/usr/samples/tcpip/netrc	Contains the sample **.netrc** file.
/etc/syslog.conf	Contains configuration information for the **syslogd** daemon.

ftpd Daemon

Purpose

Provides the server function for the Internet FTP protocol.

Syntax

Note: The **ftpd** daemon is usually started by the **inetd** daemon. It can also be controlled from the command line, using SRC commands.

/usr/sbin/ftpd [**-d**] [**-D** *DataConnTimeOut*] [**-e**][**-f**] [**-ff**] [**-k**] [**-l**] [**-U**] [**-t** *TimeOut*] [**-T** *MaxTimeOut*] [**-s**] [**-u** *OctalVal*] [**-q** [**-C**]] [**-c**] [**-H**]

Description

The **/usr/sbin/ftpd** daemon is the DARPA Internet File Transfer Protocol (FTP) server process. The **ftpd** daemon uses the Transmission Control Protocol (TCP) to listen at the port specified with the **ftp** command service specification in the **/etc/services** file.

Changes to the **ftpd** daemon can be made using the System Management Interface Tool (SMIT) or System Resource Controller (SRC), by editing the **/etc/inetd.conf** or /**etc**/**services** file. Typing `ftpd` at the command line is not recommended. The **ftpd** daemon is started by default when it is uncommented in the **/etc/inetd.conf** file.

The **inetd** daemon gets its information from the /**etc**/**inetd.conf** file and the /**etc**/**services** file.

If you change the **/etc/inetd.conf** or /**etc**/**services** file, run the **refresh -s inetd** or **kill -1** *InetdPID* command to inform the **inetd** daemon of the changes to its configuration files.

The **ftpd** daemon expands file names according to the conventions of the **csh** command. This command allows you to use such metacharacters as the * (asterisk), the ? (question mark), [ ] (left and right brackets), { } (left and right braces), and the ~ (tilde).

ftpaccess.ctl File

The **/etc/ftpaccess.ctl** file is searched for lines that start with **allow:**, **deny:**, **readonly:**, **writeonly:**, **readwrite:**, **useronly:**, **grouponly:**, **herald:** and/or **motd:**. Other lines are ignored. If the file doesn't exist, then ftp access is allowed for all hosts. The **allow:** and **deny:** lines are for restricting host access. The **readonly:**, **writeonly:** and **readwrite:** lines are for restricting ftp reads (get) and writes (put). The **useronly:** and **grouponly:** lines are for defining anonymous users. The **herald:** and **motd:** lines are for multiline messages before and after login.

The syntax for all lines in **/etc/ftpaccess.ctl** is in the form:

```
keyword: value, value, ...
```

where you can specify one or more values for every keyword. You can have multiple lines with the same keyword. The lines in **/etc/ftpaccess.ctl** are limited to 1024 characters, anything more than 1024 characters will be ignored.

The syntax for the **allow:** and **deny:** lines are:

```
allow: host, host, ...
deny: host, host, ...
```

If an **allow:** line is specified, then only the hosts listed in all the **allow:** lines are allowed ftp access. All other hosts will be refused ftp access. If there is no **allow:** line, then all hosts will be given ftp access except those hosts specified in the **deny:** line(s). The host can be specified as either a hostname or IP address.

The syntax for the `readonly:`, `writeonly:` and `readwrite:` lines is:

```
readonly: dirname, dirname, ...
writeonly: dirname, dirname, ...
readwrite: dirname, dirname, ...
```

The **readonly:** lines list the read-only directories and the **writeonly:** lines list the write-only directories. Read access is denied in a write-only directory and write access is denied in a read-only directory. All other directories are granted access except when a **readwrite:** line is specified. If a **readwrite:** line is specified, only directories listed in the **readwrite:** line and/or listed in the **readonly:** line are granted access for reading, AND only directories listed in the **readwrite:** line and/or listed in the **writeonly:** line are granted access for writing. Also, these lines can have a value of "ALL" or "NONE".

The syntax for the **useronly:**, `puseronly:`, `grouponly:`, and **pgrouponly:** lines is:

```
useronly: username, username, ...
puseronly: username, username, ...
grouponly: groupname, groupname, ...
pgrouponly: groupname, groupname, ...
```

The username is from **/etc/passwd** and the groupname is from **/etc/group**. The **useronly:** and `puseronly:` lines define an anonymous user. The **grouponly:** and `pgrouponly:` lines define a group of anonymous users. These anonymous users are similar to the user anonymous in that ftp activity is restricted to their home directories. The `useronly:` and `grouponly:` lines define anonymous users similar to the user anonymous in that they are not password protected. The `puseronly:` and `pgrouponly:` lines define anonymous users that are password protected.

Note: For `puseronly:` and `pgrouponly:` users, passwords must be created and login must be disabled.

The syntax for the **herald:** and **motd:** lines are:

```
herald: path
motd: on|off
```

The path is the full path name of the file that contains the multiline herald that displays before login. When the **motd:** line has a value of 'on', then the **$HOME/motd** file contains the multiline message that displays after login. If the user is a defined anonymous user, then the **/etc/motd** file contains the multiline message that displays after login. (Note that **/etc/motd** is in the anonymous user's chroot'ed home directory). The default for the **motd:** line is off.

If the Standard Operating system authentication method is the current authentication method :

Before the **ftpd** daemon can transfer files for a client process, it must authenticate the client process. The **ftpd** daemon authenticates client processes according to these rules:

- The user must have a password in the password database, **/etc/security/passwd**. (If the user's password is not null, the client process must provide that password.)
- The user name must not appear in the **/etc/ftpusers** file.
- The user's login shell must appear in the shells attribute of the **/etc/security/login.cfg** file.
- If the user name is `anonymous`, `ftp` or is a defined anonymous user in the **/etc/ftpaccess.ctl** file, an anonymous FTP account must be defined in the password file. In this case, the client process is allowed to log in using any password. By convention, the password is the name of the client host. The **ftpd** daemon takes special measures to restrict access by the client process to the anonymous account.

If Kerberos 5 is the current authentication method:

The **ftpd** daemon allows access only if all of the following conditions are satisfied:

- The local user of the ftp client has current DCE credentials.
- The local and remote systems both support the **AUTH** command.
- The remote system accepts the DCE credentials as sufficient for access to the remote account. See the **kvalid_user** function for additional information.

Transport Layer Security support

The **ftpd** daemon supports Transport Layer Security (TLS) as defined in RFC 4217. TLS is a cryptographic protocol that provides secure communication between clients and servers.

The main purpose of the implementation is to secure the control and data connection using encryption. The client needs to be authenticated by other means. The only supported method is the Standard Authentication method.

Upon receiving a request to start a TLS session, the **ftpd** daemon proceeds to read the **/etc/ftpd.cnf** file, loading the following configuration parameters that will be used to set up the TLS session:

Item	Description
CRL_PATH	The *CRL_PATH* parameter provides the path to the certificate revocation list file, which must be in PEM format. If specified, the digital certificate provided by the client will be verified against the certificate revocation list. If the ftp client is not using a digital certificate, the connection will fail. If the client provides a digital certificate, but the certificate has been revoked, the TLS session will fail. If this parameter is not specified, the client does not have to provide a digital certificate.
CA_PATH	The *CA_PATH* parameter provides the path to the certificate authority file, which must be in PEM format. If specified, the client certificate will be verified against the certificate authority. If the client does not provide a digital certificate, the connection will fail. If the client provides a digital certificate, but the certificate has not been signed by the security authority, the TLS session will fail. If this parameter is not specified, the client does not have to provide a digital certificate.
CIPHER_LIST	If the *CIPHER_LIST* parameter is specified, the list is used during the TLS session. If not, a default cipher list is used.
DEPTH	If the *CA_PATH* configuration parameter has been specified, the *DEPTH* value is used to verify the certificate provided by the ftp client in the digital certificate hierarchy. If not provided, a default value of 9 is used.
CERTIFICATE	The *CERTIFICATE* parameter provides a path to a valid digital certificate chain file in PEM format. This file is used in the TLS session. This parameter needs to be specified to start a TLS session. If this parameter is not specified, the ftpd server rejects all TLS requests.
CERTIFICATE_PRIVATE_KEY	The *CERTIFICATE_PRIVATE_KEY* parameter provides the path to the certificate private key, which is in PEM format, and is used during the TLS session. This parameter needs to be specified to start a TLS session. If this parameter is not specified, the ftpd server rejects all TLS requests.
DH_PARAMETERS_DIR	The *DH_PARAMETERS_DIR* parameter provides the path to a directory containing *Diffie Helman* parameters in PEM format. More than one file containing *Diffie Helman* parameters in PEM format can be included in this directory. The **ftpd** daemon searches for the appropriate parameter to use if required.

To support TLS, you must install the latest version of the OpenSSL tool from the AIX Web Download Pack Programs website.

File Transfer Protocol Subtree Guidelines

When handling an anonymous FTP user, the server performs the **chroot** command in the home directory of the FTP user account. For greater security, implement the following rules when you construct the FTP subtree:

Item	Description
~ftp	Make the home directory owned by root and mode r-xr-xr-x (555).
~ftp/bin	Make this directory owned by the root user and not writable by anyone. The **ls** program must be present in this directory to support the list command. This program must have mode 111.
~ftp/etc	Make this directory owned by the root user and not writable by anyone.
~ftp/pub	Make this directory mode 777 and owned by FTP. Users must then place files that are to be accessible through the anonymous account in this directory.

Note: The shell script **/usr/samples/tcpip/anon.ftp** uses the above rules to set up the anonymous FTP account for you.

When handling an anonymous FTP user defined in **/etc/ftpaccess.ctl**, the server performs the **chroot** command in the home directory of the user account. For greater security, implement the following rules when you construct the user's subtree:

~user
Make the home directory owned by root and mode `r-xr-xr-x` (555).

~user/bin
Make this directory owned by the root user and unwritable by anyone. The **ls** program must be present in this directory to support the list command. This program must have mode 111.

~user/etc
Make this directory owned by the root user and unwritable by anyone.

~user/pub
Make this directory mode 777 and owned by user. Users must then place files that are to be accessible through the `anonymous` account in this directory.

Note: The shell script **/usr/samples/tcpip/anon.users.ftp** uses the above rules to set up the anonymous FTP account for you.

The server must run as the root user to create sockets with privileged port numbers. The server maintains an effective user ID of the logged-in user, reverting to the root user only when binding addresses to sockets.

Supported File Transfer Protocol Requests

The **ftpd** daemon currently supports the following FTP requests:

Item	Description
ABOR	Terminates previous command.
ACCT	Specifies account (ignored).
ADAT	Specifies the Authentication/Security Data.
ALLO	Allocates storage (vacuously).
APPE	Appends to a file.
AUTH	Specifies the Authentication/Security Mechanism.
CCC	Specifies the Clear Command Channel.
CDUP	Changes to the parent directory of the current working directory.
CWD	Changes working directory.

Item	Description
DELE	Deletes a file.
ENC	Specifies the Privacy Protected Command.
HELP	Gives help information.

Item	Description
LIST	Gives list files in a directory (this FTP request is the same as the **ls -lA** command).
MKD	Makes a directory.
MDTM	Shows last modification time of file.
MIC	Specifies the Integrity Protected Command.
MODE	Specifies data transfer mode.
NLST	Gives a name list of files in directory (this FTP request is the same as the **ls** command).
NOOP	Does nothing.
PASS	Specifies a password.
PASV	Prepares for server-to-server transfers.
PBSZ	Specifies the Protection Buffer Size.
PORT	Specifies a data connection port.
PROT	Specifies the Data Channel Protection Level.
PWD	Prints the current working directory.
QUIT	Terminates session.
RETR	Retrieves a file.
RMD	Removes a directory.
RNFR	Specifies rename-from file name.
RNTO	Specifies rename-to file name.
SITE	The following nonstandard or UNIX-specific commands are supported by the **SITE** request: **UMASK** Changes umask (**SITE UMASK 002**). **IDLE** Sets idler time (**SITE IDLE 60**). **CHMOD** Changes mode of a file (**SITE CHMOD 755** *FileName*). **HELP** Gives help information (**SITE HELP**).
SIZE	Returns size of current file.
STAT	Returns the status of the server.
STOR	Stores a file.
STOU	Stores a file using a unique file name.
STRU	Specifies the structure of data transfer as a file structure.
SYST	Shows operating system type of server system.

Item	Description
TYPE	Specifies data transfer type with the *Type* parameter.
USER	Specifies user name.
XCUP	Changes the parent directory of the current working directory (not usually used).
XCWD	Changes current directory (not usually used).
XMKD	Creates a directory (not usually used).
XPWD	Prints the current working directory (not usually used).
XRMD	Removes a directory (not usually used).

The remaining FTP requests defined in Internet RFC 959 are recognized, but not implemented. The **MDTM** and **SIZE** requests are not specified by RFC 959, but are scheduled to appear in the next updated FTP RFC.

If a **STAT** request is received during a data transfer and preceded by both a Telnet **IP** signal and **SYNCH** signal, transfer status is returned.

The **ftpd** daemon must be controlled using the System Management Interface Tool (SMIT) or by changing the **/etc/inetd.conf** file. Typing `ftpd` at the command line is not recommended.

Manipulating the ftpd Daemon with the System Resource Controller

The **ftpd** daemon is a subserver of the **inetd** daemon. The **ftpd** daemon is a member of the **tcpip** SRC subsystem group. This daemon is enabled by default in the **/etc/inetd.conf** file and can be manipulated by the following SRC commands:

Item	Description
startsrc	Starts a subsystem, group of subsystems, or a subserver.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
lssrc	Gets the status of a subsystem, group of subsystems, or a subserver.

Flags

Item	Description
-C	Allows the user to specify that the outgoing file sent using the **send_file** command must be cached in the Network Buffer Cache (NBC). This flag cannot be used unless the **-q** flag is specified. This flag is only applicable when a file is being sent out in the binary mode with no protection.
-c	Suppresses the reverse host name lookup.
-d	Sends debugging information about **ftpd** daemon operations to the **syslogd** daemon. If you specify the **-d** flag, you must edit the **/etc/syslog.conf** file and add the following entry: `daemon.debug FileName` **Note:** The **syslogd** daemon's `debug` level includes `info` level messages. If you do not edit the **/etc/syslog.conf** file, no messages are produced. After changing the **/etc/syslog.conf** file, run the **refresh -s syslogd** command or **kill -1** *SyslogdPID* command to inform the **syslogd** daemon of the changes to its configuration file. For more information about debug levels, refer to the **/etc/syslog.conf** file.

Item	Description
-D *DataConnTimeOut*	Specifies the maximum number of seconds that the **ftpd** daemon holds a data connection. The default value is 300 seconds and a value of 0 specifies an indefinite wait. The value for the *DataConnTimeOut* parameter can range from 0 to *MAXINT*.
-e	Enables only TLS enabled clients to establish connection with the server.
-f	Disables checking for a privileged port when the client requests the server to connect back to a specific port. By default, **ftpd** does not allow the client to request a connection to a privileged port as a security precaution.
-ff	Disables checking for both a privileged port and an IP address that matches the one used for the control connection when the client requests the server to connect back to a specific client port. Using this flag enables the client to request that the server send data to an alternate host or interface. By default, **ftpd** does not allow this action as a security precaution.
-H	Turns on audit logging for the `FILE_Rename`, `FS_Rmdir`, and `FILE_Unlink` events if these events are enabled for the root user.
-k	Sets the **SO_KEEPALIVE** option defined in the **sys/socket.h** file on the data transfer socket to enable the data transfer to time out in the event TCP/IP hangs. The idle interval time is based on system-wide values designated by the `tcp_keepidle` and `tcp_keepintvl` options of the **no** command. Without the flag, **ftpd** data transfer will not time out.
-l	Sends logging information about **ftpd** daemon operations to the **syslogd** daemon. If you specify the **-l** flag, you must edit the **/etc/syslog.conf** file and add the following entry: `daemon.info FileName` If you do not edit the **/etc/syslog.conf** file, no messages are produced. After changing the **/etc/syslog.conf** file, run the **refresh -s syslogd** command or **kill -1** *SyslogdPID* command to inform the **syslogd** daemon of the changes to its configuration file. For more information about debug levels, refer to the **/etc/syslog.conf** file.
-q	Allows the user to specify that the **send_file** subroutine must be used for sending the file on the network. This flag is only applicable when a file is being sent out in the binary mode with no protection.
-t *TimeOut*	Logs out inactive sessions after the number of seconds specified by the *TimeOut* variable. The default limit is 15 minutes (900 seconds). The timeout applies to both the data and the control connections.
-T *MaxTimeOut*	Logs out inactive client sessions after a maximum number of seconds specified by the *MaxTimeOut* variable. The default limit is 2 hours (7200 seconds).
-s	Turns on socket-level debugging.
-u *OctalVal*	Sets the **ftpd** daemon's umask. The *OctalVal* variable must be specified as an octal value to define the umask. The default umask is an octal value of 027, which results in file permissions of rw-r---.
-U	Keep files unlocked while in transfer. If this flag is specified with **/usr/sbin/ftpd**, then the file can be opened while still in transfer.

Security

The **ftpd** daemon is a PAM-enabled application with a service name of *ftp*. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the **usw** stanza of **/etc/security/login.cfg**, to PAM_AUTH as the root user.

The authentication mechanisms used when PAM is enabled depend on the configuration for the **ftp** service in **/etc/pam.conf**. The **ftpd** daemon requires **/etc/pam.conf** entries for the **auth**, **account**, and **session** module types. Listed below is a recommended configuration in **/etc/pam.conf** for the **ftp** service:

```
#
# AIX ftp configuration
#
ftp auth      required     /usr/lib/security/pam_aix

ftp account   required     /usr/lib/security/pam_aix

ftp session   required     /usr/lib/security/pam_aix
```

Examples

Note: The arguments for the **ftpd** daemon can be specified by using SMIT or by editing the **/etc/inetd.conf** file.

1. To start the **ftpd** daemon, type the following:

   ```
   startsrc -t ftp
   ```

   The **startsrc** command with the **-t** flag starts the **ftpd** subserver. You must use the **-t** flag to specify a subserver. Otherwise, the command does not execute properly.
2. To stop the **ftpd** daemon, usually type the following:

   ```
   stopsrc -t ftp
   ```

   The **stopsrc** command with the **-t** flag stops the **ftpd** subserver. The **stopsrc** command allows all pending connections to start and all existing connections to complete, but prevents new connections from starting. You must use the **-t** flag to specify a subserver. Otherwise, the command does not execute properly.
3. To force the **ftpd** daemon and all **ftpd** connections to stop, type the following:

   ```
   stopsrc -f -t ftp
   ```

   The **stopsrc** command with the **-t** and **-f** flags forces the **ftpd** subserver to stop. It terminates all pending connections and existing connections immediately.
4. To display a short status report about the **ftpd** daemon, type the following:

   ```
   lssrc -t ftp
   ```

   The **lssrc** command with the **-t** flag returns the daemon's name, process ID, and state (active or inactive). You must use the **-t** flag to specify a subserver. Otherwise, the command does not execute properly.

Files

Item	Description
/etc/locks/ftpd	Contains interlock and process ID (PID) storage.
/etc/group	Contains passwords for groups.
/etc/passwd	Contains passwords for users.
/etc/security/login.cfg	Contains configuration information for login and user authentication.
/etc/security/passwd	Contains encrypted passwords.
/etc/syslog.conf	Contains configuration information for the **syslogd** daemon.
/usr/samples/tcpip/anon.ftp	Contains the example shell script with which to set up an anonymous FTP account. This file also contains directions for its use.

Item	Description
/etc/ftpd.cnf	Contains the configuration parameters for TLS support.

fuser Command

Purpose

Identifies processes using a file or file structure.

Syntax

fuser [[**-c** | **-C** | **-f**] [**-x**] |**-d**] [**-k** | -K { *SignalNumber* | *SignalName* }] [**-u**] [**-V**]*File* ...

Description

The **fuser** command lists the process numbers of local processes that use the local or remote files specified by the *File* parameter. For block special devices, the command lists the processes that use any file on that device.

Each process number is followed by a letter indicating how the process uses the file:

Item	Description
c	Uses the file as the current directory.
e	Uses the file as a program's executable object.
r	Uses the file as the root directory.
s	Uses the file as a shared library (or other loadable object).

The process numbers are written to standard output in a line with spaces between process numbers. A new line character is written to standard error after the last output for each file operand. All other output is written to standard error.

The **fuser** command will not detect processes that have mmap regions where that associated file descriptor has since been closed. Also, processes using FIFOs (named pipes) will not be detected until the FIFO is fully opened. For example, a process waiting for an open system call to complete will not be seen by the **fuser** command.

The **`fuser`** command is used to determine the processes that are using a file system. If the file system is a network file system (NFS) and the NFS server is not responding, the **`fuser`** command might hang. To avoid such a situation, you can set the `FUSER_VERSION` environment variable to 1.

Flags

Item	Description
-c	Reports on any open files in the file system containing *File*.
-C	Reports on any open files in the file system that is mounted at the directory specified by the **`File`** parameter. If the **`File`** parameter is not a mount point, the command reports an error.
-d	Reports on any open files which have been unlinked (deleted) from the file system containing *File*. When used in conjunction with the **-V** flag, it also reports the inode number and size of the deleted file.
-f	Reports on open instances of *File* only.

Item	Description
-K *SignalNumber* | *SignalName*	Sends the specified signal to each local process. Only the root user can kill a process of another user. Signal can be specified as either a *SignalName*, such as KILL for the SIGKILL signal or a *SignalNumber*, such as 9. Valid values for *SignalName* are those which are displayed by the `kill -l` command.
-k	Sends the **SIGKILL** signal to each local process. Only the root user can kill a process of another user. **Note:** `fuser -k` or **-K** might not be able to detect and kill new processes that are created immediately after the program starts to run.
-u	Provides the login name for local processes in parentheses after the process number.
-V	Provides verbose output.
-x	Used in conjunction with **-c** or **-f**, reports on executable and loadable objects in addition to the standard fuser output.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list the process numbers of local processes using the **/etc/passwd** file, enter:

   ```
   fuser /etc/passwd
   ```

2. To list the process numbers and user login names of processes using the **/etc/filesystems** file, enter:

   ```
   fuser -u /etc/filesystems
   ```

3. To terminate all of the processes using a given file system, enter:

   ```
   fuser -k -x -u -c /dev/hd1
   ```

   or

   ```
   fuser -kxuc /home
   ```

   Either command lists the process number and user name, and then terminates each process that is using the **/dev/hd1 (/home)** file system. Only the root user can terminate processes that belong to another user. You might want to use this command if you are trying to unmount the **/dev/hd1** file system and a process that is accessing the **/dev/hd1** file system prevents this.

4. To list all processes that are using a file which has been deleted from a given file system, enter:

   ```
   fuser -d /usr
   ```

Files

Item	Description
/dev/kmem	Used for the system image.
/dev/mem	Also used for the system image.

fwtmp Command

Purpose

Manipulates connect-time accounting records by reading binary records in **wtmp** format from standard input and converting them to formatted ASCII records. You can use the ASCII version to edit bad records.

Syntax

/usr/sbin/acct/fwtmp [**-i**] [**-c**] [-X] [-L]

Description

The `fwtmp` command manipulates the accounting records by reading binary records in `wtmp` format from standard input and converting them to formatted ASCII records.

Flags

Item	Description
-i	Accepts ASCII records in the **utmp** format as input.
-c	Converts output to **utmp** formatted binary records.
-ic	Converts ASCII **utmp** formatted input records to binary output records.
-X	Prints all available characters of each user name instead of truncating to the first 8 characters.
-L	Prints all available characters of each host name instead of truncating to the first 32 characters.

Security

Access Control: These commands should grant execute (x) access only to members of the **adm** group.

Examples

1. To convert a binary record in **wtmp** format to an ASCII record called `dummy.file`, enter:

   ```
   /usr/sbin/acct/fwtmp < /var/adm/wtmp > dummy.file
   ```

   The content of a binary **wtmp** file is redirected to a dummy ASCII file.
2. To convert an ASCII `dummy.file` to a binary file in **wtmp** format called `/var/adm/wtmp`, enter the `fwtmp` command with the `-ic` switch:

   ```
   /usr/sbin/acct/fwtmp  -ic < dummy.file  > /var/adm/wtmp
   ```

   The dummy ASCII file is redirected to a binary **wtmp** file.

Files

Item	Description
/usr/sbin/acct/fwtmp	Contains the **fwtmp** command.
/var/adm/wtmp	Contains records of date changes that include an old date and a new date.
/usr/include/utmp.h	Contains history records that include a reason, date, and time.

fxfer Command

Purpose

Transfers files between a local system and a host computer connected by HCON.

Syntax

To Restart an Interrupted File Transfer

fxfer **-R** [**-n** *SessionName*]

To Download a File from the Host

fxfer [**-n** *SessionName*] [**-a** | **-r**] [**-d**] [**-c** | **-C**] [**-J**] [**-f** *FileName*] [**-F**] [**-H** *HostType*][**-I** *InputField*] [**-q**] [**-t** [[**-l**] [**-s**] [**-b**]] | **-T** [[**-l**] [**-s**] [**-b**]]]

[**-v**] [**-x** *HostLogin*] [**-e**] [**-X** *CodeSet*] *SourceFile DestFile*

To Upload a File to the Host

fxfer [**-n** *SessionName*] [**-a** | **-r**] [**-u**] [**-c** | **-C**] [**-J**] [**-f** *FileName*] [**-H** *HostType*] [**-q**] [**-t** [[**-l**] [**-s**]] | **-T** [[**-l**] [**-s**]]] [**-l**] [**-s**] [**-v**] [**-x** *HostLogin*] [**-X** *CodeSet*] [**-F** | **-V** | **-U**] [**-B** *BlockSize*] [**-L** *LoglRecLength*] [**-I** *InputField*] [**-S** *NumberUnits* [**,***IncreaseUnits* | **,***IncreaseUnits***,***UnitType* | **,,***UnitType*]] [**-M** *Volume*] [**-N** *Unit*] [**-k**] *SourceFile DestFile*

To Display the Help Screen

fxfer **-h**

Description

The **fxfer** command transfers files between local system and mainframe hosts connected by the Host Connection Program (HCON). Files may transfer from a local system to the host (uploading) or from the host to a local system (downloading). The **fxfer** command transfers the file named by the *SourceFile* parameter to the file named by the *DestFile* parameter. The transfer occurs over an HCON session requiring a specific session profile or an existing session.

The host operating system may be VM/CMS, MVS/TSO, CICS/VS (for CICS/MVS or CICS/VSE), VSE/ESA, or VSE/SP, with the corresponding version of the 3270 File Transfer Program (**IND$FILE** or its equivalent) installed. The version of the host file transfer program is determined by the File Transfer Program value in the session profile. The **fxfer** command supports transfer of either text or binary data. Files will transfer to or from the host with or without ASCII or EBCDIC translation.

Security mechanisms prevent unauthorized access, the destruction of existing files, or the loss of data. If a non-HCON user issues the **fxfer** command, the command fails. If the **fxfer** command is interrupted before completion, the state of the transfer is saved in a RESTART file.

If the **fxfer** command is issued with the **-h** flag, it displays a help screen. If the command is issued with the **-R** flag, it searches the **$HOME** directory for a restart file. If a restart file exists, the restart menu displays, enabling a restart of the file transfer. If the **-h** and **-R** flags are not specified, the command attempts to perform the specified file transfer.

The **fxfer** command information includes:

- Flags
- Flags for Host File Characteristics
- Examples
- Files

This command requires:

- One or more adapters used to connect to a mainframe host.

- One of the following mainframe operating systems be installed on the host:
  - VM/SP CMS
  - VM/XA CMS
  - MVS/SP TSO/E
  - MVS/XA TSO/E
  - CICS/VS (for CICS/MVS or CICS/VSE)
  - VSE/ESA
- The mainframe Host-Supported File Transfer Program (**IND$FILE** or equivalent) be installed on the mainframe.

Session Profiles for Using the fxfer Command

The **fxfer** command communicates with an HCON session and may require a specific session profile. The session profile defines:

- Communication path to the host
- Host type
- Default file transfer direction (down or up)
- Recovery time
- File transfer wait period

When the **fxfer** command is performing an automatic logon, the profile can also define:

- Host logon ID
- AUTOLOG node ID
- Whether the AUTOLOG trace is on
- AUTOLOG time out value

The user usually specifies a session profile when invoking the **fxfer** command. The exception occurs when the command is run from a subshell of an existing session. In this case, if the user does not specify a session profile, the **fxfer** command uses the existing session. If the appropriate session is not running, the **fxfer** command attempts to invoke a new session.

The **fxfer** command searches for an HCON session as follows:

- When issued without the **-n** *SessionName* flag:
  - If the **fxfer** command is issued from a subshell of an existing session, the command uses the session associated with the subshell (defined by the **$SNAME** environment variable).
  - If *not* issued from a subshell of an emulator session, the **fxfer** command issues an error message and terminates.
- When issued with the **-n** *SessionName* flag, the file transfer performs over the specified session. If the specified session does not exist, the command searches for a session profile for that session. If the specified session profile cannot be found, the **fxfer** command issues an error message and terminates. If the specified profile exists, the **fxfer** command attempts an automatic logon to the host using either the AUTOLOG values defined in the session profile, the values defined with the **-x** flag, or by prompting the user for the necessary logon information.

Interrupted and Restarted File Transfers

The **fxfer** command can be interrupted by the operator or an unrecoverable communication error, before completion. If interrupted, the command saves the state of the transfer in a RESTART file. The transfer can be restarted from the beginning without loss of data.

If you run a new file transfer after an interrupted transfer, the **fxfer** command signals that a RESTART file has been created and displays these choices:

- Restart the interrupted file transfer.

- Save the RESTART file and exit the file transfer program.
- Delete the RESTART file and exit the file transfer program.
- Delete the RESTART file and continue the present transfer.

The **fxfer** command with the **-R** flag also restarts an interrupted file transfer.

If the host communication is lost or disconnected during a file transfer started with an automatic logon, the file transfer attempts to recover by reconnecting and logging back on to the host. The recovery time for this attempt is determined by the File Transfer Recovery Time value in the session profile. Once the host connection is re-established, the file transfer resumes from the start. If communication cannot be re-established, the file transfer program generates a RESTART file.

When an explicit file transfer loses communication with the host, the user must restart the emulator session and log back in to the host before attempting to restart the file transfer.

Source and Destination Files

The **fxfer** command *SourceFile* and *DestFile* parameters are required. The *SourceFile* parameter specifies the source file for a file transfer. The *DestFile* parameter specifies the destination file for a file transfer. The local system file names are in the normal format. The host file names conform to the host naming convention, which is one of the following formats:

Host Type	File Name Format
VM/CMS	**"***FileName FileType FileMode***"** **Note:** The " " (double quotation marks) are required for all VM/CMS file names to ensure proper file transfer.
MVS/TSO	**"**[**'**]*DataSetName* [**(***MemberName***)**] [**/***Password*][**'**]**"** where: ***DataSetName*** Indicates either a physical sequential data set or a partitioned data set. **(*MemberName*)** Indicates the name of one of the members in the directory of an existing partitioned data set. The **()** (parentheses) enclosing the *MemberName* are required. **/*Password*** Required if password protection is specified for the MVS/TSO data set. The **/** (slash) preceding the *Password* is required. **Notes:** 1. The **" "** (double quotation marks) are required for all MVS/TSO file names to ensure proper file transfer. 2. When specifying a complete path name for MVS/TSO file names, use **'** (single quotation marks) within the **"** (double quotation marks). Do not put spaces between the double and single quotation marks or between the quotation marks and the file names.
CICS/VS	**"***FileName***"**
VSE/ESA	**"***FileName FileType***"** **Notes:** 1. The **" "** (double quotation marks) are required for all CICS/VS, VSE/ESA, and VSE/SP file names to ensure proper file transfer. 2. CICS/VS, VSE/ESA, and VSE/SP file name conventions allow for a file name up to 8 characters long. 3. In a DBCS environment, HCON does not support a VSE host.

Flags

Note: For Double-Byte Character Set (DBCS) support that includes either Japanese-English, Japanese Katakana, Korean, or Traditional Chinese, these considerations apply:

- If the DBCS **-l** or **-s** flag is specified, one of the translate flags (**-t**, **-T**, or **-J**) must also be specified or the DBCS flags are ignored.
- The **-M**, **-N**, and **-k** flags are used only with MVS/TSO hosts.
- The **-e** flag is valid only with the CICS® program for downloading.
- The **-b** flag is valid only for downloading.

Item	Description
-a	Appends the file designated by *SourceFile* to the file designated by *DestFile*, if the destination file exists. This flag is ignored and the destination file is created if the file designated by *DestFile* does not exist. **Note:** The **-a** flag is not valid when uploading a file to a CICS/VS host. For VSE/ESA, the **-a** flag is valid only for uploading to CICS temporary storage (FILE=TS).
-b	Retains the blanks at the end of each record when used with the **-t**, **-T**, **-c**, or **-C** flags. The **-b** flag is only supported in the DBCS environment.
-c	In a DBCS environment, the **-c** flag changes LF (line-feed) code of a file to CRLF (carriage return line-feed) code if the file transfer is an upload. For a downloading file transfer, the **-c** flag changes the CRLF code of a file to LF code.
-C	In a DBCS environment, the **-C** flag inhibits the sending of the EOF (end-of-file) code of a PC-DOS file if the file transfer is an upload. For a downloading file transfer, the **-C** flag appends an EOF code: x ' 1A at the end of a PC-DOS file.
-d	Downloads the file by transferring it from the host to the local system. If neither this flag nor the **-u** flag is specified, the File Transfer Direction characteristic in the session profile determines the direction of the transfer. **Note:** When downloading a translated file from a VSE/ESA host file transfer (FILE=HTF) the file is deleted from the host system unless you specify the **-I** "KEEP" flag.
-e	Deletes the temporary storage queue at the completion of the file transfer. Use this flag only with the CICS host for downloading. The **-e** flag is only supported in the DBCS environment.
-f *FileName*	Places the file transfer process diagnostic output (or file transfer status) in the file specified by the *FileName* variable. If the **-f** flag is not specified for an asynchronous transfer, messages are placed in the **$HOME/hconerrors** file. If the **-f** flag is not specified for a synchronous transfer, messages are sent to standard output. Messages due to errors in specifying file transfer parameters or file names, or failures in the file transfer process, are directed to standard output (if it is a local system screen) or to the **$HOME/hconerrors** file (if standard output is not a local system screen).

Item	Description
-h	Displays a help screen for the **fxfer** command. This screen summarizes each available command flag and command operation. When this flag is specified all other flags are ignored and no files are transferred. **Notes:** 1. If the **-h** flag is used, all other flags are ignored. No files transfer. 2. If the **fxfer** command is not initiated from a subshell of an existing HCON session, either the **-h** flag or the **-n** flag is required.
-H *HostType*	Specifies the type of host. The *HostType* variable may have any of these values: **CMS** VM/SP CMS or VM/XA CMS **TSO** MVS/SP TSO or MVS/XA TSO **CICS** CICS/VS (The CICS host type includes CICS/VSE, CICS/MVS, CICS/ESA, and CICS/MVS/ESA.) **VSE** VSE/ESA (Not supported in a DBCS environment.) If the **-H** flag is omitted, the value specified by the Host Type characteristic in the session profile is used. The user must specify the correct host operating system. **Notes:** 1. If you specified the **CICS** or **VSE** value and the system returns an error, retry the command with the alternate value. The CICS and VSE **IND$FILE** programs are functionally interchangeable; however, there is a 6-byte header-size discrepancy that makes the versions operationally incompatible. The destination host may be using the alternate version of the program. 2. To transfer files to an MVS/TSO host, you may need to leave session manager mode before initiating the file transfer.
-I *InputField*	Specifies host file transfer options placed directly within the **IND$FILE** command. Also allows comments within the **IND$FILE** command placed after a) (right parentheses). The value specified by the *InputField* variable is placed in quotation marks, as follows: `-I "FILE=TS) This is a comment"` **Note:** The **-I** field is not supported in a DBCS environment.
-J	Allows data conversion between EBCDIC and ASCII, and normalization of SI/SO characters. The translation depends on the direction of the transfer: **Upload** Translates 1-byte characters of a file to EBCDIC code. For DBCS countries, the extended code is translated to the appropriate DBCS code. SO/SI characters are inserted into DBCS fields containing DBCS characters. If the file contains control codes 0x1E or 0x1F, they are replaced with SO and SI characters respectively. **Download** Translates EBCDIC code to 1-byte characters of a file; For DBCD, the DBCS code is translated to extended code. Deletes SO/SI characters from DBCS fields. **Note:** The **-J** field is only supported in a DBCS environment.

Item	Description
-k	Releases unused records in the data set at the completion of file transfer. Use this flag only in the MVS/TSO environment. The **-k** flag is only supported in the DBCS environment.
-l	Specifies the host language in the DBCS environment. This option must be used with one of the translate flags (**-t**, **-T**, or **-J**). If **-t**, **-T**, or **-J** is omitted, the **-l** flag is ignored. If the **-l** flag is not specified, the host language defined in the session profile is used. If the **-l** flag is specified, the host language used is the alternate language of the language defined in the session profile. For example, if the Language characteristic in the session profile is JPK (Japanese Katakana), the host language used for file transfer will be Japanese-English. The **-l** flag is only supported in the DBCS environment.
-M *Volume*	Specifies the volume serial number of the host disk for data set allocation. Use this flag only in the MVS/TSO environment. The **-M** flag is only supported in the DBCS environment.
-n *SessionName*	Specifies the name of a previously defined session whose characteristics control the file transfer. The session name is a single character in the range of a to z. Capital letters are interpreted as lowercase letters. The **-n** *SessionName* flag is required except when the user is initiating the **fxfer** command from a subshell of an existing session. In this case, if the **-n** flag is not used the **fxfer** command defaults to the existing session. **Notes:** 1. The specified session must have been previously defined by using the **smit hcon** fast path command or the **mkhcons** command. 2. If the **fxfer** command is not initiated from a subshell of an existing HCON session, either the **-h** flag or the **-n** flag is required.
-N *Unit*	Specifies the unit type of the host disk for data set allocation. Use this flag only in the MVS/TSO environment. The **-N** flag is only supported in the DBCS environment.
-q	Runs the file transfer asynchronously as a background process. If any file transfers are not completed, the current transfer request is queued. If the **-q** flag is not specified, the file transfer operation is synchronous. If the **-f** flag is not specified, diagnostic output and status is placed in the **$HOME/hconerrors** file. **Note:** The system limits the number of bytes allowed in one Interprocess Communication (IPC) message queue. As a result, the maximum number of file transfers that can be queued at any one time is approximately 580.

Item	Description
-r	Specifies replacement of an existing file on the host (upload) or an existing file on the local system (download). On downloads, the replacement is done only when the transfer is successful. This ensures the existing file is not lost or destroyed if the transfer does not complete for any reason. If the **-r** flag is specified and the file does not exist, it is created during the file transfer. If the **-r** flag is *not* specified and the destination file exists, an error message is produced. For uploading, the **-r** flag must be specified when using a version of the host file transfer program below PTF UR20455 for MVS/TSO or PTF UR90118 for VM/CMS. For VSE and CICS the **-r** flag is ignored. **Note:** The host file transfer program usually defaults to replace for a file. If it does not, add `-I "replace"` to the **fxfer** command to specify replace. **Attention:** When replacing a file on the host, you must specify a logical record length (**-L** flag) and a record format (**-F** or **-V** flag) equal to the logical record length and record format of the existing file. If you do not do this, data corruption may result. This does not apply to VSE/ESA.
-R	Restarts a previous file transfer (which was interrupted by the user or an unsuccessful recovery attempt) using the information saved in one of the RESTART files: the **$HOME/x_fxfer.r** file or the **$HOME/i_fxfer.r** file. If the file transfer is not invoked from the subshell of an existing session, the **-n** *SessionName* flag must be included to specify the session to be used. If the **-R** flag is specified in conjunction with any other file transfer flags, those flags are ignored and the RESTART file transfer menu is displayed. **Note:** With the **-R** flag, all other flags except the **-n** *SessionName* flag are ignored. The RESTART file transfer menu displays.
-s	Specifies the SO/SI handling in the DBCS environment. The **-s** flag must be used with one of the translate flags (**-t**, **-T**, or **-J**). If **-t**, **-T**, or **-J** is omitted, the **-s** flag is ignored. When the **-s** flag is specified, the following functions are performed for file transfer: **Upload** SO/SI characters are not inserted in DBCS fields. **Download** SO/SI characters are replaced with control characters (0x1E/0x1F) in DBCS fields. The **-s** flag is only supported in the DBCS environment.
-t	Performs ASCII-EBCDIC translation for a file. If downloading, the **fxfer** command translates EBCDIC to ASCII. If uploading, the **fxfer** command translates ASCII to EBCDIC. The language is specified by the Language characteristic in the session profile. The **-t** flag assumes the file is a text file. The new-line character is the line delimiter. When the **-t** flag is used in a DBCS environment with other DBCS supported flags, the behavior of the **-t** flag changes as follows: **Upload** Translates JISCII (Japan) or ASCII (Korean, Traditional Chinese) to EBCDIC. Inserts SO/SI characters in DBCS fields. **Download** Translates EBCDIC to JISCII (Japan) or ASCII (Korean, Traditional Chinese). Deletes SO/SI characters from DBCS fields.

Item	Description
-T	Performs ASCII-EBCDIC translation for a disk operating system file. The character sequence, CRLF, used as the line delimiter, and a disk operating system EOF (end-of-file) character are inserted at the end of the downloaded file. The language to be used for EBCDIC to ASCII translation is specified by the Language characteristic in the session profile. The **-T** flag is used to translate disk operating system files. **Note:** If neither the **-T**, **-t**, nor the **-J** flag is specified, the file transfer assumes no translation and transfers the information in binary form.
-u	Uploads the file by transferring the file from the local system to the host. If neither this flag nor the **-d** flag is specified, the File Transfer Direction characteristic in the session profile determines the direction of the transfer.
-v	Periodically writes the current status of the file transfer to the screen or to the status file specified by the **-f** flag. The status includes the number of bytes transferred and the elapsed time since the file transfer process began transferring data.
-x *HostLogin*	Uses the login ID specified by the *HostLogin* variable to log in to the host. The user is prompted to enter the password. The *HostLogin* string consists of the host login ID, the AUTOLOG node ID, and other optional AUTOLOG values. The string cannot contain any blanks and must contain the AUTOLOG node ID. Format the AUTOLOG string as: `UserID,AutologNodeID[,Trace,Time . . .]` If the **-x** flag is not specified, the information for the *HostLogin* string is taken from the session profile as follows: • If the host login ID is set in the session profile, you are prompted for the password. The remaining parameters are retrieved from the profile. • If the host login ID is not set in the profile, you are prompted for both the host login string and the password. • Your response to a prompt always overrides a profile parameter. For example, if the AUTOLOG time is set in the profile but you enter a different value at the prompt, the value entered at the prompt is used. If you omit certain parameters from the host login string, they are retrieved from the profile, if defined there. For example, if the you set the AUTOLOG Node ID, AUTOLOG Trace, and AUTOLOG Time parameters in the profile, only the host login ID must be entered at the prompt. The file transfer process logs in to the host and establishes an emulation session using the session profile specified with the **-n** flag. Once the process is successfully logged in, the file transfer begins. The File Transfer Wait Period parameter in the session profile determines how long the login session is maintained. Using this parameter, the host login session is maintained for subsequent file transfers. The need to log in again is eliminated.

Item	Description
-X *CodeSet*	Specifies an alternate code set to use for ASCII-EBCDIC translation. If the **-X** flag is omitted, the code set specified by the system locale is used. The following code sets are supported: **Default** Uses current system ASCII code page. **IBM-932** Uses IBM code page 932 for translation in a DBCS environment. **ISO8859-1** Uses ISO 8859-1 Latin alphabet number 1 code page. **ISO8859-7** Uses ISO 8859-7 Greek alphabet. **ISO8859-9** Uses ISO 8859-9 Turkish alphabet. **IBM-eucJP** Uses IBM Extended UNIX Code for translation in the Japanese Language environment. **IBM-eucKR** Uses IBM Extended UNIX Code for translation in Korean Language environment. **IBM-eucTW** UsesIBM Extended UNIX Code for translation in Traditional Chinese Language environment.

Flags for Host File Characteristics

The following flags specify host file characteristics and can be used only to upload files (with the exception of the **-F** flag, which can be used when downloading from a VSE host):

Item	Description
-B *BlockSize*	Specifies the block size of the host data set. The **-B** flag can only be used in the MVS/TSO environment and only for sequential data sets. The *BlockSize* variable cannot exceed the capacity of a single track. The **-B** flag is ignored if the file is being appended. A block size value of 0 causes an error.
-F	Specifies fixed-length records. This is the default if neither the **-V**, **-t**, **-T**, **-c**, nor **-C** flag is specified. The **-F** flag is ignored if the file is being appended. On a CICS or VSE host, one of the translate flags (**-t** or **-T**) or one of the CRLF flags (**-c** or **-C**) must be specified along with the **-F** flag, since the CICS and VSE host file transfer programs do not support fixed record lengths. The combination of the **-F** flag and the translate flag causes the transfer program to pad the records with blanks to the end of the logical record length. The default is 80. **Note:** Use the **-F** flag when downloading from a VSE host to prevent the deletion of trailing blanks from the translated file.

Item	Description
-L *LoglRecLength*	Specifies the logical record length in bytes of the host file. For new files, the default is 80. For variable-length records, *LoglRecLength* is the maximum size of the record. The **-L** flag is ignored if the file is being appended. A *LoglRecLength* value of 0 causes an error. Because of MVS™ overhead, the actual number of bytes stored in the variable length records on an MVS/TSO host is four bytes less than the value specified by the *LoglRecLength* variable. The CICS and VSE host file transfer programs do not support logical record lengths. For transfers to or from a CICS or VSE host the **-L** flag must be accompanied by the **-F** flag. The combination of the **-F** and **-L** flags causes the transfer program to pad the records with blanks to the end of the logical record length. The default is 80. **Note:** The **-L** flag is required if a record length is greater than the default record length of 80.
-S *NumberUnits* [**,***IncreaseUnits* | **,***IncreaseUnits*,*UnitType* | **,,***UnitType*]	Specifies the amount of space to be allocated for a new sequential data set on TSO. For large MVS files, the maximum block size permissible on the host is used to ensure that the whole disk track is filled. The **-S** flag can be used only with MVS/TSO hosts. The following variables can be used with the **-S** flag. If used, they must be specified in the order given and separated by commas. If a variable preceding another variable is omitted, a comma must be included as a placeholder. A space is required between the **-S** flag and the *NumberUnits* variable. However, no spaces can appear in the variable string. ***NumberUnits*** Specifies the number of units of space to be added initially. A value of 0 or a negative value cannot be specified for the *NumberUnits* variable. ***IncreaseUnits*** Specifies the number of units of space to be added to the data set each time the previously allocated space is filled (optional). ***UnitType*** Defines the unit of space and may be T for tracks, C for cylinders, or a number specifying the average block size (in bytes) of the records written to the data set. If the *UnitType* variable is not specified, the default is the value specified by the **-B** flag. If the **-B** *BlockSize* flag is not specified, the default value is 80. Following are the possible combinations of variables used with the **-S** flag: `-S NumberUnits,IncreaseUnits,UnitType` `-S NumberUnits,IncreaseUnits` `-S NumberUnits` `-S NumberUnits,,UnitType`
-U	Specifies records of undefined length. The **-U** flag can only be used in the MVS/TSO environment. The **-U** flag is ignored if the file is being appended.
-V	Specifies records of variable length. This is the default if the **-F** flag is not specified, and either the **-t**, **-T**, **-c**, or **-C** flag is specified. The **-V** flag is ignored if the file is being appended. The **-V** flag is not supported by the CICS or VSE host file transfer programs, since variable record lengths are the default.

Examples

The following examples assume the session profile for session a is:

```
Session   type         DFT
Communication   device         3270c0
Language         English   (U.S.A.)
Host   type         CMS
File   transfer   direction         up
File   transfer   wait   period         10
File   transfer   recovery   time         30
```

where:

- The host type is VM/CMS.
- The connection is made using the DFT 3270 connection device.
- The file transfer default direction is upload (to use session profile a for downloading files, the user must specify the **-d** flag with the **fxfer** command).
- The file transfer process stays logged in for 10 minutes.
- If a transfer is interrupted, the process attempts recovery for 30 minutes before saving information in the RESTART file for later transfer.
- The translation language is U.S.A. ASCII-EBCDIC.

1. To upload the `samplefile` file (in the current directory) to the host and translate it to EBCDIC using the U.S.A. translation table, enter:

   ```
   fxfer   -n   a   -t   samplefile   "test   file   a"
   ```

   – **-n** instructs the **fxfer** command to use session a to transfer the file.
   – **-t** instructs the command to translate using the new-line character.

   The translated data is placed in the `test file a` on the host. Because the host file name contains spaces, quotation marks around the file name are required.

2. To upload the `file2` file to the VM/CMS host `test file b`, enter:

   ```
   fxfer   -urv   -L   132   -V   -H   CMS   file2   "test   file   b"
   ```

   – **-u** instructs the **fxfer** command to upload the file.
   – **-H** indicates that the host type is a VM/CMS host. If the destination file exists, it is replaced (since the **-r** flag is specified) by the transferred file.
   – **-v** causes **fxfer** to display the number of bytes transferred and elapsed time. The status or diagnostic output is displayed on the terminal.
   – If the host file does not exist, the host file maximum logical record length is set to 132 bytes (**-L** flag).
   – The host file record format is variable (**-V** flag). No translation is performed.

3. To upload, from a subshell of emulator session a, the local system **/etc/motd** file to the CICS motdfile host file with translation and padding of blanks, enter:

   ```
   fxfer   -utFH   CICS   -I   ")This   is   a   comment"   /etc/motd   "motdfile"
   ```

   – **-u** instructs the command to upload the file.
   – **-t** causes translation from ASCII to EBCDIC.
   – **-F** causes the transfer program to pad the uploaded file with blanks to column 80 (the default record length). To change the default column, use the **-L** flag with a different record length (column).
   – **-H** specifies the host as type CICS.
   – **-I** specifies that the *InputField* value be added to the **IND$FILE** command.

In this example, "This is a comment" is a host comment field.

To upload or download files with the **fxfer** command, to or from a TSO environment other than your current environment, you must have authorization for the other environment. You must completely qualify the file (or data set) within single quotes ('), then double quotes (" ").

4. For example, to upload the file newfile to a TSO environment where the complete qualified name is sys4.parmlib.samplefile, enter:

```
fxfer   -urtvH   TSO   'newfile'   "sys4.parmlib.samplefile"
```

   - **-u** instructs the command to upload the file.
   - If the sys4.parmlib.samplefile file exists, it is replaced (**-r** flag) with the translated contents of the newfile file (**-t** flag).
   - **-v** instructs the **fxfer** command to write the file transfer status to the local screen every few seconds.
   - **-H** instructs the **fxfer** command that the host is a MVS/TSO host.

     Note: This example assumes that the **fxfer** command is issued from a subshell of an established session (use the **e789** command to establish a session).

5. To download the file spfuser.test from the MVS/TSO host to the local system, enter:

```
fxfer   -n   a   -d   -r   -H   TSO   spfuser.test   samplefile1
```

   - **-n** instructs the **fxfer** command to use session a to transfer the file. If session a has not already been established, the command attempts an automatic login. Since no host login ID is specified, the **fxfer** command checks the session profile for a login ID. If one is not specified there, the user is prompted for the login ID and password.
   - **-d** overrides the default file transfer direction of upload.
   - If the samplefile1 file already exists, it is replaced (**-r** flag) with the downloaded file from the host.
   - **-H** instructs the **fxfer** command that the host is an MVS/TSO host instead of VM/CMS (the default from the session profile).

   The transferred file is placed in the samplefile1 file on the local system. The file transfer is performed synchronously.

6. To download the VM/CMS host test file a and append it to the local system mydir/samplefile file, using session profile a and automatic login, enter:

```
fxfer   -n   a   -dat   -q   -f   status.out
-x   laura,vm1,trace   "test   file   a"   mydir/samplefile
```

   - **-n** instructs the **fxfer** command to use session profile a to transfer the file.
   - **-x** provides the host login ID. The **fxfer** command first checks to see if session is established on the local system. If so, the command transfers the file over the existing session. If session a is not established, the **fxfer** command performs an automatic login using the host logon ID laura and the AUTOLOG script vm1, and traces the login activity. The user is prompted for the password. The command transfers the file.
   - **-dat** instruct the **fxfer** command to download the file (**-d** flag), translate the data from EBCDIC to ASCII (**-t** flag) using the U.S.A. translation table (defined in the session profile), and append (**-a** flag) the translated file to the mydir/samplefile file on the local system. If the mydir/samplefile file does not already exist, the **fxfer** command ignores the **-a** flag and creates the file.
   - The status or diagnostic output is placed in the status.out file in the current local directory (**-f** flag).
   - **-q** instructs the **fxfer** command to transfer the file asynchronously.

When the user enters the password, the prompt is returned and the file transfer is performed in the background.

To queue another file transfer to be performed by the same file transfer process, enter:

```
fxfer   -n   a   -daq   -f   status.out   "test   file   b"
mydir/samplefile
```

- **-n** instructs the **fxfer** command to use session a to transfer the file. Since session a has been established by the previous command, the **fxfer** command does not need to log in to the host again.
- **-d** instructs the command to download a file from the host.
- **-a** instructs the command to append the `test file b` host file to the `mydir/samplefile` file on the local system.
- **-q** instructs the **fxfer** command to transfer the file asynchronously.

The **fxfer** command continues to send status information to the `status.out` file on the local system (**-f** flag).

Notes:

a. If the text for the **fxfer** command extends beyond the limit of the screen, the text wraps automatically to the next line. Pressing the Enter key to wrap the text causes an error.

b. Attempting to start a synchronous file transfer when there is an asynchronous transfer in the queue causes an error.

c. The user will not be prompted for a login ID or a password as long as the session remains running and the **dfxfer** process remains logged in to the host. The amount of time the process remains logged in is determined by the File Transfer Wait Period in the session profile.

7. To restart an interrupted file transfer from an emulator subshell, enter:

```
fxfer   -R
```

-R instructs the **fxfer** command to use the information saved in one of the RESTART files to execute a file transfer. The RESTART file is the **$HOME/x_fxfer.r** explicit restart file or **$HOME/i_fxfer.r** implicit restart file. If the **-R** flag is specified in conjunction with other file transfer flags, the other flags are ignored. The RESTART file transfer menu is displayed. Using this menu, instruct the **fxfer** command to transfer the interrupted file.

8. To restart the file transfer from the command line instead of from an emulator subshell, enter:

```
fxfer   -R   -n   a
```

The **-n** flag instructs the **fxfer** command to use session a to perform the restarted transfer.

Files

Item	Description
/usr/bin/fxfer	Contains the **fxfer** command.
/usr/bin/dfxfer	Contains the **dfxfer** process.
$HOME/i_fxfer.r	Contains RESTART information for automatic login queues. Temporary file created by the **fxfer** command.
$HOME/x_fxfer.r	Contains RESTART information for manual login queues. Temporary file created by the **fxfer** command.
$HOME/hconerrors	Contains HCON diagnostic output and file transfer status. Temporary file created by any HCON command.
/usr/lib/libfxfer.a	Contains the library for programmatic file transfers.

g

The following AIX commands begin with the letter g.

gated Daemon

Purpose

Provides gateway routing functions for the RIP, RIPng, EGP, BGP, BGP4+, HELLO, IS-IS, ICMP, ICMPv6, and SNMP protocols.

Note: Use SRC commands to control the **gated** daemon from the command line. Use the **rc.tcpip** file to start the daemon with each system startup.

Syntax

/usr/sbin/gated [**-c**] [**-C**] [**-n**] [**-N**] [**-t** *TraceOptions*] [**-f** *ConfigFile*] [*TraceFile*]

Description

The **/usr/sbin/gated** daemon handles multiple routing protocols and replaces **routed** and any routing daemon that speaks the (HELLO) routing protocol. The **/usr/sbin/gated** daemon currently handles the Routing Information Protocol (RIP), Routing Information Protocol Next Generation (RIPng), Exterior Gateway Protocol (EGP), Border Gateway Protocol (BGP) and BGP4+, Defense Communications Network Local-Network Protocol (HELLO), and Open Shortest Path First (OSPF), Intermediate System to Intermediate System (IS-IS), and Internet Control Message Protocol (ICMP)/Router Discovery routing protocols. In addition, the **gated** daemon supports the Simple Network Management Protocol (SNMP). The **gated** process can be configured to perform all of these protocols or any combination of them. The default configuration file for the **gated** daemon is the **/etc/gated.conf** file. The **gated** daemon stores its process ID in the **/etc/gated.pid** file.

Note: Unpredictable results may occur when the **gated** and **routed** daemons are run together on the same host.

If on the command line a trace file is specified, or no trace flags are specified, the **gated** daemon detaches from the terminal and runs in the background. If trace flags are specified without specifying a trace file, **gated** assumes that tracing is desired to **stderr** and remains in the foreground.

Note: IS-IS routing protocol cannot be run on 64-bit kernel.

Signals

The **gated** server performs the following actions when you use the **kill** command to send it signals.

Item	Description
SIGHUP	Re-read configuration. A **SIGHUP** causes gated to reread the configuration file. The **gated** daemon first performs a clean-up of all allocated policy structures. All BGP and EGP peers are flagged for deletion and the configuration file is reparsed. If the reparse is successful, any BGP and EGP peers that are no longer in the configuration are shut down, and new peers are started. The **gated** daemon attempts to determine if changes to existing peers require a shutdown and restart. **Note:** Reconfiguration is disable when OSPF (Open Shortest Path First) is enabled.

Item	Description
SIGINT	Snapshot of current state. The current state of all gated tasks, timers, protocols and tables are written to **/var/tmp/gated_dump**. This is done by forking a subprocess to dump the table information so as not to impact the **gated** daemon's routing functions.
SIGTERM	Graceful shutdown. Upon receiving a **SIGTERM** signal, the **gated** daemon attempts a graceful shutdown. All tasks and protocols are asked to shutdown. Most will terminate immediately, the exception being EGP peers which wait for confirmation. It may be necessary to repeat the **SIGTERM** once or twice if this process takes too long. All protocol routes are removed from the kernel's routing table on receipt of a **SIGTERM**. Interface routes, routes with **RTF_STATIC** set (from the **route** command where supported) and static routes specifying **retain** will remain. To terminate the **gated** daemon with the exterior routes intact, use the **SIGKILL** or **SIGQUIT** signals (which causes a core dump).
SIGUSR1	Toggle tracing. Upon receiving a **SIGUSR1** signal, the **gated** daemon will close the trace file. A subsequent **SIGUSR1** will cause it to be reopened. This will allow the file to be moved regularly. **Note:** It is not possible to use the **SIGUSR1** signal if a trace file has not been specified, or tracing is being performed to **stderr**.
SIGUSR2	Check for interface changes. Upon receiving a **SIGUSR2** signal, the **gated** daemon rescans the kernel interface list looking for changes.

The gated and snmpd Daemons

The **gated** daemon is internally configured to be an SNMP multiplexing (SMUX) protocol peer, or proxy agent, of the **snmpd** daemon. For more information, refer to "SNMP daemon processing" in *Networks and communication management*.

Manipulating the gated Daemon with the System Resource Controller

The **gated** daemon can be controlled by the System Resource Controller (SRC). The **gated** daemon is a member of the SRC **tcpip** system group. This daemon is disabled by default and can be manipulated by the following SRC commands:

Item	Description
startsrc	Starts a subsystem, group of subsystems, or a subserver.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
refresh	Causes the subsystem or group of subsystems to reread the appropriate configuration file.
lssrc	Gets the status of a subsystem, group of subsystems, or a subserver.

Note: On initial startup from the **startsrc** command, the **gated** daemon does not start responding to other SRC commands until all **gated** initialization is completed. A very large **/etc/gated.conf** file can require a minute or more to parse completely.

Flags

Item	Description
-c	Specifies parsing of the configuration file for syntax errors after which the **gated** daemon exits. If no errors occur, the **gated** daemon puts a dump file into the **/var/tmp/gated_dump** file. The **-c** flag implies the **-tgeneral,kernel,nostamp** flag. If the **-c** flag is specified, the **gated** daemon ignores all **traceoption** and **tracefile** clauses in the configuration file.
-C	Specifies that the configuration file is parsed only for syntax errors. The **gated** daemon exists with a status of **1** if it finds any errors and with a status of **0** if it does not. The **-C** flag implies the **-tnostamp** flag.
-f *ConfigFile*	Specifies an alternate configuration file. By default, the **gated** daemon uses the **/etc/gated.conf** file.
-n	Specifies that the **gated** daemon will not modify the kernel's routing table. This is used for testing **gated** configurations with actual routing data.
-N	Specifies that the **gated** daemon does not daemonize. Normally, if tracing to **stderr** is not specified and the parent process ID is not 1, the **gated** daemon daemonizes. This flag allows the use of a method similar to **/etc/inittab** of invoking the **gated** daemon that does not have a process ID of 1.
-t*TraceOptions*	Specifies which trace options are enabled at system startup. When used without the *TraceOptions* variable, this flag starts the **general** trace options. Separate each trace option from another with a comma. Do not insert a space between the flag and the first trace option. The **-t** flag must be used to trace events that take place before the **/etc/gated.conf** file is parsed, such as determining the interface configuration and reading routes from the kernel. The **gated.conf** file article describes the available trace options.

Examples

1. To start the **gated** daemon, enter a command similar to the following:

   ```
   startsrc -s gated -a "-tall /var/tmp/gated.log"
   ```

   This command starts the **gated** daemon and logs messages. Messages are sent to the **/var/tmp/gated.log** file.
2. To stop the **gated** daemon normally, enter:

   ```
   stopsrc -s gated
   ```

   This command stops the daemon. The **-s** flag specifies that the subsystem that follows is to be stopped.
3. To get short status from the **gated** daemon, enter:

   ```
   lssrc -s gated
   ```

   This command returns the name of the daemon, the process ID of the daemon, and the state of the daemon (active or inactive).

Files

Item	Description
/etc/gated.pid	Contains the **gated** process ID.

Item	Description
/var/tmp/gated_dump	Specifies the memory dump file.
/var/tmp/gated.log	Specifies the log file for error messages.

gdc Command

Purpose

Provides an operational user interface for **gated**.

Syntax

gdc [**-q**] [**-n**] [**-c** *coresize*] [**-f** *filesize*] [**-m** *datasize*] [**-s** *stacksize*] [**-t** *seconds*] *Subcommands*

Description

The **gdc** command provides a user-oriented interface for the operation of the **gated** routing daemon. It provides support for:

- starting and stopping the daemon
- the delivery of signals to manipulate the daemon when it is operating
- the maintenance and syntax checking of configuration files
- for the production and removal of state dumps and core dumps.

The **gdc** command can reliably determine **gated**'s running state and produces a reliable exit status when errors occur, making it advantageous for use in shell scripts which manipulate **gated**. Commands executed using **gdc** and, optionally, error messages produced by the execution of those commands, are logged via the same **syslogd** facility which **gated** itself uses, providing an audit trail of operations performed on the daemon.

Flags

Item	Description
-n	Runs without changing the kernel forwarding table. This is useful for testing, and when operating as a route server which does no forwarding.
-q	Runs quietly. With this flag informational messages which are normally printed to the standard output are suppressed and error messages are logged with **syslogd** instead of being printed to the standard error output. This is convenient when running **gdc** from a shell script.
-t *seconds*	Specifies the time in seconds that **gdc** waits for **gated** to complete certain operations, in particular at termination and startup. By default this value is set to 10 seconds.
-c *coresize*	Sets the maximum size of a core dump a **gated** started with **gdc** produces. This is useful on systems where the default maximum core dump size is too small for **gated** to produce a full core dump on errors.
-f *filesize*	Sets the maximum file size a **gated** started with **gdc** will produce. Useful on systems where the default maximum file dump size is too small for **gated** to produce a full state dump when requested.
-m *datasize*	Sets the maximum size of the data segment of a **gated** started with **gdc**. Useful on systems where the default data segment size is too small for **gated** to run.
-s *stacksize*	Sets the maximum size of stack of a **gated** started with **gdc**. Useful on systems where the default maximum stack size is too small for **gated** to run.

Subcommands

The following subcommands cause signals to be delivered to **gated** for various purpose:

Item	Description
COREDUMP	Sends an abort signal to **gated**, causing it to terminate with a core dump.
dump	Signals **gated** to dump its current state into the file **/var/tmp/gated_dump**.
interface	Signals **gated** to recheck the interface configuration. **gated** normally does this periodically in any event, but the facility can be used to force the daemon to check interface status immediately when changes are known to have occurred.
KILL	Causes **gated** to terminate ungracefully.
reconfig	Signals **gated** to reread its configuration file, reconfiguring its current state as appropriate.
term	Signals **gated** to terminate after shutting down all operating routing protocols gracefully. Executing this command a second time causes **gated** to terminate even if some protocols have not yet fully shut down.
toggletrace	Causes tracing to be suspended, and if **gated** is currently tracing to a file, closes the trace file. If **gated** tracing is current suspended, this subcommand causes the trace file to be reopened and tracing initiated. This is useful for moving trace files.

The following subcommands perform operations related to configuration files:

Item	Description
checkconf	Check **/etc/gated.conf** for syntax errors. This is usefully done after changes to the configuration file but before sending a **reconfig** signal to the currently running **gated**, to ensure that there are no errors in the configuration which would cause the running **gated** to terminate on reconfiguration. When this command is used, **gdc** issues an informational message indicating whether there were parse errors or not, and if so saves the error output in a file for inspection.
checknew	Like **checkconf** except that the **new** configuration file, **/etc/gated.conf+**, is checked instead.
newconf	Move the **/etc/gated.conf+** file into place as **/etc/gated.conf**, retaining the older versions of the file as described above. **gdc** will decline to do anything when given this command if the **new** configuration file doesn't exist or otherwise looks suspect.
backout	Rotate the configuration files in the **newer** direction, in effect moving the **old** configuration file to **/etc/gated.conf**. The command will decline to perform the operation if **/etc/gated.conf-** doesn't exist or is zero length, or if the operation would delete an existing, non-zero length **/etc/gated.conf+** file.
BACKOUT	Perform a **backout** operation even if **/etc/gated.conf+** exists and is of non-zero length.
modeconf	Set all configuration files to mode 664, owner root, group system.
createconf	If **/etc/gated.conf+** does not exist, create a zero length file with the file mode set to 664, owner root, group system.

The following subcommands provide support for starting and stopping **gated**, and for determining its running state:

Item	Description
running	Determine if **gated** is currently running. This is done by checking to see if **gated** has a lock on the file containing its pid, if the pid in the file is sensible and if there is a running process with that pid. Exits with zero status if **gated** is running, non-zero otherwise.
start	Start **gated**. The command returns an error if **gated** is already running. Otherwise it executes the **gated** binary and waits for up to the delay interval (10 seconds by default, as set with the **-t** option otherwise) until the newly started process obtains a lock on the pid file. A non-zero exit status is returned if an error is detected while executing the binary, or if a lock is not obtained on the pid file within the specified wait time.
stop	Stop **gated**, gracefully if possible, ungracefully if not. The command returns an error (with non-zero exit status) if **gated** is not currently running. Otherwise it sends a terminate signal to **gated** and waits for up to the delay interval (10 seconds by default, as specified with the **-t** option otherwise) for the process to exit. Should **gated** fail to exit within the delay interval it is then signaled again with a second terminate signal. Should it fail to exit by the end of the second delay interval it is signaled for a third time with a kill signal. This should force immediate termination unless something is very broken. The command terminates with zero exit status when it detects that **gated** has terminated, non-zero otherwise.
restart	If **gated** is running it is terminated via the same procedure as is used for the **stop** command above. When the previous **gated** terminates, or if it was not running prior to command execution, a new **gated** process is executed using the procedures described for the **start** command above. A non-zero exit status is returned if any step in this procedure appears to have failed.

The following subcommands allow the removal of files created by the execution of some of the commands above:

Item	Description
rmcore	Removes any existing **gated** core dump file.
rmdump	Removes any existing **gated** state dump file.
rmparse	Removes the parse error file generated when a **checkconf** or **checknew** command is executed and syntax errors are encountered in the configuration file being checked.

By default **gated** obtains its configuration from a file normally named **/etc/gated.conf**. The **gdc** program also maintains several other versions of the configuration file, in particular named:

Item	Description
/etc/gated.conf+	The new configuration file. When **gdc** is requested to install a new configuration file, this file is renamed **/etc/gated.conf**.
/etc/gated.conf-	The old configuration file. When **gdc** is requested to install a new configuration file, the previous **/etc/gated.conf** is renamed to this name.
/etc/gated.conf—	The really old configuration file. **gdc** retains the previous **old** configuration file under this name.

Files

Item	Description
/usr/sbin/gated	The **gated** binary.

Item	Description
/etc/gated.conf	Current **gated** configuration file.
/etc/gated.conf+	Newer configuration file.
/etc/gated.conf-	Older configuration file
/etc/gated.conf—	Much older configuration file
/etc/gated.pid	Where **gated** stores its pid.
/var/tmp/gated_dump	**gated**'s state dump file
/var/tmp/gated.log	Where config file parse errors go.

gencat Command

Purpose

Creates and modifies a message catalog.

Syntax

gencat *CatalogFile SourceFile ...*

Description

The **gencat** command creates a message catalog file (usually ***.cat**) from message text source files (usually ***.msg**). The **gencat** command merges the message text source files, specified by the *SourceFile* parameter, into a formatted message catalog, specified by the *CatalogFile* parameter. After entering messages into a source file, use the **gencat** command to process the source file to create a message catalog. The **gencat** command creates a catalog file if one does not already exist. If the catalog file does exist, the **gencat** command includes the new messages in the catalog file.

You can specify any number of message text source files. The **gencat** command processes multiple source files, one after another, in the sequence specified. Each successive source file modifies the catalog. If the set and message numbers collide, the new message text defined in the *SourceFile* parameter replaces the old message text currently contained in the *CatalogFile* parameter. Message numbers must be in the range of 1 through **NL_MSGMAX**. The set number must be in the range of 1 through **NL_SETMAX**.

The **gencat** command does not accept symbolic message identifiers. You must run the **mkcatdefs** command if you want to use symbolic message identifiers.

Note: Standard output is used if the **-** (dash) character is specified as the *CatalogFile* parameter. Standard input is used if the **-** (dash) character is specified as the *SourceFile* parameter.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

To generate a `test.cat` catalog from the source file `test.msg`, enter:

```
gencat test.cat test.msg
```

The `test.msg` file does not contain symbolic identifiers.

Files

Item	Description
/usr/bin/gencat	Contains the **gencat** command.

gencopy Command

Purpose

Allows software products of various packaging formats (installp, RPM, ISMP) to be copied.

Syntax

To Copy Software from Media to Target Location

gencopy **-d** *Media* [**-t** *TargetLocation*] [**-D**] [**-b** *bffcreateFlags*] [**-U**] [**-X**] **-f** *File* | *CopyList...* | all

To List Software Products and Packages on Media

gencopy **-L** **-d** *Media* [**-D**]

Description

The **gencopy** command is the wrapper to the **bffcreate** command. It determines what images must be copied and calls the appropriate command. For RPM, ISMP, or other types of images where the list of required files is unknown, all the files in the subdirectory are copied to the target location.

Flags

Item	Description
-b *bffcreateFlags*	Specifies the following flags that are valid: **l**, **q**, **v**, **w**, and **S**.
-d *Media*	Specifies the device or directory where the install images exist. Media can be a device (**/dev/cd0**, **/dev/rmt0**) or directory.
-D	Specifies debug mode. This flag is for debugging this script. It produces a large quantity of output and should not be used for normal operations.
-f *File*	Specifies a file that contains a list of images to copy to the target location. The `installp`, RPM, and ISMP images should be prefixed with `I:`, `R:`, and `J:`, respectively. Prefix the interim fix packages with an `E:`.
-L	Lists the install packages on the media. This listing is colon separated and contains the following information: file_name:package_name:fileset:V.R.M.F:type:platform:Description bos.sysmgt:bos.sysmgt:bos.sysmgt.nim.client:4.3.4.0:I:R:Network Install Manager - Client Tools bos.sysmgt:bos.sysmgt:bos.sysmgt.smit:4.3.4.0:I:R:System Management Interface Tool (SMIT)
-t *TargetLocation*	Specifies the directory where the installation image files are stored. If the **-t** flag is not specified, the files are saved in the **/usr/sys/inst.images** directory.

Item	Description
-U	Upgrades the directory structure of the destination repository to the current standard, if necessary. The current standard requires images to be organized into subdirectories according to package type and architecture. For example, installp images reside in the ***SaveDir*/installp/ppc** directory. When copying from a source containing this structure, the destination is required to conform. Specifying the **-U** flag permits the **gencopy** command to create the appropriate subdirectory structure in your repository and move any existing images into the appropriate locations. Unless invalid manual copying is performed thereafter, this flag should only need to be used once.
-X	Extends the file system automatically if space is needed.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

To copy all of the image from a CD (**/dev/cd0**) to an **LPP_SOURCE** (**/export/lpp_source/500**) use, type:

```
gencopy -d  /dev/cd0 -t  /export/lpp_source/500 all
```

Files

Item	Description
/usr/sbin/gencopy	
/usr/sys/inst.data/sys_bundles	
/usr/sys/inst.data/user_bundles	

gencore Command

Purpose

Generates a core file for a running process.

Syntax

gencore *ProcessID FileName*

Description

The **gencore** command creates a core file of the process specified by the process ID *ProcessID* without terminating the process. The created core file contains the memory image of the process, which can be used with the **dbx** command for debugging purposes. The core file generated will be named as specified by *FileName* parameter.

The `gencore` command does not create the core file in the location set by the `chcore` or `syscorepath` commands. The core file is placed in the path specified by the *FileName* parameter. If *FileName* specifies only the name of the file, the core file is placed in the current working directory.

Parameters

Item	Description
FileName	Specifies the file name of the core file the **gencore** command creates.
ProcessID	Specifies the process ID of the process from which **gencore** will create a core file.

Exit Status

0
The core file was created successfully.

>0
An Error occurred. A partial core file may be created.

Examples

1. To generate a core file named "core.1095" for the process with process ID 1095, enter:

```
gencore 1095 core.1095
```

   The creates the core file without terminating the process.

Files

Item	Description
/usr/bin/gencore	Contains the **gencore** command.

genfilt Command

Purpose

Adds a filter rule.

Syntax

genfilt -v 4|**6** [**-n** *fid*] [**-a** D|P|I|L|E|H|S] **-s** *s_addr* **-m** *s_mask* **[-d** *d_addr*] [**-M** *d_mask*] [**-g Y**|**N**] [**-c** *protocol*] [**-o** *s_opr*] [**-p** *s_port*] [**-O** *d_opr*] [**-P** *d_port*] [**-r R**|**L**|**B**] [**-w I**|**O**|**B**] [**-l Y**|**N**] [**-f Y**|**N**|**O**|**H**] [**-t** *tid*] [**-i** *interface*] [**-D** *description*] [-e *expiration_time*] [-x *quoted_pattern*] [-X *pattern_filename*] [-C *antivirus_filename*]

Description

Use the **genfilt** command to add a filter rule to the filter rule table. The filter rules generated by this command are called manual filter rules. IPsec filter rules can be configured by using the **genfilt** command or the IPsec smit (IP version 4 or IP version 6).

Flags

Item	Description
-a *Action*	The following *Action* values are allowed: • D (Deny) blocks traffic. • P (Permit) allows traffic. • I makes this an IF filter rule. • L makes this an ELSE filter rule. • E makes this an ENDIF filter rule. • H makes this a SHUN_HOST filter rule. • S makes this a SHUN_PORT filter rule. All IF rules most be close with an associated ENDIF rule. These conditional rules can be nested, but correct nesting and scope must be adhered to or the rules will not load correctly with the `mkfilt` command.
-C *antivirus_filename*	Specifies the antivirus file name. The -C flag understands some versions of ClamAV Virus Database (http://www.clamav.net).
-c *protocol*	The valid values are: **udp**, **icmp**, **icmpv6**, **tcp**, **tcp/ack**, **ospf**, **ipip**, **esp**, **ah**, and **all**. Value **all** indicates that the filter rule will apply to all the protocols. The protocol can also be specified numerically (between 1 and 252). The default value is **all**. Value **tcp/ack** implies checking for TCP packets with the ACK flag set.
-D *description*	A short description text for the filter rule. This is an optional flag for static filter rules, it's not applicable to dynamic filter rules.
-d *d_addr*	Specifies the destination address. It can be an IP address or a host name. If a host name is specified, the first IP address returned by the name server for that host will be used. This value along with the destination subnet mask will be compared against the destination address of the IP packets.
-e *expiration_time*	Specifies the expiration time. The expiration time is the amount of time the rule should remain active in seconds. The *expiration_time* does not remove the filter rule from the database. The *expiration_time* relates to the amount of time the filter rule is active while processing network traffic. If no *expiration_time* is specified, then the live time of the filter rule is infinite. If the *expiration_time* is specified in conjunction with a SHUN_PORT (-a S) or SHUN_HOST (-a H) filter rule, then this is the amount of time the remote port or remote host is denied or shunned once the filter rule parameters are met. If this *expiration_time* is specified independent of a shun rule, then this is the amount of time the filter rule will remain active once the filter rules are loaded into the kernel and start processing network traffic.
-f	Specifies the fragmentation control. This flag specifies that this rule will apply to either all packets (**Y**), fragment headers and unfragmented packets only (**H**), fragments and fragment headers only (**O**), or unfragmented packets only (**N**). The default value is **Y**.
-g	Apply to source routing? Must be specified as **Y** (yes) or **N** (No). If **Y** is specified, this filter rule can apply to IP packets that use source routing. The default value is yes (**Y**). This field only applies to permit rules.
-i *interface*	Specifies the name of IP interface(s) to which the filter rule applies. The examples of the name are: **all**, **tr0**, **en0**, **lo0**, and **pp0**. The default value is **all**.

Item	Description
-l	Specifies the log control. Must be specified as **Y**(yes) or **N** (No). If specified as **Y**, packets that match this filter rule will be included in the filter log. The default value is **N** (no).
-M	Specifies the destination subnet mask. This is used in the comparison of the IP packet's destination address with the destination address of the filter rule.
-m	Specifies the source subnet mask. This is used in the comparison of the IP packet's source address with the source address of the filter rule.
-n	Specifies the filter rule ID. The new rule will be added BEFORE the filter rule you specify. For IP version 4, the ID must be greater than 1 because the first filter rule is a system generated rule and cannot be moved. If this flag is not used, the new rule will be added to the end of the filter rule table.
-O	Specifies the destination port or ICMP code operation. This is the operation that will be used in the comparison between the destination port/ICMP code of the packet with the destination port or ICMP code (**-P** flag). The valid values are: **lt**, **le**, **gt**, **ge**, **eq**, **neq**, and **any**. The default value is **any**. This value must be **any** when the **-c** flag is **ospf**.
-o	Specifies the source port or ICMP type operation. This is the operation that will be used in the comparison between the source port/ICMP type of the packet with the source port or ICMP type(**-p** flag) specified in this filter rule. The valid values are: **lt**, **le**, **gt**, **ge**, **eq**, **neq**, and **any**. The default value is **any**. This value must be **any** when the **-c** flag is **ospf**.
-p	Specifies the source port or ICMP type. This is the value/type that will be compared to the source port (or ICMP type) of the IP packet.
-P	Specifies the destination port/ICMP code. This is the value/code that will be compared to the destination port (or ICMP code) of the IP packet.
-r	Routing. This specifies whether the rule will apply to forwarded packets (**R**), packets destined or originated from the local host (**L**), or both (**B**). The default value is **B**.
-s *s_addr*	Specifies the source address. It can be an IP address or a host name. If a host name is specified, the first IP address returned by the name server for that host will be used. This value along with the source subnet mask will be compared against the source address of the IP packets.
-t	Specifies the ID of the tunnel related to this filter rule. All the packets that match this filter rule must go through the specified tunnel. If this flag is not specified, this rule will only apply to non-tunnel traffic.
-v	Specifies the IP version of the filter rule. Valid values are **4** and **6**.
-w *Direction*	Specifies whether the rule applys to incoming packets (I), outgoing packets (O), or both (B). The default value is B. It is not valid to use the (O) outgoing direction with the `-x`, `-X`, or `-C` pattern options. It is valid to specify the (B) both directions with the pattern options, but only the incoming packets are checked against the packets.
`-X` *pattern_filename*	Specifies the pattern file name. If more than one patterns are associated with this filter rule, then a pattern file name must be used. The pattern file name must be in the format of one pattern per line. A pattern is an unquoted character string. This file is read once when the filter rules are activated. For more information, see the `mkfilt` command.

Item	Description
-x *pattern*	Specifies the quoted character string or pattern. This string specified is interpreted as an ASCII string unless it is preceded by a `0x`, in which case it is interpreted as a hexadecimal string. The -x *pattern* is compared against network traffic.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

geninstall Command

Purpose

A generic installer that installs software products of various packaging formats. For example, installp, RPM, SI, and ISMP.

Syntax

geninstall -d *Media* [**-I** *installpFlags*] [**-E** | **-T**] [**-t** *ResponseFileLocation*] [**-e** *LogFile*] [**-p**] [**-F**] [**-Y**] [**-Z**] [-D] { **-f** *File* | *Install_List*] | all}

OR

geninstall -k [**-d** *Media*] [**-p**] [**-Y**] [**-f** *File* | *install_list* | `all` | `update_all`]

OR

geninstall -u [**-e** *LogFile*] [**-E** | **-T**] [**-t** *ResponseFileLocation*] [-D] {**-f** *File* | *Uninstall_List*...}

OR

geninstall -L -d *Media* [**-e** *LogFile*] [-D]

Description

Accepts all current **installp** flags and passes them on to **installp**. Some flags (for example, **-L**) are overloaded to mean list all products on the media. Flags that don't make sense for ISMP packaged products are ignored. This allows programs (like NIM) to continue to always send in **installp** flags to **geninstall**, but only the flags that make sense are used.

The `geninstall` command provides an easy way to see what modifications have been made to the configuration files listed in `/etc/check_config.files`. When these files have been changed during a `geninstall` installation or update operation, the differences between the old and new files will be recorded in the `/var/adm/ras/config.diff`. If `/etc/check_config.files` requests that the old file be saved, the old file can be found in the `/var/adm/config` directory.

The `/etc/check_config.files` file can be edited and can be used to specify whether old configuration files that have been changed should be saved (indicated by s) or deleted (indicated by d), and has the following format:

```
d /etc/inittab
```

A summary of the **geninstall** command's install activity is kept at **/var/adm/sw/geninstall.summary**. This file contains colon-separated lists of filesets installed by **installp** and components installed by ISMP. This is used mainly to provide summary information for silent installs.

Note: Refer to the **README.ISMP** file in the **/usr/lpp/bos** directory to learn more about ISMP-packaged installations and using response files. Also, the **geninstall** command is capable of installing interim fix files containing concurrent updates. Any interim fixes containing concurrent updates must be placed inside a subdirectory called **cupdates** in the directory containing installation images, and the **geninstall** command will install them appropriately.

Flags

Item	Description
-d *Device or Directory*	Specifies the device or directory that contains the images to install. The **geninstall** command searches for the images in the following paths: • /*mount_point*/installp/ppc (installp package) • /*mount_point*/RPMS/ppc (RPM package) • /*mount_point*/emgr/ppc (Interim fix packages for AIX) • /*mount_point*/ISMP/ppc (ISMP packages for AIX) If the paths do not exist, the **geninstall** command searches for the images in the base directory of the specific device. If the image names are not preceded by a prefix that indicates the image type, the **geninstall** command identifies a type for the image. If the paths exist and the image is not found in any path, and if the image does not contain any prefix, the image is treated as an RPM-formatted image.
-D	Specifies debug mode. This flag is for debugging this script. It produces a large quantity of output and should not be used for normal operations.
-e *LogFile*	Enables event logging. The **-e** flag enables the user to append certain parts of the **geninstall** command output to the file specified by the *LogFile* variable. The *LogFile* variable must specify an existing, writable file, and the file system in which the file resides must have enough space to store the log. The log file does not wrap.
-E	Creates an ISMP response file recording in the default location, which is the directory containing the product installation files. This option requires running the ISMP installation or uninstallation interactively and completely. The resulting response file will be used to provide the same options on future installations or uninstallations of the same product. Creation of the response file recording will also result in installation or uninstallation of the product.
-f *File*	Specifies a file that contains a list of images to copy to the target location. The installp, RPM, and ISMP images should be prefixed with I:, R:, and J:, respectively. Prefix the interim fix packages with an E:.
-F	Allows the user to reinstall a package that is already installed, or to install a package that is older than the currently installed version.
-I *installpFlags*	Specifies the **installp** flags to use when calling the **installp** command. The flags that are used during an install operation for **installp** are the **a**, **b**, **c**, **D**, **e**, **E**, **F**, **g**, **I**, **J**, **M**, **N**, **O**, **p**, **Q**, **q**, **S**, **t**, **v**, **V**, **w**, and **X** flags. The **installp** flags that are not used during install are the **C**, **i**, **r** , **z**, **A**, and **l** flags. The **installp** command should be called directly to perform these functions. The **-u**, **-d**, **-L**, and **-f** flags should be given outside the **-I** flag.

Item	Description
-k	You can use this flag in IBM AIX 7.2 Technology Level 1, or later, to use the AIX Live Update operation to upgrade Service Packs (SPs), Technology levels (TLs), group or individual updates, and interim fixes that are marked as LU CAPABLE. To determine if an interim fix is LU CAPABLE, you can perform a preview installation. Before you begin the Live Update operation, you must commit any existing updates on the system. The Live Update operation always commits all updates that are applied during the process, expands any necessary file systems, and installs any requisite software. All other installation processes are prevented from starting during the Live Update operation. **Note:** If your Live Update operation includes the installation of updates, you must take a viable system backup before proceeding with the Live Update operation. The Live Update operation will not create a backup image. The Live Update operation fails if you are performing the following tasks in your environment: • Installing the interim fix in a special environment such as a workload partition (wpar), multibos, or alternate disk environment. • Installing an operating system. • The logical volume names for the required operating system logical volumes (/, /var, /opt, /usr, /etc) and the boot logical volume are not used as the default logical volume names. The Live Update operation requires additional input that is specified in the /var/adm/ras/liveupdate/lvupdate.data file. For more information about this file, see the /var/adm/ras/liveupdate/lvupdate.template. If the updates are successfully applied and committed and if the Live Update process fails, you can rerun the Live Update process by running the **geninstall** command with only the **-k** flag. In this scenario, you do not need to include the device and software in the command. In AIX 7200-01, or later, you can apply and commit any updates or interim fix by using SMIT or any method that you prefer and perform the Live Update operation without using the **-d** flag so that you do not have to restart the system. The **geninstall -k** operation does not update RPM packages. RPM packages should be updated before performing an LKU operation. The all option installs all the software that is available in the device or directory specified by the **-d** flag. This process installs all the software in the software source specified by the **-d** flag even if the software was not previously installed on the system. The update_all option installs only a higher version of the specified software that is already installed on the system. The update_all option installs new software only if the newer versions of the existing software have requisites to the new software. **Note:** The bos.liveupdate.rte fileset must be installed to perform the Live Update operation.
-L	Lists the contents of the media. The output format is the same as the **installp -Lc** format, with additional fields at the end for ISMP and RPM formatted products.

Item	Description
-p	Performs a preview of an action by running all preinstallation checks for the specified action.
-t *ResponseFileLocation*	Allows specifying an alternate location for response files or response file templates. The default location is the directory containing the product installation files. This flag can be used to create a response file recording or template in a different location. The *ResponseFileLocation* can either be a file or directory name. If the *ResponseFileLocation* is a directory, it must already exist. If the *ResponseFileLocation* is not an existing directory, it will be assumed that a file name is specified.
-T	Creates an ISMP response file template in the default location, which is the directory containing the product installation files. The resulting template can be used to create a response file for future installations or uninstallations of the same product with the desired options. Creation of the response file template will not result in installation or uninstallation of the product.
-u	Performs an uninstall of the specified software. For ISMP products, the uninstaller listed in the vendor database is called, prefixed by a "J:".
-Y	Agrees to required software license agreements for software to be installed. This flag is also accepted as an **installp** flag with the **-I** option.
-Z	Tells **geninstall** to invoke the installation in silent mode.

Examples

1. To install all the products on a CD media that is in the drive `cd0`, type:

   ```
   geninstall -d /dev/cd0 all
   ```

   If ISMP images are present on the media, a graphical interface is presented. Any **installp**, SI, or RPM images are installed without prompting, unless the **installp** images are spread out over multiple CDs.
2. To install an interim fix, named `IV12345.160101.epkg.Z`, that is located in the `/images/emgr/ppc` directory, enter the following command:

   ```
   geninstall -d /images IV12345.160101.epkg.Z
   ```

   Note: If the `/images/emgr/ppc` directory exists, but the package is in the `/images` directory (`/images/IV12345.160101.epkg.Z`), the **geninstall** command does not consider the package as an interim fix, and tries to install it as an RPM-formatted image. Fore more information, see the **-d** flag.

Files

- `/usr/sbin/gencopy`
- `/usr/sys/inst.data/sys_bundles`
- `/usr/sys/inst.data/user_bundles`

genkex Command

Purpose

The **genkex** command extracts the list of kernel extensions currently loaded onto the system and displays the address, size, and path name for each kernel extension in the list.

Syntax

genkex [-**dh**]

Description

For kernel extensions loaded onto the system, the kernel maintains a linked list consisting of data structures called loader entries. A loader entry contains the name of the extension, its starting address, and its size. This information is gathered and reported by the **genkex** command.

Flags

Item	Description
-d	Shows the address and size of the Data section, in addition to the address and size of the Text section.
-h	Displays usage statement.

Examples

To generate the list of loaded kernel extensions, enter:

```
genkex
```

genkld Command

Purpose

The **genkld** command extracts the list of shared objects currently loaded onto the system and displays the address, size, and path name for each object on the list.

Syntax

genkld [-**dh**]

Description

For shared objects loaded onto the system, the kernel maintains a linked list consisting of data structures called loader entries. A loader entry contains the name of the object, its starting address, and its size. This information is gathered and reported by the **genkld** command.

Flags

Item	Description
-d	Shows the address and size of the Data section, in addition to the address and size of the Text section.
-h	Displays usage statement.

Examples

To obtain a list of loaded shared objects, enter:

```
genkld
```

genld Command

Purpose

The **genld** command collects the list of all processes currently running on the system, and optionally reports the list of loaded objects corresponding to each process.

Syntax

genld [**-h** | **-l** [**-d**]] [-a *Area*] [**-u**]

Description

For each process currently running, the **genld** command prints a report consisting of the process ID and name, optionally followed by the list of objects loaded for that process. The object's address and path name are displayed. Members of libraries are shown between brackets. For example, `/usr/lib/libc.a[shr.o]` means `shr.o` is a loaded member of the **libc.a** library.

You can filter the output of the **genld** command by using the **-u** flag to display processes that have old versions of loaded objects. An object is considered as an old object if the object image is different from the image that is currently installed on the file system. The **-u** flag is used after applying an update to list the processes that require a restart operation to use the new binaries and libraries.

Notes:

- Unprivileged users can see loaded objects only for their processes.
- If the full path name to a loaded object cannot be determined, the **genld** command might not report updates to this object if it is located on a file system other than journaled file system (JFS2). The object might also be reported as `updated` if the object is replaced by an identical copy.

Flags

Item	Description
-a *Area*	Lists only processes using the shared library area specified by the *Area* parameter.
-d	Shows the address and size of the Data section, in addition to the address and size of the Text section. This option has no effect without the **-l** flag.
-h	Displays the usage statement.
-l	Reports the lists of loaded objects for each process running on the system.
-u	Lists only processes that have old versions of loaded objects.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Examples

- To obtain the list of loaded objects for each running process, enter the following command:

```
genld -l
```

- To obtain the list of processes that have old versions of loaded objects, enter the following command:

```
genld -u
```

- To obtain the list of processes that have an old version of the **libcrypt.a** library loaded, enter the following command:

```
genld -lu | grep -p libcrypt.a
```

gennames Command

Purpose

Gathers all the information necessary to run the **filemon** and **netpmon** commands in off-line mode.

Syntax

gennames[**-f**]

Description

The **gennames** command gathers name to address mapping information necessary for the **filemon** and **netpmon** commands to work in off-line mode. The information gathered includes:

- the list of all the loaded kernel extension, similar to what the **genkex** command reports,
- the list of all the loaded shared libraries, similar to what the **genkld** command reports
- the list of all the loaded processes, similar to what the **genld** command reports
- for **/unix** and all kernel extensions and libraries, the output of the **stripnm -z** command is collected

Flags

Item	Description
-f	Collects the device information for physical and logical volumes. It also prints out the virtual file system information used by offline **filemon**.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To collect information needed for the **filemon** command in off-line mode, type:

```
gennames -f > gen.out
```

gensyms Command

Purpose

Gathers all the necessary information to run the **curt**, **splat**, and **tprof** commands in offline mode.

Syntax

gensyms [**-o**] [**-f**] [**-F**] [**-h**] [**-s**] [**-g**] [**I**] [**-N**] [**-k** *kernel*] [**-i** *file*] [**-b** *binary*[,*binary*[,...]]] [**-P** *pid*[,*pid*[,...]]] [**-S** *path*]

Description

The **gensyms** command gathers name to address mapping information necessary for the **curt**, **splat**, and **tprof** commands to work in offline mode. The information that is gathered includes the following items:

- The list of all the loaded kernel extension.
- The list of all the loaded shared libraries.
- The list of all the loaded processes.
- For **/unix**, all kernel extensions, libraries, and all object files corresponding to processes, the output of the **stripnm** command is collected.

Flags

Item	Description
-b *binary*	Specifies an optional list of binaries for which to find symbols.
-f	Suppresses printing of source file names.
-F	Collects the device information for physical and logical volumes.
-g	Decodes symbol names.
-h	Prints help message.
-i *file*	Reads symbols from specified file.
-I	Prints binary instructions of symbols.
-k *kernel*	Specifies the name of the kernel image (default **/unix**).
-N	Prints the source line number of symbols.
-o	Prints offset instead of addresses
-P *pid*[,*pid*[,...]]	Prints the symbols of dependent modules that are loaded by the specified processes. This flag is optional.
-s	Finds symbols only for files specified by the **-k** and **-b** flags.
-S *path*	Specifies the search path list; it is used to find binaries.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To collect information that is required for the **tprof** command in offline mode with the profiling of user program test, enter the following command:

   ```
   gensyms > test.syms
   ```

2. To collect information about the specified process ID and its dependent processes, enter the following command:

   ```
   gensyms -P pid > test.syms
   ```

gentun Command

Purpose

Creates a tunnel definition in the tunnel database.

Syntax

gentun -s *src_host_IP_address* **-d** *dst_host_IP_address* **-v 4**|**6** [**-t** *tun_type*] [**-m** *pkt_mode*] [**-t** *IBM*] [**-t** *manual*] [**-m** *tunnel*] [**-m** *transport*] [**-f** *fw_address*] [**-x** dst_mask]] [**-e** [*src_esp_algo*]] [**-a** [*src_ah_algo*]] [**-p** *src_policy*] [**-A** [*dst_ah_algo]*] [**-P** *dst_policy*] [**-k** *src_esp_key*] [**-h** *src_ah_key*] [**-K** *dst_esp_key*] [**-H** *dst_ah_key*] [**-n** *src_esp_spi*] [**-u** *src_ah_spi*] [**-N** *dst_esp_spi*] [**-U** *dst_ah_spi*] [**-b** *src_enc_mac_algo*] [**-c** *src_enc_mac_key*] [**-B** *dst_enc_mac_algo*] [**-C** *dst_enc_mac_key*] [**-g**] [**-z**] [**-E**]

Description

The **gentun** command creates a definition of a tunnel between a local host and a tunnel partner host. The associated auto-generated filter rules for the tunnel can be optionally generated by this command.

Flags

Item	Description
-a	Authentication algorithm, used by source for IP packet authentication. The valid values for **-a** depend on which authentication algorithms have been installed on the host. The list of all the authentication algorithms can be displayed by issuing the **ipsecstat -A** command. The default value is HMAC_MD5 for **manual** tunnels.
-A	(**manual** tunnel only) Authentication algorithm, used by destination for IP packet authentication. The valid values for **-A** depend on which authentication algorithms have been installed on the host. The list of all the authentication algorithms can be displayed by issuing the **ipsecstat -A** command. If this flag is not used, the value used by the **-a** flag is used.
-b	(**manual** tunnel only) Source ESP Authentication Algorithm (New header format only). The valid values for **-b** depend on which authentication algorithms have been installed on the host. The list of all the authentication algorithms can be displayed by issuing the **ipsecstat -A** command.
-B	(**manual** tunnel only) Destination ESP Authentication Algorithm (New header format only). The valid values for **-B** depend on which authentication algorithms have been installed on the host. The list of all the authentication algorithms can be displayed by issuing the **ipsecstat -A** command. If this flag is not used, it is set to the same value as the **-b** flag.
-c	(**manual** tunnel only) Source ESP Authentication Key (New header format only). It must be a hexdecimal string started with "0x". If this flag is not used, the system will generate one for you.
-C	(**manual** tunnel only) Destination ESP Authentication Key (New header format only). It must be a hexdecimal string started with "0x". If this flag is not used, it is set to the same value as the **-c** flag.
-d	Destination Host IP address. In host-host case, this is the IP address of the destination host interface to be used by the tunnel. In host-firewall-host case, this is the IP address of the destination host behind the firewall. A host name is also valid and the first IP address returned by name server for the host name will be used.
-e	Encryption algorithm, used by source for IP packet encryption. The valid values for **-e** depend on which encryption algorithms have been installed on the host. The list of all the encryption algorithms can be displayed by issuing the **ipsecstat -E** command.

Item	Description
-E	(**manual** tunnel only) Encryption algorithm, used by destination for IP packet encryption. The valid values for **-E** depend on which encryption algorithms have been installed on the host. The list of all the encryption algorithms can be displayed by issuing the **ipsecstat -E** command. If this flag is not used, the value used by the **-e** flag is used.
-f	IP address of the firewall that is between the source and destination hosts. A tunnel will be established between this host and the firewall. Therefore the corresponding tunnel definition must be made on the firewall host. A host name may also be used for this flag and the first IP address returned by the name server for that host name will be used.
-g	System auto-generated filter rule flag. If this flag is not used, the command will generate two filter rules for the tunnel automatically. The auto-generated filter rules will allow IP traffic between the two end points of the tunnel to go through the tunnel. If the **-g** flag is specified, the command will only create the tunnel definition, and the user will have to add user defined filter rules to let the tunnel work.
-h	This is the AH Key String for a **manual** tunnel. The input must be a hexdecimal string started with "0x". If this flag is not used, the system will generate a key using a random number generator.
-H	(**manual** tunnel only) The Key String for destination AH. The input must be a hexdecimal string started with "0x". If this flag is not used, the system will generate a key using a random number generator.
-k	This is the ESP Key String for a **manual** tunnel. It is used by the source to create the tunnel. The input must be a hexdecimal string started with "0x". If this flag is not used, the system will generate a key using a random number generator.
-K	(**manual** tunnel only) The Key String for destination ESP. The input must be a hexdecimal string started with "0x". If this flag is not used, the system will generate a key using a random number generator.
-l	Key Lifetime, specified in minutes. For **manual** tunnels, this value indicates the time of operability before the tunnel expires. The valid values for **manual** tunnels are 0 - 44640. Value 0 indicates that the **manual** tunnel will never expire. The default value for **manual** tunnels is 480.
-m	Secure Packet Mode. This value must be specified as **tunnel** or **transport**. The default value is **tunnel**. Tunnel mode will encapsulate the entire IP packet, while the transport mode only encapsulates the data portion of the IP packet. When generating a host-firewall-host tunnel (for host behind a firewall), the value of **tunnel** must be used for this flag. The -m flag is forced to use default value (**tunnel**) if the **-f** flag is specified.
-n	(**manual** tunnel only) Security Parameter Index for source ESP. This is a numeric value that, along with the destination IP address, identifies which security association to use for packets using ESP. If this flag is not used, the system will generate an SPI for you.
-N	(**manual** tunnel only) Security Parameter Index for the destination ESP. It must be entered for a **manual** tunnel if the policy specified in the **-P** flag includes ESP. This flag does not apply to **IBM** tunnels.
-p	Source policy, identifies how the IP packet authentication and/or encryption is to be used by this host. If specified as **ea**, the IP packet gets encrypted before authentication. If specified as **ae**, it gets encrypted after authentication, whereas specifying **e** alone or **a** alone corresponds to the IP packet being encrypted only or authenticated only. The default value for this flag will depend on if the **-e** and **-a** flags are supplied. The default policy will be **ea** if either both or neither the **-e** and **-a** flags are supplied. Otherwise the policy will reflect which of the **-e** and **-a** flags were supplied.

Item	Description
-P	(**manual** tunnel only) Destination policy, identifies how the IP packet authentication and/or encryption is to be used by destination. If specified as **ea**, the IP packet gets encrypted before authentication. If specified as **ae**, it gets encrypted after authentication, whereas specifying **e** or **a** corresponds to the IP packet being encrypted only or authenticated only. The default policy will be **ea** if either both or neither the **-E** and **-A** flags are supplied. Otherwise, the policy will reflect which of the **-E** and **-A** flags were specified.
-s	Source Host IP address, IP address of the local host interface to be used by the tunnel. A host name is also valid and the first IP address returned by name server for the host name will be used.
-t	Type of the tunnel. Must be specified as **manual**. The initial tunnel key and any subsequent key updates need to be performed manually when using the **manual** tunnel. Once a key is installed manually, that same key is used for all tunnel operations until it is changed manually. The **manual** tunnel value should be selected when you want to construct a tunnel with a non-IBM IP Security host or any IP version 6 end-point, where the end-point either supports RFCs 1825-1829 or the IETF drafts for the new IP Security encapsulation formats for IP tunnels.
-u	(**manual** tunnel only) Security Parameter Index for source AH. Use SPI and the destination IP address to determine which security association to use for AH. If this flag is not used, the value of the -n SPI will be used.
-U	(**manual** tunnel only) Security Parameter Index for the destination AH. If this flag is not used, the -N spi will be used.
-v	The IP version for which the tunnel is created. For IP version 4 tunnels, use the value of **4**. For IP version 6 tunnels, use the value of **6**.
-x	Network mask for the secure network behind a firewall. The Destination host is a member of the secure network. The combination of **-d** and **-x** allows the source host to communicate with multiple hosts in the secure network through the source-firewall tunnel, which must be in tunnel mode. This flag is valid only when the **-f** flag is used.
-y	(**manual** tunnel only) Replay prevention flag. Replay prevention is valid only when the ESP or AH header is using the new header format (see the **-z** flag). The valid values for the **-y** flag are Y (yes) and N (no). All encapsulations that are used in this tunnel (AH, ESP, sending, and receiving) will use the replay field if the value of this flag is Y. The default value is N.
-z	(**manual** tunnel only) New header format flag. The new header format preserves a field in the ESP and AH headers for replay prevention and also allows ESP authentication. The replay field will only be used when the replay flag (**-y**) is set to Y. The valid values for the **-z** flag are Y (yes) and N (no). The default value when the **-z** flag is not used depends on the algorithms you've chosen for the tunnel. It will default to N unless either an algorithm other than KEYED_MD5 is used for either the **-a** or **-A** flags, or if the **-b** or **-B** flags are used.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

genxlt Command

Purpose

Generates a code set conversion table for use by the **lconv** library.

Syntax

genxlt [OutputFile]

Description

The **genxlt** command reads a source code set conversion table file from standard input and writes the compiled version to the file specified by the *OutputFile* parameter. If a value is not specified for the *OutputFile* parameter, standard output is used. The source code set conversion table file contains directives that are acted upon by the **genxlt** command to produce the compiled version.

The format of a code set conversion table source file is:

- Lines whose initial nonwhite space character is the # (pound sign) are treated as comment lines.
- Null lines and lines consisting only of white-space characters are treated as comment lines.
- Non-comment lines have to be of the following form:

```
%token <blank>   # <tab> and <space>
%token <hex>     # <zero>, <one>, <two>, <three>, <four>,
                 # <five>, <six>, <seven>, <eight>, <nine>,
                 # <a>, <b>, <c>, <d>, <e>, <f>,
                 #  <A>, <B>, <C>, <D>, <E>, <F>,
%token <any>     # any character but '\n'
line    :  offset  blank  value  blank  comment  '\n'
        | 'SUB'  blank  value  blank  comment  '\n'
        ;

blank    : <blank>
        | blank <blank>
        ;

offset   :  '0x'  <hex>
        |  offset  <hex>
        ;


value    :  offset
        |  'invalid'
        |  'substitution'
        ;


comment  :  '#'  <any>
        |  comment  <any>
        ;
```

A line where the offset is `'SUB'` is used to specify the default substitution character.

If the table is set to `'substitution'`, the **iconv** converter using this table uses the SUB value for this offset.

If the value is set to `'invalid'`, the **iconv** converter using this table returns error for its offset.

If the offset is found in the source code set conversion table file multiple times, the last entry is used in the compilation of the translation table.

The offset and value must be in the range of 0x00 through 0xff, inclusive.

The following is an excerpt of a code set conversion table:

```
SUB     0x1a     substitute character
0x80    0xc7     C  cedilla
0x81    0xfc     u  diaeresis
```

```
0x82    0xe9     e  acute
0x83    0xe2     a  circumflex
0x84    0xe4     a  diaeresis
0x85    0x40     a  grave
0x9F            substitution
0xff            invalid
```

If successful, the **genxlt** command exits with a value of 0. If the output file cannot be opened, the **genxlt** command is unsuccessful and exits with a value of 1. If a syntax error is detected in the input stream, the **genxlt** command will exit immediately with a value of 2, and write to standard error the line numbers where the syntax error occurred.

The name of the file generated by the **genxlt** command must follow the naming convention below in order for the **iconv** subsystem to recognize it as a conversion file:

```
fromcode: "IBM-850"
tocode: "ISO8859-1"
conversion table file: "IBM-850_ISO8859-1"
```

The conversion table file name is formed by concatenating the `tocode` file name onto the `fromcode` file name, with an underscore between the two.

Example

To generate a non-English, user-defined code set conversion table, enter:

```
cp  /usr/lib/nls/loc/iconvTable/ISO8859-1_IBM-850_src  $HOME
vi  $HOME/ISO8859-1_IBM-850_src
genxlt  <  $HOME/ISO8859-1_IBM-850_src  >  cs1_cs2
```

get Command

Purpose

Creates a specified version of a SCCS file.

Syntax

To Get Read-Only Versions of SCCS Files

get [**-g**] [**-m**] [**-n**] [**-p**] [**-s**] [**-c** *Cutoff*] [**-i** *List*] [**-r** *SID*] [**-t**] [**-x** *List*] [**-w** *String*] [**-l** [**p**]] [**-L**] *File* ...

To Get Editable Versions of SCCS Files

get [**-e**] [**-k**] [**-b**] [**-s**] [**-c** *Cutoff*] [**-i** *List*] [**-r** *SID*] [**-t**] [**-x** *List*] [**-l** [**p**]] [**-L**] *File* ...

Description

The **get** command reads a specified version of the Source Code Control System (SCCS) file and creates an ASCII text file according to the specified flags. The **get** command then writes each text file to a file having the same name as the original SCCS file but without the **s.** prefix (the **g-file**).

Flags and files can be specified in any order, and all flags apply to all named files. If you specify a directory for the *File* parameter, the **get** command performs the requested actions on all files in the directory that begin with the **s.** prefix. If you specify a **-** (minus sign) for the *File* parameter, the **get** command reads standard input and interprets each line as the name of an SCCS file. The **get** command continues to read input until it reads an end-of-file character.

If the effective user has write permission in the directory containing the SCCS files but the real user does not, then only one file can be named when the **-e** flag is used.

Note: The **get** command supports the Multibyte Character Set (MBCS) for the file name and string data specified with the **w** flag.

Getting Read-Only File Versions

The **get** command creates both read-only versions and editable versions of a file. Read-only versions of files should be used if the application does not require changes to the file contents. Read-only versions of source code files can be compiled. Text files can be displayed or printed from read-only versions.

The difference between an editable and a read-only version is important when using identification keywords. *Identification keywords* are symbols expanded to some text value when the **get** command retrieves the file as read-only. In editable versions, keywords are not expanded. Identification keywords can appear anywhere in an SCCS file. See the **prs** command for further information on identification keywords.

SCCS Files

In addition to the file with the **s.** prefix (the **s-file**), the **get** command creates several auxiliary files: the **g-file**, **l-file**, **p-file**, and **z-file**. These files are identified by their tag, which is the letter before the hyphen. The **get** program names auxiliary files by replacing the leading **s.** in the SCCS file name with the appropriate tag, except for the **g-file**, which is named by removing the **s.** prefix. So, for a file named **s.sample**, the auxiliary file names would be **sample**, **l.sample**, **p.sample**, and **z.sample**.

These files serve the following purposes:

Item	Description
s-file	Contains the original file text and all the changes (deltas) made to the file. It also includes information about who can change the file contents, who has made changes, when those changes were made, and the nature of changes made. You cannot edit this file directly because it is read-only. However, it contains the information needed by the SCCS commands to build the **g-file**, which you can edit.
g-file	An ASCII text file that contains the text of the SCCS file version that you specify with the **-r** flag (or the latest trunk version by default). You can edit this file directly. When you have made all your changes and want to make a new delta to the file, you can then run the **delta** command on the file. The **get** command creates the **g-file** in the current directory. Whenever it runs the **get** command creates a g-file, unless the **-g** flag or the **-p** flag is specified. The real user owns it (not the effective user). If you do not specify the **-k** or **-e** flag, the file is read-only. If the **-k** or **-e** flag is specified, the owner has write permission for the **g-file**. You must have write permission in the current directory to create a **g-file**.

Item	Description
l-file	The **get** command creates the **l-file** when the **-l** flag is specified. The **l-file** is a read-only file. It contains a table showing which deltas were applied in generating the **g-file**. You must have write permission in the current directory to create an **l-file**. Lines in the **l-file** have the following format: • A blank character if the delta was applied; otherwise, an asterisk. • A blank character if the delta was applied, or was not applied and ignored. An asterisk appears if the delta was not applied and not ignored. • A code indicating a special reason why the delta was or was not applied: ***Blankspace*** Included or excluded usually **I** Included using the **-i** flag **X** Excluded using the **-x** flag **C** Cut off using the **-c** flag • The SID. • The date and time the file was created. • The login name of person who created the delta. Comments and Modification Requests (MR) data follow on subsequent lines, indented one horizontal tab character. A blank line ends each entry. For example, for a delta cutoff with the **-c** flag, the entry in the l-file might be: `**C 1.3 85/03/13 12:44:16 pat` and the entry for the initial delta might be: `1.1 85/02/27 15:42:20 pat` `date and time created 85/02/27 15:42:20 by pat`
p-file	The **get** command creates the **p-file** when the **-e** or **-k** flag is specified. The **p-file** passes information resulting from a **get -e** command to a **delta** command. The **p-file** also prevents a subsequent execution of a **get -e** command for the same SID until a **delta** command is run or the joint edit key letter (**j**) is set in the SCCS file. The **j** key letter allows several **get** commands to be run on the same SID. The **p-file** is created in the directory containing the SCCS *File*. To create a **p-file** in the SCCS directory, you must have write permission in that directory. The permission code of the **p-file** is read-only to all but its owner, and it is owned by the effective user. The **p-file** should not be directly edited by the owner. The **p-file** contains: • Current SID • SID of new delta to be created • User name • Date and time of the **get** command • **-i** flag, if present • **-x** flag, if present The **p-file** contains an entry with the preceding information for each pending delta for the file. No two lines have the same new delta SID.

Item	Description
z-file	The **z-file** is a lock mechanism against simultaneous updates. The **z-file** contains the binary process number of the **get** command that created it. This file is created in the directory containing the SCCS file and exists only while the **get** command is running.

When you use the **get** command, it displays the SID being accessed and the number of lines created from the SCCS file. If you specify the **-e** flag, the SID of the delta to be made appears after the SID is accessed and before the number of lines created. If you specify more than one file, a directory, or standard input, the **get** command displays the file name before each file is processed. If you specify the **-i** flag, the **get** command lists included deltas below the word `Included`. If you specify the **-x** flag, the **get** command lists excluded deltas below the word `Excluded`.

The following table illustrates how the **get** command determines both the SID of the file it retrieves and the pending SID. The SID Specified column shows various ways the SID can be specified with the **-r** flag. The first column also illustrates various conditions that can exist, including whether or not the **-b** flag is used with the **get -e** command. The SID Retrieved column indicates the SID of the file that makes up the **g-file**. The SID of Delta to Be Created column indicates the SID of the version that will be created when the **delta** command is applied.

SID Determination		
SID Specified	**SID Retrieved**	**SID of Delta to Be Created**
none [1] **-b Used?** no **Other Conditions** R defaults to mR [2]	mR.mL	mR.(mL+1)
none [1] **-b Used?** yes **Other Conditions** R defaults to mR	mR.mL	mR.mL.(mB+1).1
R **-b Used?** no **Other Conditions** R>mR	mR.mL	R.1 [3]
R **-b Used?** no **Other Conditions** R=mR	mR.mL	mR.(mL+1)
R **-b Used?** yes **Other Conditions** R>mR	mR.mL	mR.mL.(mB+1).1

SID Determination *(continued)*		
SID Specified	**SID Retrieved**	**SID of Delta to Be Created**
R **-b Used?** yes **Other Conditions** R=mR	mR.mL	mR.mL.(mB+1).1
R **-b Used?** N/A **Other Conditions** R<mR and R does not exist	hR.mL [4]	hR.mL.(mB+1) .1
R **-b Used?** N/A **Other Conditions** Trunk successor in release > R and R exists	R.mL	R.mL.(mB+1).1
R.L. **-b Used?** no **Other Conditions** No trunk successor	R.L.	R.(L+1)
R.L. **-b Used?** yes **Other Conditions** No trunk successor	R.L.	R.L(mB+1).1
R.L. **-b Used?** N/A **Other Conditions** Trunk successor in release > or = R	R.L.	R.L.(mB+1).1
R.L.B. **-b Used?** no **Other Conditions** No branch successor	R.L.B.mS	R.L.B.(mS+1)
R.L.B. **-b Used?** yes **Other Conditions** No branch successor	R.L.B.mS	R.L.(mB+1).1

SID Determination *(continued)*		
SID Specified	**SID Retrieved**	**SID of Delta to Be Created**
R.L.B.S. **-b Used?** no **Other Conditions** No branch successor	R.L.B.S.	R.L.B.(S+1)
R.L.B.S. **-b Used?** yes **Other Conditions** No branch successor	R.L.B.S.	R.L.(mB+1).1
R.L.B.S. **-b Used?** N/A **Other Conditions** Branch successor	R.L.B.S.	R.L.(mB+1).1

Note: In the SID Determination table, the letters R, L, B, and S are the release, level, branch, and sequence components of the SID. The letter *m* signifies maximum.

[1] Applies only if the **-d** (default SID) flag is not present in the file (see the **admin** command).

[2] The mR indicates the maximum existing release.

[3] Forces creation of the first delta in a new release.

[4] The hR is the highest existing release lower than the specified, nonexistent release R.

Identification Keywords

Identifying information is inserted into the text retrieved from the SCCS file by replacing identification keywords with their value wherever they occur. The following keywords may be used in the text stored in an SCCS file:

Keyword	Value
%M%	Module name: either the value of the **m** flag in the file, or, if absent, the name of the SCCS file with the **s.** removed.
%I%	SCCS identification (SID) (*%R%.%L%* or *%R%.%L%.%B%.%S%*) of the retrieved text.
%R%	Release.
%L%	Level.
%B%	Branch.
%S%	Sequence.
%D%	Current date, formatted as *YY/MM/DD*.
%H%	Current date, formatted as *MM/DD/YY*.
%T%	Current time, formatted as *HH:MM:SS*.
%E%	Date newest applied delta was created, formatted as *YY/MM/DD*.
%G%	Date newest applied delta was created, formatted as *MM/DD/YY*.

Keyword	Value
01:51:20	Time newest applied delta was created, formatted as *HH:MM:SS*.
%Y%	Module type: value of the **t** flag in the SCCS file.
%F%	SCCS file name.
%P%	SCCS absolute path name.
%Q%	The value of the **-q** flag in the file.
%C%	Current line number. This keyword is intended for identifying messages output by the program, such as `this should not have happened` error messages. The **%C%** is not intended to be used on every line to provide sequence numbers.
%Z%	The four-character string `@(#)` recognizable by *what.*
%W%	A shorthand notation for constructing *what* strings: `%W% = %Z%%M%<tab>%I%`
%A%	Another shorthand notation for constructing *what* strings: `%A% = %Z%%Y% %M% %I% %Z%`

Flags

Item	Description
-b	Specifies that the delta to be created should have an SID in a new branch. The new SID is numbered according to the rules given in the SID determination table. You can use the **-b** flag only with the **-e** flag. It is only necessary when you want to branch from a leaf delta (a delta without a successor). Attempting to create a delta at a nonleaf delta automatically results in a branch, even if the **b** header flag is not set. If you do not specify the **b** header flag in the SCCS file, the **get** command ignores the **-b** flag because the file does not allow branching.
-c *Cutoff*	Specifies a cutoff date and time, in the form *YY[MM[DD[HH[MM[SS]]]]]*. The **get** command includes no deltas to the SCCS file created after the specified cutoff in the g-file. The values of any unspecified items in the *Cutoff* variable default to their maximum allowable values. Thus, a cutoff date and time specified with only the year (YY) would specify the last month, day, hour, minute, and second of that year. Any number of nonnumeric characters can separate the two-digit items of the *Cutoff* variable date and time. This allows you to specify a date and time in a number of ways, as follows: `-c85/9/2,9:00:00` `-c"85/9/2 9:00:00"` `"-c85/9/2 9:00:00"`
-e	Indicates that the g-file being created is to be edited by the user applying the **get** command. The changes are recorded later with the **delta** command. The **get -e** command creates a p-file that prevents other users from issuing another **get -e** command and editing a second g-file on the same SID before the **delta** command is run. The owner of the file can override this restriction by allowing joint editing on the same SID through the use of the **admin** command with the **-fj** flag. Other users, with permission, can obtain read-only copies by using the **get** command without the **-e** flag. The **get -e** command enforces SCCS file protection specified with the ceiling, floor, and authorized user list in the SCCS file. See the **admin** command. **Note:** If you accidentally ruin the g-file created using the **get -e** command, you can recreate the file with the **get -k** command.
-g	Suppresses the actual creation of the **g-file**. Use the **-g** flag primarily to create an **l-file** or to verify the existence of a particular SID. Do not use it with the **-e** flag.

Item	Description
-i *List*	Specifies a list of deltas to be included in the creation of a **g-file**. The SID list format consists of a combination of individual SIDs separated by commas and SID ranges indicated by two SIDs separated by a hyphen. You can specify the same SIDs with either of the following command lines: `get -e -i1.4,1.5,1.6 s.file` `get -e -i1.4-1.6 s.file` You can specify the SCCS identification of a delta in any form shown in the SID Specified column of the previous table. The **get** command interprets partial SIDs as shown in the SID Retrieved column.
-k	Suppresses replacement of identification keywords in the **g-file** by their value. The **-k** flag is implied by the **-e** flag. If you accidentally ruin the **g-file** created using the **get -e** command, you can recreate the file by reissuing the **get** command with the **-k** flag instead of the **-e** flag.
-l[**p**]	Writes a delta summary to an l-file. If you specify **-lp**, the delta summary is written to standard output, and the **get** command does not create the **l-file**. Use this flag to determine which deltas were used to create the g-file currently in use. See the **sccsfile** file for the format of the **l-file**. See also the **-L** flag.
-L	Writes a delta summary to standard output. Specifying the **-L** flag is the same as using the **-lp** flag.
-m	Writes before each line of text in the **g-file** the SID of the delta that inserted the line into the SCCS file. The format is: `SID tab line of text`
-n	Writes the value of the **%M%** keyword before each line of text in the **g-file**. The format is the value of **%M%**, followed by a horizontal tab, followed by the text line. When both the **-m** and **-n** flags are used, the format is: `%M% value tab SID tab line of text`
-p	Writes the text created from the SCCS file to standard output and does not create a g-file. All informative output usually sent to standard output is sent to standard error, unless you specify the **-s** flag with the **-p** flag. In this case, output usually sent to standard output does not appear anywhere.
-r *SID*	Specifies the SCCS identification string (SID) of the SCCS file version to be created. The SID determination table shows the version of the created file and the SID of the pending delta as functions of the specified SID.
-s	Suppresses all output usually written to standard output. Error messages (written to standard error output), remain unaffected.
-t	Accesses the most recently created delta in a given release or for a given release and level.
-w *String*	Substitutes the *String* value for the **%W%** keyword in g-files not intended for editing.
-x *List*	Excludes the specified list of deltas in the creation of the **g-file**. See the **-i** flag for the SID list format.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

The following descriptions and examples illustrate the differences between read-only and editable versions of files.

1. To print the current date and SID in a file, put the following symbols in the file:

   ```
   %H% %I%
   ```

   %H% is the symbol for the current date and **%I%** is the symbol for the SID. When the **get** command retrieves a file as editable, it leaves the symbols in the file and does not perform text value substitution.

2. The following example of the **get** command builds the version with the highest SID, because the example does not specify a version of the file:

   ```
   $ ls
   s.test.c
   $ get s.test.c
   3.5
   59 lines
   $ ls
   s.test.c test.c
   ```

3. In the next two examples, the **-r** flag specifies which version to get:

   ```
   $ get -r1.3 s.test.c
   1.3
   67 lines

   $ get -r1.3.1.4 s.test.c
   1.3.1.4
   50 lines
   ```

4. If you specify just the release number of the SID, the **get** command finds the file with the highest level within that release number.

   ```
   $ get -r2 s.test.c
   2.7
   21 lines
   ```

5. If the SID specified is greater than the highest existing SID, the **get** command gets the highest existing SID. If the SID specified is lower than the lowest existing SID, SCCS writes an error message. In the following example, release 7 is the highest existing release:

   ```
   $ get -r9 s.test.c
   7.6
   400 lines
   ```

6. The **-t** flag gets the top version in a given release or level. The top version is the most recently created delta, independent of its location. In the next example, the highest existing delta in release 3 is 3.5, while the most recently created delta is 3.2.1.5.

   ```
   $ get -t -r3 s.test.c
   3.2.1.5
   46 lines
   ```

7. The previous examples use the **get** command to get a read-only file. To create a copy of the file that can be edited and used to create a new delta, use the **get** command with the **-e** flag. Use **unget** to undo the effect of the **get -e** command and discard any changes made to the file before a delta is created. The following example shows how to use the **-e** flag:

```
$ ls
s.test.c
$ get -e s.test.c
1.3
new delta 1.4
67 lines
$ ls
p.test.c s.test.c test.c
```

The working file is `test.c`. If you edit this file and save the changes with the **delta** command, SCCS creates a new delta with an SID of `1.4`. The file `p.test.c` is a temporary file used by SCCS to keep track of file versions.

In the previous example, you could have used the **-r** flag to get a specific version. Assuming release 1 is the highest existing release and that delta `1.3` already exists and is the highest delta in release, the following three uses of the **get** command are equivalent:

```
$ get -e s.test.c
$ get -e -r1 s.test.c
$ get -e -r1.3 s.test.c
```

8. To start using a new (higher in value) release number, get the file with the **-r** flag and specify a release number greater than the highest existing release number. In the next example, release 2 does not yet exist:

```
$ get -e -r2 s.test.c
1.3
new delta 2.1
67 lines
```

Notice that the **get** command indicates the version of the new delta that will be created if the **delta** command stores changes to the SCCS file.

9. To create a branch delta, use the **-r** flag and specify the release and level where the branch occurs. In the next example, deltas `1.3` and `1.4` already exist.

```
$ get -e -r1.3 s.test.c
1.3
new delta 1.3.1.1
67 lines
```

Creates deltas on branches using the same methods.

To edit a file, get the file version using the **get -e** command and save the changes with the **delta** command. Several different editable versions of an SCCS file can exist as long as each one is in a different directory. If you try to put duplicates of an editable file version into a directory (using the **get** command) without using the **delta** command, SCCS writes an error message.

To get the same editable file version more than once, set the **j** header flag in the SCCS file with the **admin** command. Set the **j** option by using the **-f** flag. You can then get the same SID several times from different directories, creating a separate file for each **get** command. Although the files originate from a single SID, SCCS gives each of them a unique new SID.

10. In the following example, the **pwd** command displays the current directory. Then the **j** option is set with the **admin** command:

Note: You must have write access in both directories to issue the commands in this example.

```
$ pwd
/home/marty/sccs
$ admin -fj s.test.c
```

11. Then use the **get** command to retrieve the latest version of the file:

Note: You must have write access in both directories to issue the commands in this example.

```
$ get -e s.test.c
1.1
```

```
new delta 1.2
5 lines
```

12. Change to the /home/new directory, and issue the **get** command again.

    Note: You must have write access in both directories to issue the commands in this example.

    ```
    $ cd /home/new
    $ get -e /home/marty/sccs/s.test.c
    1.2
    new delta 1.1.1.1
    5 lines
    ```

    Notice that SCCS creates two deltas, `1.2` and `1.1.1.1`, from the single original file version of `1.1`. Look at the **p.test.c** file. It shows a separate entry for each version currently in use. The **p.test.c** file remains in the directory until you take care of both file versions with either the **delta** command or the **unget** command.

Files

Item	Description
/usr/bin/get	Contains the **get** command.

getconf Command

Purpose

Writes system configuration variable values to standard output.

Syntax

getconf [**-v** *specification*] [*SystemwideConfiguration* | *PathConfiguration PathName*] [*DeviceVariable DeviceName*]

getconf -a

Description

The **getconf** command, invoked with the *SystemwideConfiguration* parameter, writes the value of the variable, as specified by the *SystemwideConfiguration* parameter, to standard output.

The **getconf** command, invoked with the *PathConfiguration* and *Pathname* parameters, writes the value of the variable, as specified by the *PathConfiguration* parameter for the path specified by the *PathName* parameter, to standard output.

The **getconf** command, invoked with the **-a** flag, writes the values of all system configuration variables to standard output.

The **getconf** command, invoked with the *DeviceVariable* and *DeviceName* parameters, writes the value of the disk device name or location, for the device path specified by the *DeviceName* parameter, to the standard output.

If the specified variable is defined on the system and its value is described to be available from the **confstr** subroutine, the value of the specified variable is written in the following format:

```
"%s\n", <value>
```

Otherwise, if the specified variable is defined on the system, its value is written in the following format:

```
"%d\n", <value>
```

If the specified variable is valid but undefined on the system, the following is written to standard output:

```
"undefined\n"
```

If the variable name is invalid or an error occurs, a diagnostic message is written to the standard error.

Flags

Item	Description
-v *specification*	Indicates a specific specification and version for which configuration variables are to be determined. If this flag is not specified, the values returned will correspond to an implementation default XBS5 conforming compilation environment.
-a	Writes the values of all system configuration variables to standard output.

Parameters

Item	Description
PathName	Specifies a path name for the *PathConfiguration* parameter.
SystemwideConfiguration	Specifies a system configuration variable.
PathConfiguration	Specifies a system path configuration variable.
DeviceName	Specifies the path name of a device.
DeviceVariable	Specifies a device variable.

When the symbol listed in the first column of the following table is used as the **system_var** operand, **getconf** will yield the same value as **confstr** when called with the value in the second column:

Note: The **_CS_AIX_ARCHITECTURE** and **_CS_AIX_BOOTDEV** variables, used as parameters to **confstr**, are available only to the root user.

system_var	confstr Name Value
BOOT_DEVICE	_CS_AIX_BOOTDEV
MACHINE_ARCHITECTURE	_CS_AIX_ARCHITECTURE
MODEL_CODE	_CS_AIX_MODEL_CODE
PATH	_CS_PATH
XBS5_ILP32_OFF32_CFLAGS	_CS_XBS5_ILP32_OFF32_CFLAGS
XBS5_ILP32_OFF32_LDFLAGS	_CS_XBS5_ILP32_OFF32_LDFLAGS
XBS5_ILP32_OFF32_LIBS	_CS_XBS5_ILP32_OFF32_LIBS
XBS5_ILP32_OFF32_LINTFLAGS	_CS_XBS5_ILP32_OFF32_LINTFLAGS
XBS5_ILP32_OFFBIG_CFLAGS	_CS_XBS5_ILP32_OFFBIG_CFLAGS
XBS5_ILP32_OFFBIG_LDFLAGS	_CS_XBS5_ILP32_OFFBIG_LDFLAGS
XBS5_ILP32_OFFBIG_LIBS	_CS_XBS5_ILP32_OFFBIG_LIBS
XBS5_ILP32_OFFBIG_LINTFLAGS	_CS_XBS5_ILPBIG_OFF32_LINTFLAGS
XBS5_LP64_OFF64_CFLAGS	_CS_XBS5_LP64_OFF64_CFLAGS
XBS5_LP64_OFF64_LDFLAGS	_CS_XBS5_LP64_OFF64_LDFLAGS
XBS5_LP64_OFF64_LIBS	_CS_XBS5_LP64_OFF64_LIBS

system_var	confstr Name Value
XBS5_LP64_OFF64_LINTFLAGS	_CS_XBS5_LP64_OFF64_LINTFLAGS
XBS5_LPBIG_OFFBIG_CFLAGS	_CS_XBS5_LPBIG_OFFBIG_CFLAGS
XBS5_LPBIG_OFFBIG_LDFLAGS	_CS_XBS5_LPBIG_OFFBIG_LDFLAGS
XBS5_LPBIG_OFFBIG_LIBS	_CS_XBS5_LPBIG_OFFBIG_LIBS
XBS5_LPBIG_OFFBIG_LINTFLAGS	_CS_XBS5_LPBIG_OFFBIG_LINTFLAGS

Environment Variables

The following environment variables affect the execution of **getconf**:

Item	Description
LANG	Provide a default value for the internationalisation variables that are unset or null. If LANG is unset or null, the corresponding value from the implementation-dependent | default locale will be used. If any of the internationalisation variables contains an invalid setting, the utility will behave as if none of the variables had been defined.
LC_CALL	If set to a non-empty string value, override the values of all the other internationalisation variables.
LC_CTYPE	Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single- as opposed to multi-byte characters in arguments).
LC_MESSAGES	Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.
NLSPATH	Determine the location of message catalogues for the processing of LC_MESSAGES.

Systemwide Configuration Variables

The *SystemwideConfiguration* parameter specifies system configuration variables whose values are valid throughout the system. There are two kinds of system configuration variables:

- Systemwide configuration variables
- System standards configuration variables

Systemwide Configuration Variables

Systemwide configuration variables contain the minimum values met throughout all portions of the system. The following list defines the systemwide configuration variables used with the **getconf** command:

Item	Description
_CS_PATH	Value for the **PATH** environment variable used to find commands.
ARG_MAX	Maximum length, in bytes, of the arguments for one of the **exec** subroutines, including environment data.
BC_BASE_MAX	Maximum value allowed for the **obase** variable with the **bc** command.
BC_DIM_MAX	Maximum number of elements permitted in an array by the **bc** command.
BC_SCALE_MAX	Maximum value allowed for the **scale** variable with the **bc** command.
BC_STRING_MAX	Maximum length of a string constant accepted by the **bc** command.

Item	Description
CHARCLASS_NAME_MAX	Maximum number of bytes in a character class name.
CHAR_BIT	Number of bits in a type **character**.
CHAR_MAX	Maximum value of a type **character**.
CHAR_MIN	Minimum value of a type **character**.
CHILD_MAX	Maximum number of simultaneous processes for each real user ID.
CLK_TCK	Number of clock ticks per second returned by the **time** subroutine.
COLL_WEIGHTS_MAX	Maximum number of weights that can be assigned to an entry in the **LC_COLLATE** locale stanza in a locale-definition file.
CS_PATH	Value of the **PATH** environment variable used to find commands.
EXPR_NEST_MAX	Maximum number of expressions that can be nested within parentheses by the **expr** command.
INT_MAX	Maximum value of a type **int**.
INT_MIN	Minimum value of a type **int**.
LINE_MAX	Maximum length, in bytes, of a command's input line (either standard input or another file) when the utility is described as processing text files. The length includes room for the trailing new-line character.
LONG_BIT	Number of bits in a type **long int**.
LONG_MAX	Maximum value of a type **long int**.
LONG_MIN	Minimum value of a type **long int**.
MB_LEN_MAX	Maximum number of bytes in a character for any supported locale.
NGROUPS_MAX	Maximum number of simultaneous supplementary group IDs for each process.
NL_ARGMAX	Maximum value of digit in calls to the **printf** and **scanf** subroutines.
NL_LANGMAX	Maximum number of bytes in a LANG name.
NL_MSGMAX	Maximum message number.
NL_NMAX	Maximum number of bytes in an N-to-1 collation mapping.
NL_SETMAX	Maximum set number.
NL_TEXTMAX	Maximum number of bytes in a message string.
NZERO	Default process priority.
OPEN_MAX	Maximum number of files that one process can have open at one time.
PATH	Sequence of colon-separated path prefixes used to find commands.
RE_DUP_MAX	Maximum number of repeated occurrences of a regular expression permitted when using the interval-notation parameters, such as the *m* and *n* parameters with the **ed** command.
SCHAR_MAX	Maximum value of a type **signed char**.
SCHAR_MIN	Minimum value of a type **signed char**.
SHRT_MAX	Maximum value of a type **short**.
SHRT_MIN	Minimum value of a type **short**.
SSIZE_MAX	Maximum value of an object of type **ssize_t**.
STREAM_MAX	Number of streams that one process can have open at one time.

Item	Description
TMP_MAX	Minimum number of unique path names generated by the **tmpnam** subroutine. Maximum number of times an application can reliably call the **tmpnam** subroutine.
TZNAME_MAX	Maximum number of bytes supported for the name of a time zone (not the length of the **TZ** environment variable).
UCHAR_MAX	Maximum value of a type **unsigned char**.
UINT_MAX	Maximum value of a type **unsigned int**.
ULONG_MAX	Maximum value of a type **unsigned long int**.
USHRT_MAX	Maximum value of a type **unsigned short int**.
WORD_BIT	Number of bits in a word or type **int**.
KERNEL_BITMODE	Bit mode of the kernel, 32-bit or 64-bit.
REAL_MEMORY	Real memory size.
HARDWARE_BITMODE	Bit mode of the machine hardware, 32-bit or 64-bit.
MP_CAPABLE	MP-capability of the machine.

System Standards Configuration Variables

System standards configuration variables contain the *minimum* values required by a particular system standard. The **_POSIX_**, **POSIX2_**, and **_XOPEN_** prefixes indicate that the variable contains the minimum value for a system characteristic required by the POSIX 1003.1, POSIX 1003.2, and X/Open system standards, respectively. System standards are systemwide minimums that the system meets to support the particular system standard. Actual Configuration values may exceed these standards. The system standards configuration variables for the **getconf** command are defined as follows:

Item	Description
_POSIX_ARG_MAX	Maximum length, in bytes, of the arguments for one of the **exec** subroutines, including environment data.
_POSIX_CHILD_MAX	Maximum number of simultaneous processes for each real user ID.
_POSIX_JOB_CONTROL	Value of 1 if the system supports job control.
_POSIX_LINK_MAX	Maximum number of links to a single file.
_POSIX_MAX_CANON	Maximum number of bytes in a terminal canonical input queue.
_POSIX_MAX_INPUT	Maximum number of bytes allowed in a terminal input queue.
_POSIX_NAME_MAX	Maximum number of bytes in a file name (not including terminating null).
_POSIX_NGROUPS_MAX	Maximum number of simultaneous supplementary group IDs for each process.
_POSIX_OPEN_MAX	Maximum number of files that one process can have open at one time.
_POSIX_PATH_MAX	Maximum number of bytes in a path name.
_POSIX_PIPE_BUF	Maximum number of bytes guaranteed to be atomic when writing to a pipe.
_POSIX_SAVED_IDS	Value of 1. Each process has a saved set-user-ID and a saved set-group-ID.
_POSIX_SSIZE_MAX	Maximum value that can be stored in an object of type **ssize_t**.

Item	Description
_POSIX_STREAM_MAX	Number of streams that one process can have open at one time.
_POSIX_TIMESTAMP_RESOLUTION	Resolution of all file time stamps in nanoseconds.
_POSIX_TZNAME_MAX	Maximum number of bytes supported for the name of a time zone (not the length of the **TZ** environment variable).
_POSIX_VERSION	Version of the POSIX 1 standard (C Language Binding) to which the operating system conforms.
_XOPEN_CRYPT	Value of 1 if the system supports the X/Open Encryption Feature Group.
_XOPEN_ENH_I18N	Value of 1 if the system supports the X/Open Enhanced Internationalisation Feature Group.
_XOPEN_SHM	Value of 1 if the system supports the X/Open Shared Memory Feature Group.
_XOPEN_VERSION	Version of the X/Open Portability Guide to which the operating system conforms.
_XOPEN_XCU_VERSION	Version of the X/Open Commands and Utilities specification to which the operating system conforms.
_XOPEN_XPG2	Value of 1 if the system supports the X/Open Portability Guide, Volume 2, January 1987, XVS System Calls and Libraries, otherwise undefined.
_XOPEN_XPG3	Value of 1 if the system supports the X/Open Specification, February 1992, System Interfaces and Headers, Issue 3, otherwise undefined.
_XOPEN_XPG4	Value of 1 if the system supports the X/Open CAE Specification, July 1992, System Interfaces and Headers, Issue 4, otherwise undefined.
POSIX2_BC_BASE_MAX	Maximum value allowed for the **obase** variable with the **bc** command.
POSIX2_BC_DIM_MAX	Maximum number of elements permitted in an array by the **bc** command.
POSIX2_BC_SCALE_MAX	Maximum value allowed for the **scale** variable with the **bc** command.
POSIX2_BC_STRING_MAX	Maximum length of a string constant accepted by the **bc** command.
POSIX2_CHAR_TERM	Value of 1 if the system supports at least one terminal type; otherwise it has the value -1.
POSIX2_COLL_WEIGHTS_MAX	Maximum number of weights that can be assigned to an entry of the **LC_COLLATE** locale variable in a locale-definition file.
POSIX2_C_BIND	Value of 1 if the system supports the C Language Binding Option from POSIX 2; otherwise, it has the value -1.
POSIX2_C_DEV	Value of 1 if the system supports the C Language Development Utilities from POSIX 2; otherwise, it has the value -1.
POSIX2_C_VERSION	Version of the POSIX 2 standard (C Language Binding) to which the operating system conforms.
POSIX2_EXPR_NEST_MAX	Maximum number of expressions that can be nested within parentheses by the **expr** command.

Item	Description
POSIX2_FORT_DEV	Value of 1 if the system supports the FORTRAN Development Utilities Option from POSIX 2; otherwise, it has the value -1.
POSIX2_FORT_RUN	Value of 1 if the system supports the FORTRAN Runtime Utilities Option from POSIX 2; otherwise, it has the value -1.
POSIX2_LINE_MAX	The maximum length, in bytes, of a command's input line (either standard input or another file) when the command is described as processing text files. The length includes room for the trailing new-line character.
POSIX2_LOCALEDEF	Value of 1 if the system supports the creation of the locales by the **localedef** command; otherwise, it is undefined.
POSIX2_RE_DUP_MAX	Maximum number of repeated occurrences of a regular expression permitted when using the interval-notation parameters, such as the *m* and *n* parameters with the **ed** command.
POSIX2_SW_DEV	Value of 1 if the system supports the Software Development Utilities Option; otherwise, it has the value -1.
POSIX2_UPE	Value of 1 if the system supports the User Portability Utilities Option from POSIX 2; otherwise, it as the value -1.
POSIX2_VERSION	Date of approval of the most current version of the POSIX 2 standard that the system supports. The date is a six-digit number, with the first four digits signifying the year and the last two digits the month. Different versions of the POSIX 2 standard are periodically approved by the IEEE Standards Board, and the date of approval is used to distinguish between different versions.

System Path Configuration Variables

The *PathConfiguration* parameter specifies system path configuration variables whose values contain information about paths and path structures in the system. The following list defines these variables:

Item	Description
_POSIX_CHOWN_RESTRICTED	The **chown**() subroutine is restricted to a process with appropriate privileges, and to changing the group ID of a file only to the effective group ID of the process or to one of its supplementary group IDs. If the *PathName* parameter refers to a directory, the value returned applies to any files except directories that exist or can be created within the directory.
_POSIX_NO_TRUNC	Path names longer than the limit specified by the *NAME_MAX* variable will generate an error. If the *PathName* parameter refers to a directory, the value returned applies to file names within the directory.
_POSIX_VDISABLE	Terminal special characters, defined in the **termios.h** file, can be disabled using this character value.
LINK_MAX	Maximum number of links to a single file. If the *PathName* parameter refers to a directory, the value returned applies to the directory.
MAX_CANON	Maximum number of bytes in a terminal canonical input line.

Item	Description
MAX_INPUT	Maximum number of bytes for which space is available in a terminal input queue.
NAME_MAX	Maximum number of bytes in a file name (not including terminating null). If the *PathName* parameter refers to a directory, the value returned applies to the file names within the directory.
PATH_MAX	Maximum number of bytes in a path name, including the terminating null character. If the *PathName* parameter refers to a directory, the value returned is the maximum length of a relative path name when the specified directory is the working directory.
PIPE_BUF	Maximum number of bytes guaranteed to be atomic when writing to a pipe. If the *PathName* parameter refers to a FIFO or a pipe, the value returned applies to the referenced object. If the *PathName* parameter refers to a directory, the value returned applies to any FIFO that exists or can be created within the directory.
DISK_PARTITION	Physical partition size of the disk. **Note:** For the **DISK_PARTITION** path configuration variable, the *PathName* parameter must specify the complete path of the disk for which information is being queried.
DISK_SIZE	Disk size in megabytes. **Note:** For the **DISK_SIZE** path configuration variable, the *PathName* parameter must specify the complete path of the disk for which information is being queried.

Device Variables

The *DeviceVariable* parameter indicates that the *DeviceName* parameter is the path of a device, such as **/dev/hdisk0**. Given the path of a disk, the **getconf** command displays the device name or location of the disk.

Item	Description
DISK_DEVNAME	Device name or location of the device.

Exit Status

This command returns the following exit values:

Item	Description
0	The specified variable is valid and information about its current state was successfully written.
>0	An error occurred.

Examples

1. To display the value of the **ARG_MAX** variable, enter the following command:

```
getconf ARG_MAX
```

2. To display the value of the **NAME_MAX** variable for the **/usr** directory, enter the following command:

```
getconf NAME_MAX /usr
```

3. The following sequence of shell commands shows how to handle unspecified results:

```
if value=$(getconf PATH_MAX /usr)
then
     if [ "$value" = "undefined" ]
     then
             echo
                   The value of PATH_MAX in /usr is undefined.
     else
             echo
                   The value of PATH_MAX in /usr is $value.
     fi
else
     echo Error in the getconf command.
fi
```

4. If the command:

```
getconf _XBS5_ILP32_OFF32
```

   does not write -1\n or undefined\n to standard output, then commands of the form:

```
getconf -v XBS5_ILP32_OFF32 ...
```

   determine values for configuration variables corresponding to the XBS5_ILP32_OFF32 compilation environment specified in **c89**, Extended Description.

5. If the command:

```
getconf _XBS5_ILP32_OFFBIG
```

   does not write -1\n or undefined\n to standard output, then commands of the form:

```
getconf -v XBS5_ILP32_OFFBIG ...
```

   determine values for configuration variables corresponding to the XBS5_ILP32_OFFBIG compilation environment specified in **c89**, Extended Description.

6. If the command:

```
getconf _XBS5_LP64_OFF64
```

   does not write -1\n or undefined\n" to standard output, then commands of the | form:

```
getconf -v XBS5_LP64_OFF64 ...
```

   determine values for configuration variables corresponding to the XBS5_LP64_OFF64 compilation environment specified in c89, Extended Description.

7. If the command:

```
getconf _XBS5_LPBIG_OFFBIG
```

   does not write -1\n or undefined\n to standard output, then commands of the form:

```
getconf -v _XBS5_LPBIG_OFFBIG
```

   determine values for configuration variables corresponding to the XBS5_LPBIG_OFFBIG compilation environment specified in c89, Extended Description.

8. To determine the disk size for disk hdisk0, as root user, enter the following command:

```
getconf DISK_SIZE /dev/hdisk0
```

9. To determine the real memory size, enter the following command:

```
getconf REAL_MEMORY
```

10. To determine if the machine hardware is 32-bit or 64-bit, enter the following command:

```
getconf HARDWARE_BITMODE
```

11. To determine if the kernel is 32-bit or 64-bit, enter the following command:

```
getconf KERNEL_BITMODE
```

12. To determine the device name or location of disk `hdisk0`, enter the following command:

```
getconf DISK_DEVNAME hdisk0
```

Files

Item	Description
/usr/bin/getconf	Contains the **getconf** command.
/usr/include/limits.h	Defines system configuration variables.
/usr/include/unistd.h	Defines system configuration variables.

getdev Command

Purpose

Lists devices that match the specified criteria.

Syntax

getdev [**-a**] [**-e**] [*Criteria*] [*DeviceList*]

Description

Lists devices that match the given criteria. The criteria is given in the form of expressions. The **getdev** command can check all devices on the system or a specified list of devices.

Flags

Item	Description
-a	Specifies that a device must match all criteria to be included in the list generated by this command. The **-a** flag has no effect if no criteria are defined.
-e	Specifies that the devices provided in the *devicelist* be excluded from the list generated by the **getdev** command. Without the *-e* flag only devices in the *devicelist* are generated. This flag is ignored if no devices are specified.

Parameters

Item	Description
Criteria	Defines criteria that a device must match before it can be included in the generated list. *Criteria* can be specified as an expression or a list of expressions which a device must meet for it to be included in the list generated by **getdev**. If no criteria are provided, all devices are included in the list. Devices must satisfy at least one of the criteria in the list. However, the **-a** option can be used to specify that a "logical and" operation should be performed. Then, only those devices that match all of the criteria in a list will be included. There are four possible expression types which the criteria specified in the *Criteria* parameter may follow: ***Attribute*=*Value*** Fetches all devices with a member which has*Attribute* defined and is equal to *Value*. ***Attribute*!=*Value*** Fetches all devices with a member which has *Attribute* defined and does not equal *Value*. ***Attribute*:*** Fetches all devices with a member which has *Attribute* defined. ***Attribute*!:*** Fetches all devices with a member which does not have *Attribute* defined. The following are the valid device attributes: **alias** The name by which a device is known. **desc** A description of the device. **type** A token describing the type of the device. The valid set of values for the **type** attribute can be obtained by executing the following command. **`odmget PdDv | grep -w class | awk '{print $3}' | sed 's/"//g' | sort | uniq`** **status** The current state of the device.The list of possible values for status are: 1.Defined 2.Available 3.Stopped 4.Diagnose The values for **status** are not case sensitive.
DeviceList	Specifies a space-separated list of devices to be checked for the *Criteria*.

Exit Status

0
The command completed successfully

> 1
Failure has occurred.

Examples

1. To display all devices, enter:

```
getdev
```

2. To list devices which are of type "logical_volume", enter:

```
getdev type=logical_volume
```

3. To list devices which are not of type "logical_volume", enter:

```
getdev type!=logical_volume
```

4. To list devices which are of type "logical_volume" or whose device alias is "sys0", enter:

```
getdev type=logical_volume alias=sys0
```

The output will look similar to the following:

```
hd1
hd2
hd3
hd4
...
sys0
```

5. To list devices which are of type "logical_volume" and whose device alias is "lv01", enter:

```
getdev -a type=logical_volume alias=lv01
```

6. To display devices for which the **status** attribute is defined , enter:

```
getdev status:*
```

7. To display devices for which the **desc** attribute is not defined , enter:

```
getdev desc!:*
```

Files

Item	Description
/usr/sbin/getdev	Contains the **getdev** command

getdgrp Command

Purpose

Lists device classes that match the specified criteria.

Syntax

getdgrp [**-a**] [**-e**] [**-l**][*Criteria*] [*DeviceClassList*]

Description

Lists device classes that contain devices matching the given criteria. The criteria is given in the form of expressions.

Flags

Item	Description
-a	Indicates that a device must match all criteria of the device class to be included in the report generated by this command. The **-a** flag has no effect if no criteria are defined.
-e	Indicates that thedevices classes specified in the parameter list be excluded from the report generated by this command. The **-e** flag has no effect if no devices are specified.

Item	Description
-l	Indicates that all device classes that are subject to the **-e** option and the **dgroup** list, be listed even if they contain no valid device members. This option has no affect if *Criteria* is specified on the command line.

Parameters

Item	Description
Criteria	Defines criteria that a device must match before a device class to which it belongs can be included in the generated list. *Criteria* can be specified as an expression or a list of expressions which a device must meet for its class to be included in the list generated by **getdgrp**. If no criteria are given, all device classes are included in the list. Devices must satisfy at least one of the criteria in the list. However, the **-a** option can be used to specify that a "logical and" operation should be performed. Then, only those classes containing devices that match all of the criteria in a list will be included. There are four possible expression types which the criteria specified in the *Criteria* parameter may follow: ***Attribute*=*Value*** Fetches all device classes with a member which has*Attribute* defined and is equal to *Value*. ***Attribute*!=*Value*** Fetches all device classes with a member which has *Attribute* defined and does not equal *Value*. ***Attribute*:*** Fetches all device classes with a member which has *Attribute* defined. ***Attribute*!:*** Fetches all device classes with a member which does not have *Attribute* defined. The following are the valid device attributes: **alias** The name by which a device is known. **desc** A description of the device. **type** A token describing the type of the device. **status** The current state of the device.The list of possible values for status are : 1.Defined 2.Available 3.Stopped 4.Diagnose The values for **status** are not case sensitive.
DeviceClassList	Specifies device class name in the Customized Device Configuration database or in the Predefined Device Configuration database.

Exit Status

0
The command completed successfully

1
Command syntax was incorrect, invalid option was used, or an internal error occurred.

2
The Customized Devices object class or the Predefined Devices object class could not be opened for reading.

Examples

1. To display all device classes, enter:

   ```
   getdgrp
   ```

   The output looks similar to the following:

   ```
   adapter
   aio
   bus
   cdrom
   disk
   diskette
   gxme
   if
   keyboard
   lft
   logical_volume
   lvm
   memory
   mouse
   planar
   processor
   pty
   pwrmgt
   rcm
   sys
   tape
   tcpip
   tty
   ```

2. To list device classes whose devices are of type "logical_volume", enter:

   ```
   getdgrp type=logical_volume
   ```

   The output looks like the following:

   ```
   logical_volume
   ```

3. To list device classes whose devices are of type "logical_volume" or whose device alias is "sys0", enter:

   ```
   getdgrp type=logical_volume alias=sys0
   ```

   The output looks like the following:

   ```
   logical_volume
   sys
   ```

4. To list device classes whose devices status attribute is defined , enter:

   ```
   getdgrp status=defined
   ```

   The output looks like the following:

   ```
   logical_volume
   posix_aio
   rcm
   ```

5. To display device classes for whose devices the **status** attribute is defined and belong to the "processor" device class, enter:

   ```
   getdgrp status:* processor
   ```

   The output looks like the following:

   ```
   processor
   ```

6. To display device classes for whose devices the **status** attribute is not defined, enter:

```
getdgrp status!:* processor
```

Files

Item	Description
/usr/sbin/getdgrp	Contains the **getdgrp** command

getea Command

Purpose

Retrieves named extended attributes from a file.

Syntax

getea [**-n** *Name*] [**-l**] [**-e** *RegExp*] [**-s**] *FileName*

Description

The **getea** command reads named extended attributes from a file. If the **-n** *Name* parameter is specified then just extended attributes matching *Name* are retrieved.

Note: To prevent naming collisions, JFS2 has reserved the 8-character prefix `(0xf8)SYSTEM(0xF8)` for system-defined extended attributes. Avoid using this prefix for naming user-defined extended attributes.

If the **-e** *RegExp* parameter is specified then just extended attributes matching the regular expression *RegExp* are retrieved. If neither **-n** or **-e** flags are specified all extended attributes are retrieved.

This command is not used to get ACLs. The **aclget** command is used to get ACLs.

Flags

Item	Description
-e *RegExp*	Specifies a regular expression to retrieve all extended attributes which match. The values are displayed in character format.
-l	Specifies to get the extended attributes from the symbolic link itself rather than the file to which it is pointing.
-n *Name*	Specifies the name of specific extended attributes to retrieve. The values are displayed in character format.
-s	Displays only the names and not the values for the extended attributes.
FileName	Specifies the file from which to read the extended attributes.

Exit Status

Item	Description
0	Successful completion.
Positive integer	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To retrieve all named extended attributes for the file `design.html`, type:

```
getea design.html
```

2. To retrieve the named extended attribute, `Approver`, for the file `design.html`, type:

```
getea -n Approver design.html
```

3. To retrieve just the names of all named extended attributes for the file `design.html`, type:

```
getea -s design.html
```

4. To retrieve all named extended attributes for the symbolic link `design.html`, type:

```
getea -l design.html
```

Location

/usr/sbin

getopt Command

Purpose

Parses command line flags and parameters.

Syntax

getopt *Format Tokens*

Description

The **getopt** command parses a list of tokens using a format that specifies expected flags and arguments. A flag is a single ASCII letter and when followed by a : (colon) is expected to have an argument that may or may not be separated from it by one or more tabs or spaces. You can include multibyte characters in arguments, but not as a flag letter.

The **getopt** command completes processing when it has read all tokens or when it encounters the special token — (double hyphen). The **getopt** command then outputs the processed flags, a — (double hyphen), and any remaining tokens.

If a token fails to match a flag, the **getopt** command writes a message to standard error.

Examples

The **getopt** command can be used in a skeleton shell script to parse options, as in the following example:

```
#!/usr/bin/bsh
# parse command line into arguments
set -- `getopt a:bc $*`
# check result of parsing
if [ $? != 0 ]
then
```

```
        exit 1
fi
while [ $1 != -- ]
do
        case $1 in
        -a)     # set up the -a flag
                AFLG=1
                AARG=$2
                shift;;
        -b)     # set up the -b flag
                BFLG=1;;
        -c)     # set up the -c flag
                CFLG=1;;
        esac
        shift   # next flag
done
shift   # skip --
# now do the work
.
.
.
```

Note: In the C shell, use the following command to run the **getopt** command:

```
set argv=`getopt OptionString $*`
```

In each of the following examples, the **getopt** command would process the flags and arguments in the same way:

- `-a ARG -b -c`
- `-a ARG -bc`
- `-aARG -b -c`
- `-b -c -a ARG`

Files

Item	Description
/usr/bin/getopt	Contains the **getopt** command.

getopts Command

Purpose

Processes command-line arguments and checks for valid options.

Syntax

getopts *OptionString Name* [*Argument* ...]

Description

The **getopts** command is a Korn/POSIX Shell built-in command that retrieves options and option-arguments from a list of parameters. An option begins with a **+** (plus sign) or a **-** (minus sign) followed by a character. An option that does not begin with either a **+** or a **-** ends the *OptionString*. Each time the **getopts** command is invoked, it places the value of the next option in *Name* and the index of the next argument to be processed in the shell variable **OPTIND**. Whenever the shell is invoked, **OPTIND** is initialized to 1. When an option begins with **+**, a **+** is prepended to the value in *Name*.

If a character in *OptionString* is followed by a **:** (colon), that option is expected to have an argument. When an option requires an option-argument, the **getopts** command places it in the variable **OPTARG**.

When an option character not contained in *OptionString* is found, or an option found does not have the required option-argument:

- If *OptionString* does not begin with a **:** (colon),
  - *Name* will be set to a **?** (question mark) character,
  - **OPTARG**. will be unset, and
  - a diagnostic message will be written to standard error.

This condition is considered to be an error detected in the way arguments were presented to the invoking application, but is not an error in the processing of the **getopts** command; a diagnostic message will be written as stated, but the exit status will be zero.

- If *OptionString* begins with a **:** (colon),
  - *Name* will be set to a **?** (question mark) character for an unknown option or to a **:** (colon) character for a missing required option,
  - **OPTARG** will be set to the option character found, and
  - no output will be written to standard error.

Any of the following identifies the end of options: the special option **- -,** finding an argument that does not begin with a **-**, or **+**, or encountering an error.

When the end of options is encountered:

- the **getopts** command will exit with a return value greater than zero,
- **OPTARG** will be set to the index of the first non-option-argument, where the first **- -** argument is considered to be an option-argument if there are no other non-option-arguments appearing before it, or the value $#+1 if there are no non-option-arguments,
- *Name* will be set to a **?** (question mark) character.

Parameters

Item	Description
OptionString	Contains the string of option characters recognized by the **getopts** command. If a character is followed by a colon, that option is expected to have an argument, which should be supplied as a separate argument. The options can be separated from the argument by blanks. The first character in *OptionString* determines how the **getopts** command behaves if an option character is not known or an option-argument is missing. **Note:** The characters question mark and colon must not be used as option characters by an application. The use of other characters that are not alphanumeric produces unspecified results.
Name	Set by the **getopts** command to the option character that was found.
Argument ...	One or more strings separated by white space, checked by the **getopts** command for legal options. If *Argument* is omitted, the positional parameters are used. **Note:** Generally, you won't specify *Argument* as part of the **getopts** command, but it may be helpful when debugging your script.

Exit Status

This command returns the following exit values:

Item	Description
0	An option, specified or unspecified by *OptionString*, was found.

Item	Description
>0	The end of options was encountered or an error occurred.

Examples

1. The following **getopts** command specifies that a, b, and c are valid options, and that options a and c have arguments:

```
getopts a:bc: OPT
```

2. The following **getopts** command specifies that a, b, and c are valid options, that options a and b have arguments, and that **getopts** set the value of 0PT to **?** when it encounters an undefined option on the command line:

```
getopts :a:b:c OPT
```

3. The following script parses and displays it arguments:

```
aflag=
bflag=

while getopts ab: name
do
             case $name in
             a)     aflag=1;;
             b)     bflag=1
                           bval="$OPTARG";;
             ?)     printf "Usage: %s: [-a] [-b value] args\n" $0
                           exit 2;;
            esac
done

if [ ! -z "$aflag" ]; then
            printf "Option -a specified\n"
fi

if [ ! -z "$bflag" ]; then
            printf 'Option -b "%s" specified\n' "$bval"
fi

shift $(($OPTIND -1))
printf "Remaining arguments are: %s\n" "$*"
```

getrunmode Command

Purpose

Displays the mode the system is running in.

Syntax

getrunmode

Description

The **getrunmode** command displays the mode the system is running in. A run mode is either the CONFIGURATION mode or the OPERATIONAL mode.

Examples

To retrieve the run mode, enter:

```
getrunmode
```

Files

Item	Description
/usr/sbin/getrunmode	Contains the **getrunmode** command.

getsecconf Command

Purpose

Displays the system security flags.

Syntax

getsecconf { **-c** | **-o** }

Description

The **getsecconf** command displays the system security flags. When invoked without any options, the **getsecconf** command displays the security flags that pertain to the mode the system is running in.

Flags

Item	Description
-c	Specifies the CONFIGURATION mode.
-o	Specifies the OPERATIONAL mode.

Exit Status

The **getsecconf** command returns the following exit values:

Item	Description
0	Successful execution.
>0	An error occurred.

Examples

1. To display the system security flags in the CONFIGURATION mode, enter:

   ```
   getsecconf –c
   ```

2. To display the system security flags in the OPERATIONAL mode, enter:

   ```
   getsecconf –o
   ```

Files

Item	Description
/usr/sbin/getsecconf	Contains the **getsecconf** command.

getsyslab Command

Purpose

Displays the minimum and maximum labels of the system.

Syntax

getsyslab

Description

The **getsyslab** command is used to display the system minimum and maximum sensitivity label (SL), and minimum and maximum integrity label (TL).

Security

The **getsyslab** command is a privileged command. Successfully running the command requires the following authorization:

Item	Description
aix.mls.system.label.read	Required to list the system labels.

Files Accessed:

Item	Description
Mode	**File**
r	**/etc/security/enc/LabelEncodings**

Example

To display the system labels, enter:

```
getsyslab
```

Files

Item	Description
/usr/sbin/getsyslab	Contains the **getsyslab** command.
/etc/security/enc/LabelEncodings	System default label encodings file.

gettable Command

Purpose

Gets Network Information Center (NIC) format host tables from a host.

Syntax

/usr/sbin/gettable [**-v**] *Host* [*OutFile*]

Description

The **/usr/sbin/gettable** command is used to obtain the NIC standard host tables from a server indicated by the *Host* parameter. The tables, if retrieved, are placed in the file indicated by the *OutFile* parameter.

The **gettable** command opens a Transmission Control Protocol (TCP) connection to the port indicated in the service specification for the *Host* parameter. A request is then made for all names, and the resultant information is placed in the output file.

The **gettable** command is best used in conjunction with the **htable** command, which converts the NIC standard file format to that used by the network library lookup routines.

Flags

Item	Description
-v	Gets just the version number instead of the complete host table and puts the output in *OutFile* or, by default, in a file named **hosts.ver**.

Parameters

Item	Description
Host	Specifies the server that provides the host table information.
OutFile	Specifies the file where you want to place the host table information. If you use the **gettable** command without the **-v** flag, the default file name is **hosts.txt**.

gettrc Command

Purpose

Manages the collection of trace files.

Syntax

gettrc [**-c**] [**-C** *dirname*] [**-m**] [**-M** *dirname*] [**-s**] [**-S** *filename*]

Description

The `gettrc` command is a script that is used in conjunction with the `snap` command. It manages the collection of system trace files, lightweight memory trace (LMT) files, and component trace (CT) files.

Flags

Flag	Description
`-c`	Collects component trace files.
`-C` *dirname*	Collects component trace files from the directory that are specified by *dirname*.
`-m`	Collects memory trace files.
`-M` *dirname*	Collects lightweight memory trace files from the directory that are specified by *dirname*

Flag	Description
`-s`	Collects system trace files.
`-S` *filename*	Collects system trace files from the directory specified by *filename*

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Examples

1. To use `gettrc` in conjunction with the `snap` command to retrieve the different kinds of trace files, enter:

   ```
   snap "gettrc -c -C dirname -m -M dirname -s -S filename"
   ```

   This command returns system trace files, LMT files, and CT files, including those files that are listed in the directory specified by *dirname*.

Location

`/usr/lib/ras/snapscripts/gettrc`

Files

/usr/lib/ras/cpufmt

/etc/trcfmt

getty Command

Purpose

Sets the characteristics of ports.

Syntax

getty [[**-r** | **-u** | **-U**] [**-d**] [**-H** *HeraldString*] [**-M** *motdFile*] [**-N**]] *PortName*

Description

The **getty** command sets and manages terminal lines and ports. The **getty** command is run by the **init** command. The **getty** command is linked to the Terminal State Manager program. The Terminal State Manager program provides combined terminal control and login functions.

You can configure the **getty** command to create your home directory at your login if you do not have a home directory already. The **getty** command calls the **mkuser.sys** command to create the home directory and customize the account. To enable this capability, set the **mkhomeatlogin** attribute of the **usw** stanza in the **/etc/security/login.cfg** file to true.

> **Note:** The **getty** command is not entered on the command line.

When invoked as the **getty** command, the Terminal State Manager program provides the normal port management functions that include:

Item	Description
Bidirectional use	Allows terminal lines to be used to initiate and accept connections.
Line speed	Sets the baud rates for sending and receiving.
Parity	Sets the parity to be even, odd or none.
Delays	Sets the delays for carriage return, tab, new line, and form feed.
Character set mapping	Sets the character set mapping for case, tabs, and carriage control.
Logger Program	Specifies the program used to log the user into the system. If the attribute is set, the Secure Attention Key (SAK) processing is disabled. If the attribute is not set, it defaults to **/usr/sbin/login**. The **logger** attribute is contained within the Object Data Manager (ODM) database.
Character and line erase	Sets the keystroke used for character and line erase.
Echoing mode	Sets the echo to local or remote.

When the **getty** command is invoked, the following steps occur:

1. The port protection is set according to the **owner** and **protection** attributes in the ODM database. If these attributes are not specified, they default to root and 622.
2. The port specified by the *PortName* parameter is opened. If the carrier detection is available on the port, the open does not complete until the carrier is present or another process has lost the carrier with the port.
3. The specified port might be locked. If the **getty** command is run with the **-u** or **-r** flag, it attempts to lock the port. If the port is already locked the command waits until the port is available and then exits. If the **-r** flag was specified, the **getty** command waits for a byte of data to be received on the port before continuing.
4. The terminal attributes are set according to the configuration information for the specified port. Secure Attention Key processing can be enabled at this point depending on the system configuration.
5. The herald message is written to the specified port.
6. The login name is read from the specified port. If a framing error or a break occurs, the **getty** command repeats steps four and five with the next group of configured terminal attributes. This is most commonly used to cycle the baud rates for modems. But any ODM field (except `logmodes` and `runmodes`) may be cycled by entering a list of comma separated values in the ODM database.
7. The terminal modes are reset according to the `runmodes` parameter and the login name. If the login name is terminated by a new line, the **getty** command turns on the carriage-return to new line mapping. If all alphabetic characters are in uppercase, the user is prompted to log in using lowercase characters if possible, and mapping from lowercase to uppercase is turned on.
8. If a program is specified by the logger parameter, it is executed and Secure Attention Key processing is disabled. Otherwise, the Terminal State Manager program performs a standard system login.

   Note: If the Secure Attention Key sequence is typed during a user login, the user is logged into the trusted shell (if the system is configured where that port is trusted and the user is allowed on the trusted path).

Flags

Item	Description
-d	Provides debugging information.

Item	Description
-H *HeraldString*	Specifies an alternate herald message to write on the port to prompt for a login name. The message string must be one word and cannot contain any spaces. This string will take precedence over herald messages defined in the **/etc/security/login.cfg** file. If no string is specified with this option or in the **login.cfg** file, the default herald from the message catalog will be used.
-M *motdFile*	Specifies the path to an alternate message of the day file. If not specified, this value is **/etc/motd** by default.
-N	Causes **getty** to bypass any checking for the process ID in the **/etc/utmp** file. This allows a process other than the lowest login shell to **exec getty**.
-r	Makes the port available for shared (bi-directional) use. If the lock is unsuccessful, the **getty** command waits until the lock is available and then exits. If the lock is successful, the **getty** command waits for a byte of data on the port after locking the port.
-u	Makes the port available for shared (bi-directional) use. If the lock is unsuccessful, the **getty** command waits until the lock is available and then exits.
-U	Same as the **-u** flag, except **getty** does not wait for the lock to be available. It makes the port available regardless of the lock.

Security

Access Control: This program should be installed as a program in the Trusted Computing Base, executable by any user and **setuid** to root.

Example

To enable logging onto tty0, add the following line to the **/etc/inittab** file:

```
tty0:2:respawn: /usr/sbin/getty /dev/tty0
```

This command initializes the port `/dev/tty0` and sets up the characteristics of the port.

Files

Item	Description
/usr/sbin/getty	Contains the **getty** command.
/etc/locks	Contains lock files that prevent multiple uses of communications devices and multiple calls to remote systems.
/usr/sbin/login	The **login** command.
/etc/security/login.cfg	Contains port login configurations.
/etc/motd	Contains the message of the day displayed after login.
/usr/bin/setmaps	The **setmaps** command.
/etc/utmp	Contains information about users logged into the system.

gmvgstat Command

The **gmvgstat** command man page provides reference information for the **gmvgstat** command.

Purpose

Displays GMVG statistics.

Syntax

```
gmvgstat [-h] | [-r] [-t] [-i Interval [-c Count] [-w]]
[gmvg_name . . .]
```

Description

The **gmvgstat** command displays status information for one or more GMVGs including:

- Number of Physical Volumes
- Number of Remote Physical Volumes
- Total Number of Volumes (PVs and RPVs)
- Number of Stale Volumes
- Total Number of Physical Partitions (PPs)
- Number of Stale PPs
- Percentage GMVG is synchronized

The **gmvgstat** command can optionally be invoked in monitor mode by specifying the -i and -c flags.

If one or more GMVG names are supplied on the command line, then the **gmvgstat** command verifies that each listed GMVG name is a valid, available, online GMVG. In monitor mode, the user-supplied list of GMVGs is verified during each loop.

If no GMVG names are supplied on the command line the **gmvgstat** command reports information on all valid, available, online GMVGs. In monitor mode the list of GMVGs to report on is regenerated during each loop.

Flags

Table 6. gmvgstat command flags

Flag	Description
-h	Display command syntax and help.
-r	Include information for each individual RPV Client associated with the displayed GMVGs.
-t	Display header with date and time.

Table 6. gmvgstat command flags (continued)	
Flag	**Description**
-i Interval	Automatically redisplay status every <Interval> seconds. The value of the <Interval> parameter must be an integer greater than zero and less than or equal to 3600. If the <Interval> parameter is not specified, then display the status information once. The -i interval is the time, in seconds, between each successive gathering and display of GMVG statistics in monitor mode. This interval is not a precise measure of the elapsed time between each successive updated display. The gmvgstat command obtains some of the information it displays by calling other commands and has no control over the amount of time these commands take to complete their processing. Larger numbers of GMVGs will result in the gmvgtstat command taking longer to gather information and will elongate the time between successive displays in monitor mode. In some cases, an underlying command may take excessively long to complete and will result in the gmvgstat command taking much longer than the -i interval between displays.
-c Count	Redisplay information at the indicated interval <Count> times. The value of the <Count> parameter must be an integer greater than zero and less than or equal to 999999. If the <Interval> parameter is specified, but the <Count> parameter is not, then redisplay indefinitely.
-w	Clear the screen between each redisplay.

Operands

Table 7. Operand field	
Field	**Value**
gmvg_name	Name of one or more GMVGs for which to display information. If no GMVG names are specified, then information for all valid, available, online GMVGs is displayed.

Exit Status

Table 8. Exit status	
Value	**Description**
0	No errors.
>0	An error occurred.

Examples

1. To display statistical information for all GMVGs, enter:

   ```
   gmvgstat
   ```

2. To display statistical information for the GMVG named red_gmvg7, enter:

   ```
   gmvgstat red_gmvg7
   ```

3. To display statistical information for the GMVG named red_gmvg7 with statistics for all the RPVs associated with that volume group, enter:

   ```
   gmvgstat -r red_gmvg7
   ```

4. To display detailed information for GMVG red_gmvg7 that is automatically redisplayed every 10 seconds, enter:

```
gmvgstat red_gmvg7 -i 10
```

5. To display detailed information for GMVG red_gmvg7 that is automatically redisplayed every 10 seconds for 20 intervals and clears the screen between each redisplay, enter:

```
gmvgstat red_gmvg7 -i 10 -c 20 -w
```

Files

/usr/sbin/gmvgstat contains the gmvgstat command.

gprof Command

Purpose

Displays call graph profile data.

Syntax

/usr/ccs/bin/gprof [**-b**] [**-c** [*filename*]] [**-e** *Name*] [**-E** *Name*] [**-f** *Name*] [**-g** *filename*] [**-i** *filename*] [**-p** *filename*] [**-F** *Name*] [**-L** *PathName*] [**-s**] [**-x** [*filename*]] [**-z**] [**a.out** [**gmon.out** ...]]

Description

The **gprof** command produces an execution profile of C, FORTRAN, or COBOL programs. The effect of called routines is incorporated into the profile of each caller. The **gprof** command is useful in identifying how a program consumes processor resource. To find out which functions (routines) in the program are using the processor, you can profile the program with the **gprof** command.

The profile data is taken from the call graph profile file (**gmon.out** by default) created by programs that are compiled with the **cc** command by using the **-pg** option. The **-pg** option also links in versions of library routines that are compiled for profiling, and reads the symbol table in the named object file (**a.out** by default), correlating it with the call graph profile file. If more than one profile file is specified, the **gprof** command output shows the sum of the profile information in the specified profile files.

The **-pg** option causes the compiler to insert a call to the **mcount** subroutine into the object code that is generated for each recompiled function of your program. During program execution, each time a parent calls a child function the child calls the **mcount** subroutine to increment a distinct counter for that parent-child pair. Programs that are not recompiled with the **-pg** option do not have the **mcount** subroutine, and therefore keep no record of who called them.

Note: Symbols from C++ object file names get changed before they are used.

The GPROF environment variable can be used to set different options for profiling. The syntax of this environment variable is defined as follows:

```
GPROF = profile:<profile-type>,scale:<scaling-factor>,file:<file-type>,filename:<filename>
```

where:

- <profile-type> describes what type of profiling is to be performed; it can be either process or thread. Type 'process' indicates that profiling granularity is at process level, 'thread' indicates that profiling granularity is at thread level.
- <scaling-factor> describes how much memory is required to be allocated for call graph profile, by default the scaling factor is 2 for process level profiling and 8 for thread level profiling. A scaling factor of 2 indicates that a memory of half of the process size is allocated for every process or thread, scaling factor of 8 indicates that a memory of one eighth of the process size is allocated for every process of thread. This memory is the buffer area to store the call graph information.

- <file-type> describes what type of **gmon.out** file is required, a value of **multi** indicates that one **gmon.out** file per process is required, a value of multithread indicates that one **gmon.out** file per thread is required. If an application is profiled with the **-pg** option, and it forks, then specifying the file type as **multi** generates a **gmon.out** file for the parent process and another for the child process. The naming convention for the generated **gmon.out** files is as follows:
  - For multi file type: <prefix>-processname-pid.out
  - For multithread file type: <prefix>-processname-pid-Pthread<threadid>.out

  The <prefix> is by default **gmon**. You can define your own prefix by using the *filename* parameter of the GPROF environment variable.
- <filename> describes the prefix that requires to be used for the generated **gmon.out** files. By default, the prefix is **gmon**.

Note: Specifying `profile:thread` generates a format **gmon.out** file that can be read only by AIX 5.3 **gprof** command. If you want an old format **gmon.out** file and still want to specify **profile:thread**, then you must specify **file:multithread**. It generates an old format **gmon.out** file per thread. Hence, if your application has 2 threads, then 2 **gmon.out** files are generated, one per thread, by using the naming convention. You cannot enable thread level profiling by compiling an application with the **-pg** flag in AIX 5.2 or earlier and running it in AIX 5.3. To enable thread level profiling, you must compile that application with the **-pg** flag in AIX 5.3 and later.

The **gprof** command produces three items:

1. First, a flat profile is produced similar to the profile that is provided by the **prof** command. This listing gives total execution times and call counts for each of the functions in the program, which is sorted by decreasing time. The times are then propagated along the edges of the call graph. Cycles are discovered, and calls into a cycle are made to share the time of the cycle.
2. A second listing shows the functions that are sorted according to the time they represent, including the time of their call-graph descendants. Below each function entry are its (direct) call-graph children, with an indication of how their times are propagated to this function. A similar display above the function shows how the time of the function and the time of its descendants are propagated to its (direct) call-graph parents.
3. Cycles are also shown, with an entry for the cycle as a whole and a listing of the members of the cycle and their contributions to the time and call counts of the cycle.

Note: If the input to the **gprof** command contains thread level profiling data (format **gmon.out** file), then the **gprof** command produces the specified three items for every thread, starting with a cumulative report, followed by per thread reported (sorted in the ascending order of thread IDs).

The **grpof** command can also be used to analyze the execution profile of a program on a remote machine. It can be done by running the **gprof** command with the **-c** option on the call graph profile file (**gmon.out** by default) to generate a file (**gprof.remote** by default), which can then be processed on a remote machine. If a call graph profile file other than **gmon.out** is to be used, the call graph profile file name must be specified after **-c** *Filename* and the executable name. *Filename* must be specified if the **GPROF** environment variable's **file** attribute is set to **multi**; multiple **gmon.out** files are created, with one **gmon.out** file for each PID when the running program forks. The **-x** option can be used on the remote machine to process the **gprof.remote** (by default) file to generate profile reports.

Profiling with the fork and exec subroutines

Profiling by using the **gprof** command is problematic if your program runs the **fork** or **exec** subroutine on multiple, concurrent processes. Profiling is an attribute of the environment of each process, so if you are profiling a process that forks a new process, the child is also profiled. However, both processes write a **gmon.out** file in the directory from which you run the parent process, overwriting one of them. The **tprof** command is recommended for multiple-process profiling. You can use **file:multi** to avoid deleting the **gmon.out** file of the parent process, **file:multi** by using the AIX naming convention to generate the **gmon.out** files, hence the child processes **gmon.out** file does not have the same name as the parent, which avoids overwrites.

Profiling without source code

If you do not have source for your program, you can profile by using the **gprof** command without recompiling. You must, however, be able to relink your program modules with the appropriate compiler command (for example, **cc** for C). If you do not recompile, you do not get call frequency counts, although the flat profile is still useful without them. As an added benefit, your program runs almost as fast as it usually does. The following explains how to profile:

```
cc -c dhry.c         # Create dhry.o without call counting code.
cc -pg dhry.o -L/lib -L/usr/lib -o dhryfast
                     # Re-link (and avoid -pg libraries).
dhryfast             # Create gmon.out without call counts.
gprof >dhryfast.out  # You get an error message about no call counts
                     #  -- ignore it.
```

A result of running without call counts is that some quickly running functions (which you know had to be called) do not appear in the listing. Although nonintuitive, this result is normal for the **gprof** command. The **gprof** command lists only functions that were either called at least once, or which registered at least one clock tick. Even though they ran, quickly running functions often receive no clock ticks. Since call-counting was suspended, these small functions are not listed at all. (You can get call counts for the runtime routines by omitting the **-L** options on the **cc -pg** command line.)

Using less real memory

Profiling with the **gprof** command can cause programs to page excessively since the **-pg** option dedicates pinned real-memory buffer space equal to one-half the size of your program text. Excessive paging does not affect the data that is generated by profiling, since profiled programs do not generate ticks when waiting on I/O but only when using the processor. If the time delay caused by excessive paging is unacceptable, it is recommended to use the**tprof** command.

Flags

Item	Description
-b	Suppresses the printing of a description of each field in the profile.
-c *Filename*	Creates a file that contains the information that is needed for remote processing of profiling information. Do not use the **-c** flag in combination with other flags.
-E *Name*	Suppresses the printing of the graph profile entry for routine *Name* and its descendants, similar to the **-e** flag, but excludes the time that is spent by routine *Name* and its descendants from the total and percentage time computations. (**-E** *MonitorCount* **-E** *MonitorCleanup* is the default.)
-e *Name*	Suppresses the printing of the graph profile entry for routine *Name* and all its descendants (unless they have other ancestors that are not suppressed). More than one **-e** flag can be given. Only one routine can be specified with each **-e** flag.
-F *Name*	Prints the graph profile entry of the routine *Name* and its descendants similar to the **-f** flag, but uses only the times of the printed routines in total time and percentage computations. More than one **-F** flag can be given. Only one routine can be specified with each **-F** flag. The **-F** flag overrides the **-E** flag.
-f *Name*	Prints the graph profile entry of the specified routine *Name* and its descendants. More than one **-f** flag can be given. Only one routine can be specified with each **-f** flag.
-g *Filename*	Writes call graph information to the specified output *filename*. It also suppresses the profile information unless the **-p** flag is used.
-i *Ffilename*	Writes the routine index table to the specified output *filename*. If this flag is not used, the index table goes either at the end of the standard output, or at the bottom of the filename specified with the **-p** and **-g** flags.
-L *PathName*	Uses an alternative path name for locating shared objects.

Item	Description
-p *Filename*	Writes flat profile information to the specified output file name. It also suppresses the call graph information unless the **-g** flag is used.
-s	Produces the **gmon.sum** profile file, which represents the sum of the profile information in all the specified profile files. This summary profile file might be given to subsequent executions of the **gprof** command (by using the **-s** flag) to accumulate profile data across several runs of an **a.out** file.
-x *Filename*	Retrieves information from *Filename* (a file that is created with the **-c** option) to generate profile reports. If *Filename* is not specified, the **gprof** command searches for the default **gprof.remote** file.
-z	Displays routines that have zero usage (as indicated by call counts and accumulated time).

Examples

1. To obtain profiled output, enter the following command:

```
gprof
```

2. To get profiling output from a command run earlier and possibly moved, enter the following command:

```
gprof -L/home/score/lib runfile runfile.gmon
```

   This example uses the **runfile.gmon** file for sample data and the **runfile** file for local symbols, and checks the **/u/score/lib** file for loadable objects.

3. To profile the sample program **dhry.c**:

   a. Recompile the application program with the **cc -pg** command, as follows:

```
cc -pg dhry.c -o dhry # Re-compile to produce gprof output.
```

   b. Run the recompiled program. A file named `gmon.out` is created in the current working directory (not the directory in which the program executable file is located).

```
dhry    # Execute program to generate ./gmon.out file.
```

   c. Run the **gprof** command in the directory with the `gmon.out` file to produce the call graph and flat profile reports.

```
gprof >gprof.out     # Name the report whatever you like
vi gprof.out         # Read flat profile first.
```

   d. To generated thread level profiling granularity, export the GPROF environment variable as follows, and run the application, enter the following command:

```
export GPROF=profile:thread
dhry   # Execute program to generate ./gmon.out file which has thread level granularity
```

   e. To generate per process **gmon.out** file with a prefix of `mygmon`, enter the following command:

```
export GPROF=file:multi,filename:mygom
dhry  # Execute program to generate ./gmon-dhry-2468.out
```

   f. To generate per thread **gmon.out** file, with a scaling factor of 10, with a file name prefixed as `tgmon`, enter the following command:

```
export GPROF=profile:thread,file:multithread,scale:10,filename:tgmon
dhry # Execute program to generate ./tgmon-dhry-2468-Pthread215.out
```

g. To see only flat profile report from the `gmon-dhry-2468.out`, enter the following command:

```
gprof -p fprofile.out ./dhry ./gmon-dhry-2468.out
```

h. To see only call graph profile report from the `gmon-dhry-2468.out`, enter the following command:

```
gprof -g callgraph.out ./dhry ./gmon-dhry-2468.out
```

4. To use the remote processing feature of **gprof** command:

a. Recompile the application program with **cc -pg** command:

```
cc -pg thread.c -o thread  -lpthread
```

b. Enable thread level profiling granularity and use a different name for **gmon.out**:

```
export GPROF=profile:thread,filename:mygmon
```

c. Run the recompiled program. A file named **mygmon.out** is created in the current working directory (not the directory in which the program executable file is located):

```
thread    # Execute program to generate mygmon.out file.
```

d. Use the **-c** flag to generate the **my.remote** file, which can then be taken to a remote machine for processing:

```
gprof -c my.remote thread mygmon.out
```

e. On a remote machine, use the **-x** flag to extract information from the **my.remote** file:

```
gprof -x my.remote
```

Throughout this description of the **gprof** command, most of the examples use the C program **dhry.c**. However, the discussion and examples apply equally to FORTRAN or COBOL modules by substituting the appropriate compiler name in place of the C compiler, **cc**, and the word *subroutine* for the word *function*. For example, the following commands show how to profile a FORTRAN program named `matrix.f`:

```
xlf -pg matrix.f -o matrix # FORTRAN compile of matrix.f program
matrix                     # Execute with gprof profiling,
                           #   generating gmon.out file
gprof > matrix.out         # Generate profile reports in
                           #   matrix.out from gmon.out
vi matrix.out              # Read flat profile first.
```

Files

Item	Description
a.out	Name list and text space
gmon.out	Dynamic call graph and profile
gmon.sum	Summarized dynamic call graph and profile
gprof.remote	File for remote profiling
/usr/ucb/gprof	Contains the **gprof** command.
/usr/ccs/bin/gprof	Contains the **gprof** command

grap Command

Purpose

Typesets graphs to be processed by the **pic** command.

Syntax

grap [**-l**] [**-T** *Name*] [**—**] [*File* ...]

Description

The **grap** command processes grap language input files and generates input to the **pic** command. The grap language is a language for typesetting graphs. A typical command line is:

```
grap  File  |  pic  |  troff  |  Typesetter
```

Graphs are surrounded by the **.G1** and **.G2 troff** command requests. Data enclosed by these requests are scaled and plotted, with tick marks automatically supplied. Commands exist to modify the frame, add labels, override the default ticks, change the plotting style, define coordinate ranges and transformations, and include data from files. In addition, the **grap** command provides the same loops, conditionals, and macroprocessing as the **pic** command.

Grap language files contain grap programs. A grap program is written in the form:

```
.G1
grap Statement
grap Statement
grap Statement
.G2
```

Parameter

Item	Description
File	Specifies grap language files (grap programs) to be processed by the **grap** command for input to the **pic** command.

grap Statements Summary

Following is a summary of the grap statements you can use to create a grap program:

Item	Description
frame	Defines the frame that surrounds the graph. The syntax is: `frame [ht Expression] [wid Expression] [[Side] LineDescription]`

The attributes are defined as follows:

- *Side*: `top`, `bot`, `left`, `right`
- *LineDescription*: `solid`, `invis`, `dotted [Expression]`, `dashed [Expression]`

Height defaults to 2 inches, width defaults to 3 inches, sides default to solid. If `side` is omitted, the *linedesc* applies to the entire frame.

Item	Description
label	Places a label on a specified side of the graph. The syntax is:

```
label Side StringList ... Shift
```

The attributes are defined as follows:

- *Shift*: `left`, `right`, up, or down *expression*
- *StringList*: str ... `rjust`, `ljust`, `above, below [size (+)Expression] ...`
- *String*: `"..."`

Item	Description
coord	Defines an overriding system. The syntax is:

```
coord [Name] [x Expression,Expression] [y  Expression,Expression] [[log x] [log y]
[log log]]
```

ticks	Places tick marks on one side of the frame. The syntax is:

```
ticks side [[in] [out] [Expression]] [Shift] [TickLocations]
```

The attributes are defined as follows:

- *Shift*: `left`, `right`, up, `down Expression`
- *TickLocations*: at [*Name*] *Expression* [*String*], *Expression* [*String*], ... from [*Name*] *Expression* to *Expression* [by [*Operation*] *Expression*] *String*

If no ticks are specified, they will be provided automatically; `ticks off` suppresses automatic ticks.

Item	Description
grid	Produces grid lines along (that is, perpendicular to) the named side. The syntax is:

```
grid Side [LineDescription] [Shift] [TickLocations]
```

Grids are labeled by the same mechanism as ticks.

Item	Description
plot	Places text at a point. The syntax is:

```
StartList at Point plot Expression [Start] at Point
```

The attributes are defined as follows:

- *StringList*: str ... `rjust`, `ljust`, `above, below [size +)Expression] ...`
- *Point*: `[Name] Expression Expression`

Item	Description
line	Draws a line or arrow from one point to another. The syntax is:

```
{line | arrow} from Point to Point [LineDescription]
```

The attributes linedesc are defined as follows:

- *Point*: [*Name*] *Expression Expression*

- *LineDescription*: `solid`, `invis`, `dotted [Expression], dashed Expression]`

Item	Description
circle	Draws a circle. The syntax is: `circle at Point [radius Expression]`

The radius is in inches; the default size is small.

Item	Description
draw	Defines a sequence of lines. The syntax is: `draw [Name] at Point[LineDescription]`
next	Continues a sequence. The syntax is: `next [Name] at Point [LineDescription]`
new	Starts a new sequence. The syntax is: `new [Name] at Point [LineDescription]`
numberlist	Creates a line from a given set of numbers. The numbers are treated as points *x*, *y*1, *y*2, and so on; and plotted at the single *x* value. The syntax is: `number x, y1, y2 ...`
for	Creates a loop. The syntax is: `for Variable {from | =} Expression to Expression \` `[by [arithmetic or multiplicative operator] Expression] do X Anything X` X is any single character that does not appear in the string. If X is a left brace {, then the string may contain internally balanced braces followed by a right brace}. The text `Anything` is repeated as the `Variable` takes on values from the first `Expression` to the second `Expression`.
if	Creates a conditional evaluation. The syntax is: `if Expression then X Anything X [else X Anything X]`
define	Provides the same macroprocessor that Priority Interrupt Controller (PIC) does. The syntax is: `define MacroName X Anything X`
copy	Copies a file; includes the current contents of the file. The syntax is: `copy Filename`
copy-thru	Copies the file through the macro. `copy Filename thru MacroName` Each number or quoted string is treated as an argument. Copying continues until end of file or the next `.G2`. The optional clause `until String` causes copying to stop when a line whose first field is `String` occurs. The following statement copies subsequent lines through the macro: `copy thru MacroName` In all cases, you can specify the macro by inline rather than by name: `copy thru x MacroBody x`
sh	Passes text through to the UNIX shell. The syntax is: `sh x Anything x` The variable `Anything` is scanned for macros. The **pid** macro is built-in. It is a string consisting of the process identification number; you can use it to generate unique file names.

Item	Description
pic	Passes text through to **pic** with the `pic` removed. Variables and macros are not evaluated. Lines beginning with a period (that are not numbers) are passed through literally, under the assumption that they are **troff** commands.
graph	Defines a new graph named *Picname*, and resets all coordinate systems. The syntax is: `graph Picname [pic-text]` If **graph** commands are used in a grap program, the statement after the `.G1` must be a **graph** command. You can use the `pic-text` to position this graph relative to previous graphs by referring to their Frames as in the following example. `graph First` `  ...` `graph Second with .Frames.w at First.Frame.e + [0.1,0]` Macros and expressions in `pic-text` are not evaluated. `Picnames` must begin with a capital letter according to **pic** syntax.
print	Writes on `stderr` as grap processes its input. This statement can be helpful in debugging. The syntax is: `print [Expression | String]`

grap Language Conventions

The following conventions apply:

- The # (pound sign) introduces a comment. The comment ends automatically at the end of a line.
- Statements that continue for more than one line must be preceded by a \ (backslash character) at the beginning of each new line.
- Multiple statements appearing on one line must be separated by semicolons.
- The grap language ignores blank lines.
- Predefined strings include `bullet`, `plus`, `box`, `star`, `dot`, `times`, `htick`, `vtick`, `square`, and `delta`.
- Built-in functions available in grap include `log` (base 10), `exp` (base 10), `int`, `sin`, `cos`, `atan2`, `sqrt`, `min`, `max`, and `rand`.

Flags

Item	Description
-l	Stops the **grap** command from looking for the **/usr/lib/dwb/grap.defines** library file of macro definitions.
-T*Name*	Specifies the value of the *Name* variable as the **grap** command output device. The default value is **-Tibm3816**.
- -	(Double dash) Indicates the end of flags.

File

Item	Description
/usr/lib/dwb/grap.defines	Contains definitions of standard plotting characters.

greek Command

Purpose

Converts English-language output from a Teletype Model 37 workstation to output for other workstations.

Syntax

greek [**-T** *Name*]

Description

The **greek** command reinterprets the Teletype Model 37 character set, including reverse and half-line motions, for display on other workstations. It simulates special characters, when possible, by overstriking. The **greek** command reads standard input and writes to standard output.

Flags

Item	Description
-T*Name*	Uses the specified workstation name. If you omit the **-T** flag, the **greek** command attempts to use the workstation specified in the **$TERM** environment variable. The value of the *Name* variable can be any one of the following:
300	DASI 300
300-12	DASI 300 in 12-pitch
300s	DASI 300s
300s-12	DASI 300s, in 12-pitch
450	DASI 450
450-12	DASI 450, in 12-pitch
2621	Hewlett-Packard 2621, 2640, and 2645
2640	Hewlett-Packard 2621, 2640, and 2645
2645	Hewlett-Packard 2621, 2640, and 2645
4014	Tektronix 4014
hp	Hewlett-Packard 2621, 2640, and 2645
tek	Tektronix 4014.

Environment Variables

Item	Description
$TERM	Specifies a workstation name.

grep Command

Purpose

Searches for a pattern in a file.

Syntax

grep [**-E** | **-F**] [**-i**] [**-h**] [**-H**] [**-L**] [**-r** | **-R**] [**-s**][**-u**] [**-v**] [**-w**] [**-x**] [**-y**] [[[**-b**] [**-n**]] | [**-c** | **-l** | **-q**]] [**-p** [*Separator*]] { [**-e** *PatternList* ...] [**-f** *PatternFile* ...] | *PatternList* ... } [*File* ...]

Description

The **grep** command searches for the pattern specified by the *Pattern* parameter and writes each matching line to standard output. The patterns are limited regular expressions in the style of the **ed** or **egrep** command. The **grep** command uses a compact non-deterministic algorithm.

The **grep** command displays the name of the file containing the matched line if you specify more than one name in the *File* parameter. Characters with special meaning to the shell ($, *, [, |, ^, (,), \ ) must be in quotation marks when they appear in the *Pattern* parameter. When the *Pattern* parameter is not a

simple string, you usually must enclose the entire pattern in single quotation marks. In an expression such as `[a-z]`, the - (minus sign) cml specifies a range, according to the current collating sequence. A collating sequence may define equivalence classes for use in character ranges. If no files are specified, **grep** assumes standard input.

Notes:

1. Do not run the **grep** command on a special file because it produces unpredictable results. Input lines should not contain the NULL character.
2. Input files should end with the newline character.
3. The newline character will not be matched by the regular expressions.
4. Although some flags can be specified simultaneously, some flags override others. For example, the **-l** option takes precedence over all other flags. And if you specify both the **-E** and **-F** flags, the last one specified takes priority.

Flags

Item	Description
-b	Precedes each line by the block number on which it was found. Use this flag to help find disk block numbers by context. The **-b** flag cannot be used with input from stdin or pipes.
-c	Displays only a count of matching lines.
-E	Treats each pattern specified as an extended regular expression (ERE). A NULL value for the ERE matches every line. **Note:** The **grep** command with the **-E** flag is the same as the **egrep** command, except that error and usage messages are different and the **-s** flag functions differently.
-e *PatternList*	Specifies one or more search patterns. This works like a simple pattern but is useful when the pattern begins with a - (minus). Patterns should be separated by a new-line character. A NULL pattern can be specified by two adjacent new-line characters or a quotation mark followed by a new-line character ("\n). Each pattern is treated like a basic regular expression (BRE) unless the **-E** or **-F** flag is also specified. Multiple **-e** and **-f** flags are accepted by **grep**. All of the specified patterns are used when matching lines, but the order of evaluation is unspecified.
-F	Treats each specified pattern as a string instead of a regular expression. A NULL string matches every line. **Note:** The **grep** command with the **-F** flag is the same as the **fgrep** command, except that error and usage messages are different and the **-s** flag functions differently.
-f *PatternFile*	Specifies a file containing search patterns. Each pattern should be separated by a new-line character, and an empty line is considered a NULL pattern. Each pattern is treated like a basic regular expression (BRE), unless the **-E** or **-F** flag is also specified.
-h	Prevents the name of the file containing the matching line from being appended to that line. Suppresses file names when multiple files are specified.
-H	If the **-r** or **-R** option is specified and a symbolic link referencing a file of type directory is specified on the command line, **grep** will search the files of the directory referenced by the symbolic link and all the files in the file hierarchy below it.

Item	Description
-i	Ignores the case (uppercase or lowercase) of letters when making comparisons.

Item	Description
-l	Lists just the names of files (once) which contain matching lines. Each file name is separated by a new-line character. If standard input is searched, a path name of `(StandardInput)` is returned. The **-l** flag with any combination of the **-c** and **-n** flags behaves like the **-l** flag only.
-L	If the **-r** or **-R** option is specified and a symbolic link referencing a file of type directory is specified on the command line or encountered during the traversal of a file hierarchy, **grep** shall search the files of the directory referenced by the symbolic link and all the files in the file hierarchy below it. If both **-H** and **-L** are specified, the last option specified on the command line takes effect.
-n	Precedes each line with the relative line number in the file. Each file starts at line 1, and the line counter is reset for each file processed.
-p[*Separator*]	Displays the entire paragraph containing matched lines. Paragraphs are delimited by paragraph separators, as specified by the *Separator* parameter, which are patterns in the same form as the search pattern. Lines containing the paragraph separators are used only as separators; they are never included in the output. The default paragraph separator is a blank line.
-q	Suppresses all writing to standard output, regardless of matching lines. Exits with a zero status if an input line is selected. The **-q** flag with any combination of the **-c**, **-l** and **-n** flags behaves like the **-q** flag only.
-r	Searches directories recursively. By default, links to directories are followed.
-R	Searches directories recursively. By default, links to directories are not followed.
-s	Suppresses error messages ordinarily written for nonexistent or unreadable files. Other error messages are not suppressed.
-u	Causes output to be unbuffered.
-v	Displays all lines not matching the specified pattern.
-w	Does a word search.
-x	Displays lines that match the specified pattern exactly with no additional characters.
-y	Ignores the case of letters when making comparisons.
PatternList	Specifies one or more patterns to be used during the search. The patterns are treated as if they were specified using the **-e** flag.
File	Specifies a name of a file to be searched for patterns. If no *File* variable is given, the standard input is used.

Exit Status

This command returns the following exit values:

Item	Description
0	A match was found.
1	No match was found.
>1	A syntax error was found or a file was inaccessible (even if matches were found).

Examples

1. To use a pattern that contains some of the pattern-matching characters *, ^, ?, [,], \(, \), \{, and \}, enter:

   ```
   grep  "^[a-zA-Z]"  pgm.s
   ```

   This displays every line in `pgm.s` whose first character is a letter.
2. To display all lines that do not match a pattern, enter:

   ```
   grep   -v   "^#" pgm.s
   ```

   This displays every line in `pgm.s` whose first character is not a # (pound sign).
3. To display all lines in the `file1` file that match either the `abc` or `xyz` string, enter:

   ```
   grep   -E   "abc|xyz"  file1
   ```
4. To search for a $ (dollar sign) in the file named `test2`, enter:

   ```
   grep \\$ test2
   ```

   The \\ (double backslash) characters are necessary in order to force the shell to pass a \$ (single backslash, dollar sign) to the **grep** command. The \ (single backslash) character tells the **grep** command to treat the following character (in this example the $) as a literal character rather than an expression character. Use the **fgrep** command to avoid the necessity of using escape characters such as the backslash.
5. To search recursively through **/tmp** to find files which have the word `IBM` without recursing through links pointing to directories, type:

   ```
   grep –R IBM /tmp
   ```

   OR

   ```
   grep –r -H IBM /tmp
   ```
6. To search recursively through **/tmp** to find files which have the word `IBM` and recurse through links as well, type:

   ```
   grep –r IBM /tmp
   ```

   OR

   ```
   grep -R -L IBM /tmp
   ```

Files

Item	Description
/usr/bin/grep	Contains the **grep** command.

groups Command

Purpose

Displays group membership.

Syntax

groups [*User...*]

Description

By default, the **groups** command writes the group membership information of the current process to the standard output. If multiple users are specified as command parameters, the group membership for each user is displayed from the database.

The **groups** command will continue its operation with the next user in the parameter list after issuing a warning message if the user given is not found in the user database.

Security

Access Control: This program should be installed as a normal user program in the Trusted Computing Base.

Examples

To display the group membership of users listed in the parameter list, enter:

```
$ groups sys root lp adm
sys : sys
root : system bin sys security cron audit lp
lp : lp printq
adm : adm
```

Files

Item	Description
/usr/bin/groups	Contains the **groups** command
/usr/ucb/groups	Symbolic link to the **groups** command
/etc/group	Group file; contains group IDs
/etc/ogroup	Previous version of the group file
/etc/passwd	Password file; contains user IDs
/etc/opasswd	Previous version of the password file.

grpck Command

Purpose

Verifies the correctness of a group definition. This document describes both the AIX **grpck** command and the System V **grpck** command.

Syntax

grpck { **-n** | **-p** | **-t** | **-y** } { **ALL** | *Group* ... }

Description

The **grpck** command verifies the correctness of the group definitions in the user database files by checking the definitions for all the groups or for the groups that are specified by the *Group* parameter. If more than one group is specified, there must be a space between the groups.

Note: This command writes its messages to `stderr`.

You must select a flag to indicate whether the system must try to fix erroneous attributes. The following attributes are checked:

Item	Description
name	Checks the uniqueness and composition of the group name. The group name must be a unique string of 8 bytes or less. It cannot begin with a + (plus sign), a : (colon), a - (minus sign), or a ~ (tilde). It cannot contain a : (colon) in the string and cannot be the `ALL` or `default` keywords. No system fix is possible.
groupID	Checks the uniqueness and composition of the group ID. The ID must not be null and must consist of decimal digits only. No system fix is possible.
users	Checks the existence of the users that are listed in the group database files. If you indicate that the system must fix errors, it deletes all the users that are not found in the user database files.
adms	Checks the existence of the users that are listed as group administrators in the group database files. If you indicate that the system must fix errors, it deletes all the administrators that are not found in the user database files.
admin	Checks for a valid admin attribute for each group in the `/etc/security/group` file. No system fix is available.

Generally, the **sysck** command calls the **grpck** command as part of the verification of a trusted-system installation. In addition, the root user or a member of the security group can enter the command.

The **grpck** command checks to see whether the database management security files (`/etc/passwd.nm.idx`, `/etc/passwd.id.idx`, `/etc/security/passwd.idx`, and `/etc/security/lastlog.idx`) files are up-to-date or newer than the corresponding system security files. It is acceptable for `/etc/security/lastlog.idx` to be not newer than `/etc/security/lastlog`. If the database management security files are out-of-date, a warning message appears indicating that the root user must run the **mkpasswd** command.

Flags

Item	Description
-n	Reports errors but does not fix them.
-p	Fixes errors but does not report them.
-t	Reports errors and asks if they must be fixed.
-y	Fixes errors and reports them.

Security

Access Control: This command must grant execute (x) access to the root user and members of the security group. The **setuid** command for the root user must have the **trusted computing base** attribute.

Files Accessed:

Mode	File
r	`/etc/passwd`

Mode	File
r	/etc/security/user
rw	/etc/security/group
rw	/etc/group

Auditing Events:

Event	Information
GROUP_User	user, groups, attribute | error, status
GROUP_Adms	user, groups, attribute | error, status

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To verify that all the group members and administrators exist in the user database, and to report all the errors but not fix them, enter the following command:

   ```
   grpck -n ALL
   ```

2. To verify that all the group members and administrators exist in the user database, and to fix all the errors but not report them, enter the following command:

   ```
   grpck -p ALL
   ```

3. To verify the uniqueness of the group name and group ID defined for the `install` group, enter the following command:

   ```
   grpck -n install
   ```

   Or,

   ```
   grpck -t install
   ```

   Or,

   ```
   grpck -y install
   ```

   The **grpck** command does not correct the group names and IDs. Therefore, the **-n**, **-t**, and **-y** flags report problems with group names and group IDs, but do not correct them.

Files

Item	Description
/usr/sbin/grpck	Contains the **grpck** command.
/etc/passwd	Contains the basic attributes of users.
/etc/security/user	Contains the extended attributes of users.
/etc/group	Contains the basic attributes of groups.
/etc/security/group	Contains the extended attributes of groups.

System V grpck command

Syntax

/usr/sysv/bin/grpck

Description

The **/usr/sysv/bin/grpck** command verifies the correctness of the group definitions in the user database files by checking the definitions for all the groups. This **/usr/sysv/bin/grpck** command is a System V version of the existing **grpck** command in /usr/sbin/. This command calls the **/usr/sbin/grpck** command with the **-n** flag and **ALL** options.

Exit Status

0
Successful completion.

>0
An error occurred.

Examples

1. To verify that all the group members and administrators exist in the user database, and have any errors that are reported (but not fixed), enter the following command:

```
/usr/sysv/bin/grpck
```

Files

/usr/sysv/bin/grpck
Contains the System V version of the **grpck** command.

grpsvcsctrl Command

Purpose

Starts the group services subsystems.

Syntax

grpsvcsctrl { -a | -s | -k | -d | -c | -u | -t | -o | -h }

Description

The grpsvcsctrl command starts the group services subsystems. This control script controls the operation of the subsystems that are required for group services. These subsystems are under the control of the system resource controller (SRC) and belong to a subsystem group called grpsvcs. A daemon is associated with each subsystem. From an operational point of view, the group services subsystem group is organized as follows:

Subsystem
group services

Subsystem group
grpsvcs

SRC subsystem
grpsvcs — associated with the hagsd daemon. The subsystem name on the nodes is grpsvcs. The grpsvcs subsystem on each node is associated with the cluster to which the node belongs.

Daemon
hagsd — provides the majority of the group services functions.

The grpsvcsctrl script is not normally run from the command line. It is normally called by the startup command during installation of the cluster.

The grpsvcsctrl script provides a variety of controls for operating the group services subsystems:

- Adding, starting, stopping, deleting, and cleaning up the subsystems
- Turning tracing on and off

Before performing any of these functions, the script obtains the current cluster name.

Adding the subsystem: When the -a flag is specified, the control script uses the mkssys command to add the group services subsystems to the SRC. The control script operates as follows:

1. It makes sure the grpsvcs subsystem is stopped.
2. It gets the port number for the grpsvcs subsystem for this cluster from the global object data manager (ODM) and makes sure the port number is set in the /etc/services file. The range of valid port numbers is 10000 to 10100, inclusive.
3. The service name that is entered in the /etc/services file is grpsvcs.*cluster_name*.
4. It removes the grpsvcs subsystem from the SRC (in case it is still there).
5. It adds the grpsvcs subsystem to the SRC. The cluster name is configured as a daemon parameter on the mkssys command.

Starting the subsystem: When the -s flag is specified, the control script uses the startsrc command to start the group services subsystem, grpsvcs.

Stopping the subsystem: When the -k flag is specified, the control script uses the stopsrc command to stop the group services subsystem, grpsvcs.

Deleting the subsystem: When the -d flag is specified, the control script uses the rmssys command to remove the group services subsystem from the SRC. The control script operates as follows:

1. It makes sure the grpsvcs subsystem is stopped.
2. It removes the grpsvcs subsystem from the SRC using the rmssys command.
3. It removes the port number from the /etc/services file.

Cleaning up the subsystems: When the -c flag is specified, the control script stops and removes the group services subsystems for all system partitions from the SRC. The control script operates as follows:

1. It stops all instances of subsystems in the subsystem group in all partitions, using the stopsrc -g grpsvcs command.
2. It removes all instances of subsystems in the subsystem group in all partitions from the SRC using the rmssys command.

Turning tracing on: When the -t flag is specified, the control script turns tracing on for the hagsd daemon, using the traceson command.

Turning tracing off: When the -o flag is specified, the control script turns tracing off (returns it to its default level) for the hagsd daemon, using the tracesoff command.

Logging: While they are running, the group services daemons provide information about their operation and errors by writing entries in a log file in the /var/ha/log directory.

Each daemon limits the log size to a pre-established number of lines. The default is 5000 lines. When the limit is reached, the daemon appends the string .bak to the name of the current log file and begins a new log. If a .bak version already exists, it is removed before the current log is renamed.

Flags

-a
Adds the subsystem.

-s
Starts the subsystems.

-k
Stops the subsystems.

-d
Deletes the subsystems.

-c
Cleans the subsystems (that is, deletes them from all system partitions).

-u
Removes the group services subsystem from all partitions.

-t
Turns tracing on for the subsystems.

-o
Turns tracing off for the subsystems.

-h
Writes the script's usage statement to standard output.

Security

You must be running with an effective user ID of `root`.

Exit Status

0
Indicates the successful completion of the command.

1
Indicates that an error occurred.

Restrictions

This script is valid in an HACMP environment only.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output.

Standard Error

This command writes error messages (as necessary) to standard error.

Examples

1. To add the group services subsystems to the SRC, enter:

   ```
   grpsvcsctrl -a
   ```

2. To start the group services subsystems, enter:

   ```
   grpsvcsctrl -s
   ```

3. To stop the group services subsystems, enter:

   ```
   grpsvcsctrl -k
   ```

4. To delete the group services subsystems from the SRC, enter:

   ```
   grpsvcsctrl -d
   ```

5. To clean up the group services subsystems, enter:

   ```
   grpsvcsctrl -c
   ```

6. To turn tracing on for the group services daemon hagsd, enter:

```
grpsvcsctrl -t
```

7. To turn tracing off for the group services daemon hagsd, enter:

```
grpsvcsctrl -o
```

Location

/opt/rsct/bin/grpsvcsctrl
Contains the grpsvcsctrl script

Files

/var/ha/log/grpsvcs_*nodenum*_*instnum*.*cluster_name*
Contains the log of the hagsd daemons on the nodes

The file name includes these variables:

nodenum
is the node number on which the daemon is running

instnum
is the instance number of the daemon

cluster_name
is the name of the cluster in which the daemon is running

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

gssd Daemon

Purpose

Services kernel requests for GSS operations.

Syntax

/usr/sbin/gssd

Description

Some NFS security methods, such as Kerberos 5, are provided under a more general mechanism called General Security Services, or GSS. In AIX, GSS services are provided by a library in the IBM Network Authentication Service (NAS) fileset. NAS is shipped on the expansion pack. The gssd daemon makes these GSS services available to the NFS server kernel code. If the gssd daemon is not running, then efforts to access files via NFS using GSS security methods such as Kerberos 5 will fail. The gssd daemon registers using RPC program number 400234.

The gssd daemon is started and stopped with the following System Resource Controller (SRC) commands:

```
startsrc -s gssd
stopsrc -s gssd
```

Files

Item	Description
/etc/nfs/hostkey	Specifies keytab file location and host principal in the following format: `path to keytab file` `host principal`
/etc/nfs/princmap	Specifies mappings to host principals in the following format: `principal1 alias1 alias2 alias3` `principal2 alias1` The aliases can be IP addresses or hostnames; the principal must match the host key maintained by kerberos.

h

The following AIX commands begin with the letter h.

ha.vsd Command

Purpose

Queries and controls the activity of the **rvsd** daemon of the recoverable virtual shared disk subsystem.

Syntax

ha.vsd
{**adapter_recovery [on** | **off]** | **debug [off]** | **mksrc** | **query** | **quorum** *n* | **qsrc** | **refresh [noquorum]** | **reset** | **reset_quorum** | **rmsrc** | **start** | **stop** | **trace [off]**}

Description

Use this command to display information about the recoverable virtual shared disk subsystem, to change the number of nodes needed for quorum, and to change the status of the subsystem.

Flags

-a
Specifies all virtual shared disks.

-v *vsd_name_list*
Specifies one or more virtual shared disk names, separated by commas.

-n *node_list*
Specifies one or more node numbers, separated by commas.

Parameters

adapter_recovery [on | off]
Enables or disables communication adapter recovery. The default is on.

The recoverable virtual shared disk subsystem must be restarted for this operand to take effect.

debug [off]
Specify debug to redirect the recoverable virtual shared disk subsystem's standard output and standard error to the console and cause the recoverable virtual shared disk subsystem to not respawn if it exits with an error. (You can use the `lscons` command to determine the current console.)

The recoverable virtual shared disk subsystem must be restarted for this operand to take effect.

Once debugging is turned on and the recoverable virtual shared disk subsystem has been restarted, `ha.vsd trace` should be issued to turn on tracing.

Use this operand under the direction of your IBM service representative.

Note: The default when the node is booted is to have standard output and standard error routed to the console. If debugging is turned off standard output and standard error will be routed to **/dev/null** and all further trace messages will be lost. You can determine if debug has been turned on by issuing `ha.vsd qsrc`. If debug has been turned on the return value will be:

```
action = "2"
```

mksrc
Uses `mkssys` to create the recoverable virtual shared disk subsystem.

query
Displays the current status of the recoverable virtual shared disk subsystem in detail.

quorum *n*
Sets the value of the quorum, which is the total number of nodes that must join the group before the virtual shared disks will be activated. Usually, quorum is defined as a majority of the nodes that are defined as virtual shared disk nodes in an RSCT peer domain, but this command allows you to override that definition.

The Recoverable virtual shared disk subsystem must be in the active state when you issue this command. This is not a persistent change.

qsrc
Displays the System Resource Controller (SRC) configuration of the Recoverable virtual shared disk daemon.

refresh [noquorum]
Uses the `refresh` command to asynchronously start a refresh protocol to all running recoverable virtual shared disk subsystems. The quorum will be reset before the refresh occurs, unless `noquorum` is specified. Use `ha.vsd query` to check for completion. The following items are refreshed in the device driver:

1. Nodes that have been added or deleted
2. Virtual shared disks that have been added or deleted
3. Changed attribute size_in_MB for virtual shared disks

reset
Stops and restarts the recoverable virtual shared disk subsystem.

reset_quorum
Resets the default quorum.

rmsrc
Uses `rmssys` to remove the recoverable virtual shared disk subsystem.

start
Starts the recoverable virtual shared disk subsystem.

stop
Stops the recoverable virtual shared disk subsystem.

trace [off]
Requests or stops tracing of the recoverable virtual shared disk subsystem. The recoverable virtual shared disk subsystem must be in the active state when this command is issued.

This operand is only meaningful after the `debug` operand has been used to send standard output and standard error to the console and the recoverable virtual shared disk subsystem has been restarted.

Security

You must have `root` authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to *RSCT Administration Guide* .

Examples

1. To stop the recoverable virtual shared disk subsystem and restart it, enter:

   ```
   ha.vsd reset
   ```

   The system returns the messages:

   ```
   Waiting for the rvsd subsystem to exit.
   rvsd subsystem exited successfully.
   Starting rvsd subsystem.
   rvsd subsystem started PID=xxx.
   ```

2. To change the quorum to five nodes of the RSCT peer domain, enter:

   ```
   ha.vsd quorum 5
   ```

   The system returns the message:

   ```
   Quorum has been changed from 8 to 5.
   ```

3. To query the rvsd subsystem, enter:

   ```
   ha.vsd query
   ```

   The system displays a message similar to the following:

   ```
   Subsystem         Group            PID     Status
         rvsd              rvsd             18320   active
         rvsd(vsd): quorum= 9/4, active=1, state=idle, isolation=member,
                    NoNodes=10, lastProtocol=nodes_failing,
                    adapter_recovery=on, adapter_status=up,
                    RefreshProtocol has never been issued from this node,
                    Running function level 4.1.0.0.
   ```

   where:

   quorum
   Is the number of total nodes or server nodes that must join the group before virtual shared disks will be activated. In the system output above, `quorum 9/4` indicates the total number of nodes (9) and the number of server nodes (4).

   active
   Indicates the activation status of the group that is being joined:

   0:
   the group is not active (quorum has not been met).

   1:
   the group is active and the shared disks have been activated.

   state
   Indicates the current protocol that is running.

   isolation
   Indicates the group membership status

   isolated:
   a group "join" has not been proposed.

   proposed:
   a group "join" has been proposed.

member:
we are a member (provider) of the group.

NoNodes
Indicates the number of nodes that have joined the group

lastProtocol
Indicates the last protocol that was run across the group.

adapter_recovery
Indicates communication adapter recovery support:

on:
adapter recovery is enabled.

off:
adapter recovery is disabled.

adapter_status
Indicates communication adapter status:

up:
the adapter is up.

down:
the adapter is down.

unknown:
the adapter status is unknown.

RefreshProtocol ...
Indicates whether a refresh protocol has been issued from this node. If so, the date and time of success or error will be displayed.

Running function level
Indicates the function level that the subsystem is running, in version, release, modification, fix level format (vrmf). (Coexistence with lower levels of the subsystem, may restrict us to running at a reduced function level.)

Location

/opt/rsct/vsd/bin/ha.vsd

ha_star Command

Purpose

Processes high availability event.

Syntax

ha_star [**-C**]

Description

The **ha_star** command is the generic high availability handling command. It is automatically invoked by the operating system through **/etc/rc.ha_star** when a CPU predictive failure is reported by the firmware.

If **ha_star** is invoked without flags, only new events are handled. If **ha_star** does not find any new event, it exits.

When running, **ha_star** handles all new events, even those which arrive while **ha_star** is handling already existing events. Only one instance of **ha_star** can be running at any given time. Should a second instance of **ha_star** be launched, it exits.

The operating system invokes **ha_star** when a high availability event is reported. The event handling may fail or it may be cancelled (for example, by signals). Aborted or cancelled events are held in memory within the kernel. When the cause of the abort has been corrected, then the event handling can be retried. This is when **ha_star** is invoked manually by the system administrator.

The **ha_star** command generates error or failure error log entries.

Description by Event Type

The **ha_star** command is invoked by the operating system to deallocate a CPU when a predictive processor failure event is detected. This deallocation may fail because some threads remain bound to the CPU being deallocated. In some cases, system administrators can fix the condition which led to the failure of the deallocation. For example, they may be able to identify and stop applications with threads bound to the last logical CPU.

The **-C** flag indicates that the high availability event to be resumed is a CPU deallocation event.

Flags

Item	Description
-C	Specifies that the event to be restarted is a CPU deallocation.

Files

Item	Description
/**usr**/**sbin**/**ha_star**	Contains the **ha_star** command.

ha_vsd Command

Purpose

Starts and restarts the Recoverable virtual shared disk subsystem. This includes configuring virtual shared disks and activating the recoverability subsystem.

Syntax

ha_vsd [**reset**]

Description

Use this command to start the recoverable virtual shared disk software after you install it, or, with the `reset` option, to stop and restart the program.

Flags

-a
Specifies all virtual shared disks.

-v ***vsd_name_list***
Specifies one or more virtual shared disk names, separated by commas.

-n ***node_list***
Specifies one or more node numbers, separated by commas.

Parameters

reset
Stops and restarts the recoverable virtual shared disk subsystem.

Security

You must have `root` authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

1
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to *RSCT Administration Guide* .

Examples

To stop the recoverable virtual shared disk subsystem and restart it, enter:

```
ha_vsd reset
```

Location

/opt/rsct/vsd/bin/ha_vsd

haemd Daemon

Purpose

Observes resource variable instances that are updated by resource monitors and generates and reports events to client programs.

Syntax

haemd

Description

The haemd (event manager) daemon observes resource variable instances that are updated by resource monitors and generates and reports events to client programs.

One instance of the haemd daemon executes on every node of a cluster. The haemd daemon is under system resource controller (SRC) control.

Because the daemon is under SRC control, it cannot be started directly from the command line. It is normally started by the `emsvcsctrl` command. If you must start or stop the daemon directly, use the `emsvcsctrl` command.

When SRC creates the haemd daemon, the actual program started is haemd_HACMP. The haemd_HACMP program, after collecting information needed by the daemon, then runs the haemd program. In other words, the haemd_HACMP program is replaced by the haemd program in the process created by SRC.

For more information about the event manager daemon, see the `emsvcsctrl` command.

Implementation Specifics

This daemon is part of Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/opt/rsct/bin/haemd
Location of the haemd daemon

haemd_HACMP Command

Purpose

Startup program for the event manager daemon.

Syntax

haemd_HACMP [-d *trace_arg*]

Description

The haemd_HACMP command is the startup program for the haemd daemon. When the event management subsystem is configured in the system resource controller (SRC) by the emsvcsctrl command, haemd_HACMP is specified as the program to be started.

This program can only be invoked by the SRC. To start the event management subsystem, use the emsvcsctrl command.

Flags

-d *trace_arg*
Should only be used under the direction of the IBM Support Center. The possible trace arguments are the same as for the haemtrcon command, except for reg and dinsts. To use this flag, the emsvcs subsystem definition in the SRC must be changed using the chssys command with the -a flag. The daemon must then be stopped and restarted.

Restrictions

This command is valid in an HACMP environment only.

Implementation Specifics

This script is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

Location

/opt/rsct/bin/haemd_HACMP
Location of the haemd_HACMP program

haemqvar Command

Purpose

Queries resource variables.

Syntax

haemqvar [-H *domain* | -S *domain*] [-c | -d | -i] [-f *file*] [-h] [*class var rsrcID* ["]]

Description

The haemqvar command queries the Event Management subsystem for information about resource variables. By default, the command writes to standard output the definitions for all resource variables in the current SP domain, that is, the current SP system partition as defined by the SP_NAME environment variable. If SP_NAME is not set the default system partition is used. The -S flag can be used to specify another SP domain (system partition). To query variables in an HACMP domain, use the -H flag. For an SP domain, the domain flag argument is a system partition name. For an HACMP domain, the domain flag argument is an HACMP cluster name. When the -H flag is specified, the command must be executed on one of the nodes in the HACMP/ES cluster.

The following information is reported for each resource variable definition:

- Variable Name
- Value Type
- Data Type
- SBS Format (if data type is Structured Byte String)
- Initial Value
- Class
- Locator
- Variable Description
- Resource ID and its description
- Default Expression (if defined) and its description

Because the default behavior of this command can produce a large amount of output, standard output should be redirected to a file.

If the -d flag is specified only the resource variable name and a short description are written to standard output, one name and description per line.

If the -c flag is specified the current values of all resource variables instances are written to standard output, one per line. The line of output contains the location of the resource variable instance (node number), the resource variable name, the resource ID of the instance and the resource variable instance value. If the resource variable is a Structured Byte String (SBS) data type, then the value of each SBS field is reported.

The -i flag reports the same information as the -c flag except that the value of the variable instance is the last known value rather than the current value. The -i flag is useful for determining what resource variable instances exist.

For both the -c and the -i flags, if an error is encountered in obtaining information about a resource variable instance, the output line contains an error message, symbolic error codes, the location of where the error originated (if it can be determined), the resource variable name and the resource ID.

To return information about specific resource variables, specify the class, var and rsrcID operands. These operands can be repeated to specify additional resource variables. In addition, the var and rsrcID operands can be wildcarded to match a number of resource variables. Note that null string operands or an asterisk must be quoted in the shells.

If class is not a null string, then all variables in the specified class, as further limited by the var and rsrcID arguments, are targets of the query. If class is a null string, then variables of all classes, as further limited by the var and rsrcID arguments, are targets of the query. The var argument can be wildcarded in one of two ways:

1. Specify the variable name as a null string
2. Truncate the name after any component

When the resource variable name is wildcarded in the first manner, then all resource variables, as further limited by the class and rsrcID arguments, are targets of the query. When the resource variable name is

wildcarded in the second manner, all resource variables whose high-order (leftmost) components match the var argument, as further limited by the class and rsrcID arguments, are targets of the query.

All resource variable instances, or definitions if neither the -c nor the -i flags are specified, of the variables specified by the class and var arguments that match the rsrcID argument are the targets of the query.

If neither the -c nor the -i flags are specified, the rsrcID argument is a semicolon-separated list of resource ID element names. If either the -c or the -i flags is specified, the rsrcID argument is a semicolon-separated list of name/value pairs. A name/value pair consists of a resource ID element name followed by an equal sign followed by a value of the resource ID element. An element value may consist of a single value, a range of values, a comma-separated list of single values or a comma-separated list of ranges. A range takes the form a-b and is valid only for resource ID elements of type integer (the type information can be obtained from the variable definition). There can be no blanks in the resource ID.

A resource ID element is wildcarded by specifying its value as the asterisk character. Only variables that are defined to contain the elements, and only the elements, specified in the rsrcID argument are targets of the query. If any element of the resource ID consists of the asterisk character, rather than a name/value pair (or just a name if querying for definitions), all variables that are defined to contain at least the remaining specified elements are targets of the query. The entire resource ID is wildcarded if it consists of only the asterisk character; all instances of all resource variables, as further limited by the class and var arguments, are targets of the query.

Note that the rsrcID argument must be quoted in the shells if it contains semicolons or asterisks.

The class, var and rsrcID operands can be placed in a file, one set of operands per line, instead of being specified as command arguments. Use the -f flag to specify the name of the file to the command. If the -f flag is used, any operands to the command are ignored. Within the file, null strings are specified as two adjacent double quotation marks. A completely wildcarded resource ID can either be a single asterisk (*) or an asterisk in double quotation marks ("*"). The arguments must be separated by blank spaces or tabs on each line.

Some examples of using wildcards in the rsrcID argument follow. For these examples, assume the class and var arguments are null strings. If either the class or var arguments or both are not null strings, targets for the query are restricted accordingly. In the first three examples, all variables whose resource IDs are defined to contain the elements NodeNum, VG and LV, and only those elements, are matched.

1. In this example, only one instance is matched:

   ```
   NodeNum=5;VG=rootvg;LV=hd4
   ```

2. In this example, one instance from each node is matched:

   ```
   NodeNum=*;VG=rootvg;LV=hd4
   ```

3. In this example, all instances of the matching resource variables are matched:

   ```
   NodeNum=*;VG=*;LV=*
   ```

4. In this example, all variables whose resource IDs are defined to contain only the element NodeNum are matched. The instances matched are associated with node 9:

   ```
   NodeNum=9
   ```

5. In this example, the same set of variables are matched, but all instances of each variable are matched:

   ```
   NodeNum=*
   ```

6. In this example, all variables whose resource IDs are defined to contain elements NodeNum and VG, as well as zero or more additional elements, are matched. The instances matched are associated with node 9:

   ```
   NodeNum=9;VG=*;*
   ```

7. In this example, all variables whose resource IDs are defined to contain the element NodeNum, as well as zero or more additional elements, are matched. All instances of the variables are matched:

```
NodeNum=*;*
```

Given the flexibility in specifying resource variables for query, it is possible that no resource variable instance or resource variable definition will match. If there is no match appropriate error information is reported, either in the form described above or as follows.

If the specification of the class, var or rsrcID arguments are in error, the output line contains an error message, symbolic error codes and the specified class name, resource variable name, and resource ID.

Flags

-H *domain*
Queries resource variables in the HACMP domain specified by *domain*.

-S *domain*
Queries resource variables in the SP domain specified by *domain*.

-c
Queries current resource variable values.

-d
Queries resource variable definitions but produces short form output.

-i
Queries instances of resource variables.

-f *file*
Queries resource variables specified in *file*.

-h
Displays a usage statement.

Parameters

class
Specifies the name of the resource variable class or a null string.

var
Specifies the name of the resource variable or a null string.

rsrcID
Specifies a resource ID or an asterisk.

Security

You must have root privilege and write access to the SDR to run this command.

You should be running on the control workstation. Before running this command, you must set the SP_NAME environment variable to the appropriate system partition name.

Exit Status

0
Indicates the successful completion of the command.

1
Indicates that an error occurred. It is accompanied by one or more error messages that indicate the cause of the error.

Restrictions

This command is valid in a PSSP environment only.

Standard Output

When the command executes successfully, it writes the following informational messages:

```
Reading Event Management data for partition syspar_name
CDB=new_EMCDB_file_name Version=EMCDB_version_string
```

Standard Error

This command writes error messages (as necessary) to standard error.

Examples

1. To obtain the definitions of all resource variables in the current cluster and place the output in a file, enter:

   ```
   haemqvar -H HAcluster > vardefs.out
   ```

2. To obtain a short form list of all resource variables whose resource IDs contain the element VG, in the HACMP cluster named HAcluster, enter:

   ```
   haemqvar -H HAcluster -d "" "" "VG;*"
   ```

3. To obtain resource variables whose resource IDs contain only the elements VG and NodeNum, enter:

   ```
   haemqvar -H HAcluster -d "" "" "VG;NodeNum"
   ```

Location

/opt/rsct/bin/haemqvar
Location of the `haemqvar` command

Files

/opt/rsct/install/config/haemloadlist
Contains the default configuration data for the Event Management subsystem

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

haemtrcoff Command

Purpose

Turns tracing off for the Event Manager daemon.

Syntax

`haemtrcoff -s` *subsys_name* `-a` *trace_list*

Description

The haemtrcoff command is used to turn tracing off for specified activities of the Event Manager daemon. Trace output is placed in an Event Management trace log for the system partition.

Flags

-s *subsys_name*
Specifies the name of the Event Management subsystem. On a node this is emsvcs. This argument must be specified.

-a *trace_list*
Specifies a list of trace arguments. Each argument specifies the type of activity for which tracing is to be turned off. At least one argument must be specified. If more than one argument is specified, the arguments must be separated by commas. The list may not include blanks.

Parameters

The following trace arguments can be specified:

init
Stops tracing the initialization of the Event Manager daemon.

config
Stops dumping information from the configuration file.

insts
Stops tracing resource variable instances that are handled by the daemon.

rmctrl
Stops tracing Resource Monitor control.

cci
Stops tracing the client communication (internal) interface.

emp
Stops tracing the event manager protocol.

obsv
Stops tracing resource variable observations.

evgn
Stops tracing event generation and notification.

reg
Stops tracing event registration and unregistration.

pci
Stops tracing the peer communication (internal) interface.

msgs
Stops tracing all messages that come to and are issued from the daemon.

query
Stops tracing queries that are handled by the daemon.

gsi
Stops tracing the Group Services (internal) interface.

eval
Stops tracing expression evaluation.

rdi
Stops tracing the reliable daemon (internal) interface.

sched
Stops tracing the internal scheduler.

shm
Stops tracing shared memory management activity.

all
Stops tracing all activities.

all_but_msgs
Stops tracing all activities except for messages. Message activity is defined by the msgs argument.

Security

You must have root privilege and write access to the SDR to run this command.

You should be running on the control workstation. Before running this command, you must set the SP_NAME environment variable to the appropriate system partition name.

Exit Status

0
Indicates the successful completion of the command.

1
Indicates that an error occurred. It is accompanied by one or more error messages that indicate the cause of the error.

Restrictions

Do not use this command during normal operation. Use this command only under the direction of the IBM Support Center. It provides information for debugging purposes and may degrade the performance of the event management subsystem or anything else that is running in the system partition.

Standard Output

When the command executes successfully, it writes the following informational messages:

```
Reading Event Management data for partition syspar_name
CDB=new_EMCDB_file_name Version=EMCDB_version_string
```

Standard Error

This command writes error messages (as necessary) to standard error.

Examples

1. To turn off all tracing for the Event Management subsystem on one of the cluster nodes, log in to the node and enter:

   ```
   haemtrcoff -s emsvcs -a all
   ```

2. To turn off all tracing of initialization and configuration for the Event Management subsystem on a cluster node, log in to the node and enter:

   ```
   haemtrcoff -s emsvcs -a init,config
   ```

Location

/opt/rsct/bin/haemtrcoff
Location of the `haemtrcoff` command

Files

/var/ha/log/em.trace.cluster_name
Contains the trace log of the `haemd` daemon on the cluster named *cluster_name*

/var/ha/log/em.msgtrace.cluster_name
Contains message trace output from the Event Manager daemon on the cluster named *cluster_name*

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

haemtrcon Command

Purpose

Turns tracing on for the event manager daemon.

Syntax

`haemtrcon -s` *subsys_name* `-a` *trace_list*

Description

The `haemtrcon` command is used to turn tracing on for specified activities of the event manager daemon. Trace output is placed in an event management trace log for the system partition. When used, the `regs`, `dinsts`, `iolists`, and `olists` parameters perform a one-time trace. The specified information is placed in the trace log, but no further tracing is done.

Flags

-s *cluster_name*
Specifies the name of the event management subsystem. On a node, *cluster_name* is `emsvcs`. This flag and parameter must be specified.

-a *trace_list*
Specifies a list of trace parameters. Each parameter specifies the type of activity for which tracing is to be turned on. At least one parameter must be specified. If more than one parameter is specified, the parameters must be separated by commas. The list may not include blanks.

Parameters

The following trace parameters can be specified:

init
Traces the initialization of the event manager daemon.

config
Dumps information from the configuration file.

insts
Traces resource variable instances that are handled by the daemon.

rmctrl
Traces resource monitor control.

cci
Traces the client communication (internal) interface.

emp
Traces the event manager protocol.

obsv
Traces resource variable observations.

evgn
Traces event generation and notification.

reg
Traces event registration and unregistration.

pci
Traces the peer communication (internal) interface.

msgs
Traces all messages that come to and are issued from the daemon.

query
Traces queries that are handled by the daemon.

gsi
Traces the group services (internal) interface.

eval
Traces expression evaluation.

rdi
Traces the reliable daemon (internal) interface.

sched
Traces the internal scheduler.

shm
Traces shared memory management activity.

all
Traces all activities.

all_but_msgs
Stops tracing all activities except for messages. Message activity is defined by the msgs argument.

regs
Traces currently registered events.

dinsts
Traces all resource variable instances known to the daemon.

iolists
Traces immediate observation lists

olists
Traces observation lists

Restrictions

Do not use this command during normal operation. Use this command only under the direction of the IBM Support Center. It provides information for debugging purposes and may degrade the performance of the event management subsystem or anything else that is running in the system partition.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

Examples

1. To turn on all tracing for the event management subsystem on one of the cluster nodes, log in to the node and enter:

```
haemtrcon -s emsvcs -a all
```

2. To turn on all tracing of initialization and configuration for the event management subsystem on a cluster node, log in to the node and enter:

```
haemtrcon -s emsvcs -a init,config
```

Location

/opt/rsct/bin/haemtrcon
Location of the `haemtrcon` command

haemunlkrm Command

Purpose

Unlocks and starts a resource monitor.

Syntax

`haemunlkrm -s` *subsys_name* `-a` *resmon_name*

Description

If the event management daemon cannot successfully start a resource monitor after three attempts within a two-hour interval, or if the daemon has successfully connected to the instances of a resource monitor *n* times within a two-hour interval, the resource monitor is "locked" and no further attempts are made to start it or to connect to any of its instances. *n* is 3 in an HACMP/ES cluster. Once the cause of the failure is determined and the problem corrected, the `haemunlkrm` command can be used to unlock the resource monitor and attempt to start it or connect to the resource monitor instances.

The status of the event manager daemon, as displayed by the `lssrc` command, indicates whether a resource monitor is locked.

Flags

-s *subsys_name*
Specifies the name of the event management subsystem. On a node, *subsys_name* is `emsvcs`. This flag and parameter must be specified.

-a *resmon_name*
Specifies the name of the resource monitor to unlock and start.

Parameters

The following trace parameters can be specified:

init
Traces the initialization of the event manager daemon.

config
Dumps information from the configuration file.

insts
Traces resource variable instances that are handled by the daemon.

rmctrl
Traces resource monitor control.

cci
Traces the client communication (internal) interface.

emp
Traces the event manager protocol.

obsv
Traces resource variable observations.

evgn
Traces event generation and notification.

reg
Traces event registration and unregistration.

pci
Traces the peer communication (internal) interface.

msgs
Traces all messages that come to and are issued from the daemon.

query
Traces queries that are handled by the daemon.

gsi
Traces the group services (internal) interface.

eval
Traces expression evaluation.

rdi
Traces the reliable daemon (internal) interface.

sched
Traces the internal scheduler.

shm
Traces shared memory management activity.

all
Traces all activities.

all_but_msgs
Stops tracing all activities except for messages. Message activity is defined by the msgs argument.

regs
Traces currently registered events.

dinsts
Traces all resource variable instances known to the daemon.

iolists
Traces immediate observation lists

olists
Traces observation lists

Security

You must have root privilege and write access to the SDR to run this command.

You should be running on the control workstation. Before running this command, you must set the SP_NAME environment variable to the appropriate system partition name.

Exit Status

0
Indicates the successful completion of the command.

1
Indicates that an error occurred. It is accompanied by one or more error messages that indicate the cause of the error.

Restrictions

Do not use this command during normal operation. Use this command only under the direction of the IBM Support Center. It provides information for debugging purposes and may degrade the performance of the event management subsystem or anything else that is running in the system partition.

Standard Output

When the command executes successfully, it writes the following informational messages:

```
Reading Event Management data for partition syspar_name
CDB=new_EMCDB_file_name Version=EMCDB_version_string
```

Standard Error

This command writes error messages (as necessary) to standard error.

Examples

1. This example applies to unlocking a resource monitor on a node.

   If the output of the `lssrc` command indicates that the program resource monitor `IBM.PSSP.harmpd` is locked, correct the condition that prevented the resource monitor from being started and enter:

   ```
   haemunlkrm -s emsvcs -a IBM.PSSP.harmpd
   ```

Location

/opt/rsct/bin/haemunlkrm
Location of the `haemunlkrm` command

Files

/var/ha/log/em.trace.*cluster_name*
Contains the trace log of the haemd daemon on the cluster named *cluster_name*.

/var/ha/log/em.msgtrace.cluster_name
Contains message trace output from the event manager daemon on the cluster named *cluster_name*.

hagsd Daemon

Purpose

Observes resource variable instances that are updated by resource monitors and generates and reports events to client programs.

Syntax

`hagsd [-a] [-s] [-k] [-d] [-c] [-u] [-t] [-o] [-r] [-h]` *daemon_name*

Description

The `hagsd` daemon is part of the group services subsystem, which provides a general-purpose facility for coordinating and monitoring changes to the state of an application that is running on the nodes of a cluster. This daemon provides most of the services of the subsystem. *daemon_name* specifies the name used by the daemon to name log files and identify its messages in the AIX error log.

One instance of the `hagsd` daemon executes on each cluster node. The `hagsd` daemon is under the control of the system resource controller (SRC).

Because the daemon is under SRC control, it is better not to start it directly from the command line. It is normally called by the `grpsvcsctrl` command, which is in turn called by the cluster startup process. If you must start or stop the daemon directly, use the `startsrc` or `stopsrc` command.

Flags

-a
Adds the subsystems.

-s
Starts the subsystems.

-k
Stops the subsystems.

-d
Deletes the subsystems.

-c
Cleans the subsystems, that is, delete them from all system partitions.

-u
Unconfigures the subsystems from all system partitions.

-t
Turns tracing on for the subsystems.

-o
Turns tracing off for the subsystems.

-r
Refreshes the subsystem.

-h
Displays usage information.

Parameters

daemon_name
Specifies the name used by the daemon to name log files and identify its messages in the AIX error log.

Security

You must have `root` privilege to run this script.

Exit Status

0
Indicates the successful completion of the command.

1
Indicates that an error occurred.

Restrictions

This command is valid in a PSSP environment only.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output.

Standard Error

This command writes error messages (as necessary) to standard error.

Examples

1. To add the group services subsystems to the SRC in the current system partition, set the SP_NAME environment variable to the appropriate system partition name and enter:

   ```
   hagsctrl -a
   ```

2. To start the group services subsystems in the current system partition, set the SP_NAME environment variable to the appropriate system partition name and enter:

   ```
   hagsctrl -s
   ```

3. To stop the group services subsystems in the current system partition, set the SP_NAME environment variable to the appropriate system partition name and enter:

   ```
   hagsctrl -k
   ```

4. To delete the group services subsystems from the SRC in the current system partition, set the SP_NAME environment variable to the appropriate system partition name and enter:

   ```
   hagsctrl -d
   ```

5. To clean up the group services subsystems on all system partitions, enter:

   ```
   hagsctrl -c
   ```

6. To unconfigure the group servicess subsystem from all system partitions, on the control workstation, enter:

   ```
   hagsctrl -u
   ```

7. To turn tracing on for the group services daemon in the current system partition, set the SP_NAME environment variable to the appropriate system partition name and enter:

   ```
   hagsctrl -t
   ```

8. To turn tracing off for the group services daemon in the current system partition, set the SP_NAME environment variable to the appropriate system partition name and enter:

   ```
   hagsctrl -o
   ```

Location

/opt/rsct/bin/hagsd
Contains the hagsd daemon

Files

/var/ha/log/hags_*nodenum_instnum*. *syspar_name*
Contains the log of the hagsd daemons on the nodes.

/var/ha/log/hags.*syspar_name_nodenum_instnum.syspar_name*
Contains the log of each hagsd daemon on the control workstation.

The file names include the following variables:

- *nodenum* is the node number on which the daemon is running
- *instnum* is the instance number of the daemon
- *syspar_name* is the name of the system partition in which the daemon is running.

hagsns Command

Purpose

Gets group services name server information.

Syntax

hagsns [-h *host*] [-c] -g *group_name*

hagsns [-h *host*] [-c] -s *subsystem_name*

hagsns [-h *host*] [-c] -p *subsystem_pid*

Description

Use the hagsns command to query the status of the group services nameserver.

Flags

-c
Forces the output as "English_only." If the -c flag is not specified, the daemon's locale will be used for the output.

-g *group_name*
Specifies a group of subsystems to get status for. The command is unsuccessful if the *group_name* variable is not contained in the subsystem object class.

-h *host*
Specifies the host to obtain name server status for.

-p *subsystem_pid*
Specifies a particular instance of the *subsystem_pid* to obtain name server status for.

-s *subsystem_name*
Specifies a subsystem to get status for. The *subsystem_name* variable can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the *subsystem_name* variable is not contained in the subsystem object class.

Parameters

daemon_name
Specifies the name used by the daemon to name log files and identify its messages in the AIX error log.

Security

You must have root authority to run this command.

Exit Status

0
Indicates that the command completed successfully.

a non-zero value
Indicates that an error occurred.

Restrictions

This command is valid in a PSSP environment only.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output.

Standard Error

This command writes error messages, as necessary, to standard error.

Examples

To get domain information from the group services subsystem, enter:

```
hagsns -c -s cthags
```

or

```
hagsns -s cthags
```

The output will look like this:

```
HA GS NameServer Status
NodeID=1.16, pid=14460, domainID=6.14, NS established,CodeLevel=GSLevel(DRL=8)
NS state=kCertain, protocolInProgress=kNoProtocol,outstandingBroadcast=KNoBcast
Process started on Jun 19 18:34:20, (10d 20:19:22) ago, HB connection took (19:14:9).
Initial NS certainty on Jun 20 13:48:45, (10d 1:4:57) ago, taking (0:0:15).
Our current epoch of Jun 23 13:05:19 started on (7d 1:48:23), ago.
Number of UP nodes: 12
List of UP nodes: 0 1 5 6 7 8 9 11 17 19 23 26
```

In this example, domainID=6.14 means that node 6 is the name server (NS) node. The domain ID consists of a node number and an incarnation number. The incarnation number is an integer, incremented whenever the group services daemon is started. NS established means that the name server was established.

Location

/opt/rsct/bin/hagsns
Contains the hagsns command

Files

/var/ha/log/hags_*nodenum_instnum*. *syspar_name*
Contains the log of the hagsd daemons on the nodes.

/var/ha/log/hags.*syspar_name_nodenum_instnum.syspar_name*
Contains the log of each hagsd daemon on the control workstation.

The file names include the following variables:

- *nodenum* is the node number on which the daemon is running.
- *instnum* is the instance number of the daemon.
- *syspar_name* is the name of the system partition in which the daemon is running.

hagsvote Command

Purpose

Gets vote information for group services groups.

Syntax

hagsvote [-h *host*] [-l] [-a *argument*] [-c] -g *group_name*

`hagsvote` [`-h` *host*] [`-l`] [`-a` *argument*] [`-c`] `-s` *subsystem_name*

`hagsvote` [`-h` *host*] [`-l`] [`-a` *argument*] [`-c`] `-p` *subsystem_pid*

Description

Use the `hagsvote` command to query the status of voting protocols for group services.

Flags

-a
Specifies a group services group name. This group name is different from that of the `-g` flag. In this case, the group was created from the client's first call to join the protocol.

-c
Requests the canonical output of the group services voting information. The output is displayed in English regardless of the installed language locale. If `-c` is not specified, the daemon's locale will be used for the output.

-g *group_name*
Specifies a group of subsystems to get status for. The command is unsuccessful if the *group_name* variable is not contained in the subsystem object class.

-h *host*
Specifies the host name which is getting status.

-l
Requests detailed output in "long" form.

-p *subsystem_pid*
Specifies a particular instance of the *subsystem_pid* variable to get the vote for.

-s *subsystem_name*
Specifies a subsystem to vote on. The *subsystem_name* variable can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the *subsystem_name* variable is not contained in the subsystem object class.

Parameters

daemon_name
Specifies the name used by the daemon to name log files and identify its messages in the AIX error log.

Security

You must have `root` privilege to run this command.

Exit Status

0
Indicates the successful completion of the command.

non-zero
Indicates that an error occurred.

Restrictions

This command is valid in a PSSP environment only.

Standard Output

This command writes error messages (as necessary) to standard error.

Standard Error

This command writes error messages, as necessary, to standard error.

Examples

1. To see information about the status of the voting protocol for the group `theSourceGroup` in long form, enter:

```
hagsvote -ls cthags -a theSourceGroup (locale-dependent)
```

   The output will look like this:

```
Number of groups: 4
Group name [theSourceGroup] GL node [26] voting data:
GL in phase [1] of n-phase protocol of type [Join].
Local voting data:
Number of providers: 1
Number of providers not yet voted: 1 (vote not submitted).
Given vote: [No vote value] Default vote: [No vote value]
ProviderID Voted? Failed? Conditional?
[101/26] No No Yes
Global voting data:
Number providers not yet voted: 1
Given vote: [No vote value] Default vote: [No vote value]
Nodes that have voted: []
Nodes that have not voted: [26]
```

   The first line of the output means that the total number of groups is 4. The second line provides the group name and the group leader node (in this case 26). The remaining lines give the voting data:

   - The group leader is in phase 1 of a n-phase protocol.
   - The protocol is the Join protocol.
   - For the local node, it has 1 provider, the number of providers which have not voted yet is 1.
   - No default vote value is given and no vote value is given.
   - Under the line "ProviderID Voted? Failed? Conditional?," "[101/16] No No Yes," means that the provider ID is 101/26, not voted yet, not failed, but wait for the vote (so it is conditional).

   The output then shows the global voting status:

   - The number of providers that have not voted yet is 1.
   - No vote value given yet, no default vote value.
   - The nodes that have voted is none.
   - The nodes that have not voted is node 26.

2. In the following example, the meaning of each line of output is the same as in the first example except that node 26 is the group leader node.

```
hagsvote -ls cthags -a theSourceGroup -c (canonical form)
```

   The output will look like this:

```
Number of groups: 4
Group Name: theSourceGroup
GL Node: 26 (I am GL)
Current phase number of an n-phase protocol: 1
Protocol name: [Join]
Local voting data:
Number of local providers: 1
Number of local providers not yet voted: 1 (vote not submitted)
Given vote: [No vote value] Default vote: [No vote value]Global voting data:
Number of nodes in group: 1
Number of global providers not yet voted: 1
Given vote: [No vote value] Default vote: [No vote value]
Nodes that have voted: []
Nodes that have not voted: [26]
```

Location

/opt/rsct/bin/hagsvote
Contains the `hagsvote` command

Files

`/var/ha/log/hags_`*nodenum_instnum. syspar_name*
Contains the log of the `hagsd` daemons on the nodes.

`/var/ha/log/hags.`*syspar_name_nodenum_instnum.syspar_name*
Contains the log of each `hagsd` daemon on the control workstation.

The file names include the following variables:

- *nodenum* is the node number on which the daemon is running
- *instnum* is the instance number of the daemon
- *syspar_name* is the name of the system partition in which the daemon is running.

halt or fasthalt Command

Purpose

Stops the processor.

Syntax

{**halt** | **fasthalt**} [**-l**] [**-n**] [**-p**] [**-q**] [**-y**]

Description

The **halt** command writes data to the disk and then stops the processor. The machine does not restart. Only a root user can run this command. Do not use this command if other users are logged in to the system. If no other users are logged in, the **halt** command can be used. Use the **halt** command if you are not going to restart the machine immediately. When the message `....Halt completed....` is displayed, you can turn off the power.

The **halt** command logs the shutdown by using the **syslogd** command and places a record of the shutdown in `/var/adm/wtmp`, the login accounting file. The system also writes an entry into the error log that states that the system was shut down.

The **fasthalt** command stops the system by calling the **halt** command. The **fasthalt** command provides BSD compatibility.

Flags

Item	Description
-l	Does not log the halt in the accounting file. The **-l** flag does not suppress accounting file update. The **-n** and **-q** flags imply the **-l** flag.
-n	Prevents the **sync** before it stops.
-p	Halts the system without a power down. **Note:** The **-p** flag has no effect if used in combination with flags not requiring a permanent halt. Power is still turned off if other operands request a delayed power-on and restart.

Item	Description
-q	Causes a quick halt. **Notes:** • Running the **halt** command with **-q** flag does not issue **sync**, so the system halts immediately. • If you run the **halt** command with the **-q** flag in a workload partition (WPAR), the **halt** command can stop the WPAR and bring it to the D (defined) state. The WPAR might not stop completely and bring the WPAR to the T (transitional) state because of the timeout condition or a delay caused while unmounting the file system.
-y	Halts the system from a dial-up operation.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To halt the system without logging the halt in the accounting file, enter the following command:

```
halt -l
```

2. To halt the system quickly, enter the following command:

```
halt -q
```

3. To halt the system from a dial-up operation, enter the following command:

```
halt -y
```

Files

Item	Description
/etc/rc	Specifies the system startup script.
/var/adm/wtmp	Specifies the login accounting file.

hangman Command

Purpose

Starts the hangman word-guessing game.

Syntax

hangman [*File*]

Description

The **hangman** command chooses a word of at least seven letters from a standard dictionary. The *File* parameter specifies an alternate dictionary. You guess the word by guessing letters one at a time. You are allowed seven mistakes.

When you start hangman, the game displays:

```
guesses: word: ....... errors: 0/7
guess:
```

The `guesses` displays the letters you have used as guesses. Every letter you guess is listed after `guesses`. The `word: .......` displays the number of letters in the mystery word. In this case there are seven `.` (periods) so there are seven letters in the word. As you correctly guess letters, the game replaces the appropriate `.` with the correct letter. The `errors: 0/7` displays the number of incorrect guesses. You enter your letter guess at the `guess:` prompt. For example:

```
guesses: word: .......... errors: 0/7
guess: q
guesses: q word: .......... errors: 1/7
guess: a
guesses: aq word: .a....a... errors: 1/7
guess: b
guesses: abq word: .a....a... errors 2/7
guess: j
guesses: abjq word: .a....a... errors: 3/7
guess: s
guesses: abjqs word: .a....a..s errors: 3/7
guess: z
guesses: abjqsz word: .a....a..s errors: 4/7
guess: y
guesses: abjqsyz word: .a....a..s errors: 5/7
guess: k
guesses: abjkqsyz word: .a....a..s errors: 6/7
guess: x
the answer was calculates, you blew it
```

To quit the game, press the Interrupt (Ctrl-C) or End Of File (Ctrl-D) key sequence.

Files

Item	Description
/usr/games	Location of the system's games.

hash Command

Purpose

Remembers or reports command path names.

Syntax

To Add the Path of a Command to the Path Name List:

hash [*Command* ...]

To Clear Path Name List:

hash -r

Description

The **hash** command affects the way the current shell remembers a command's path name, either by adding a path name to a list or purging the contents of the list.

When no parameter or flag is specified, the **hash** command reports to standard output the contents of the path name list. The report includes the path name of commands in the current shell environment that were found by previous **hash** command invocations. The display may also contain those commands invoked and found through the normal command search process.

Note: Shell built-in commands are not reported by the **hash** command.

You can use the **-r** flag to clear the contents of the command path name list. Path names can also be cleared from the list by resetting the value of the **PATH** environment variable. In the simplest form, this would be achieved by entering:

```
PATH="$PATH"
```

If the *Command* parameter is used, the **hash** command searches for the path name of the specified command and adds this path to the list. Do not use a / (slash) when you specify the command.

Since the **hash** command affects the current shell environment, it is provided as a Korn shell or POSIX shell regular built-in command. If the **hash** command is called in a separate command execution environment, as in the following examples, it will not affect the command search process of the caller's environment:

```
nohup hash -r
find . -type f | xargs hash
```

Using the **hash** command is equivalent to using the **alias -t** command.

Flag

Item	Description
-r	Clears the contents of the path name list.

Parameter

Item	Description
Command	Specifies the *Command* to add to the path name list.

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To find the path name of the **wc** command and add it to the path name list, enter:

   ```
   hash wc
   ```

2. To clear the contents of the path name list, enter:

   ```
   hash -r
   ```

Files

Item	Description
/usr/bin/ksh	Contains the Korn shell **hash** built-in command.
/usr/bin/hash	Contains the **hash** command.

hatsoptions Command

Purpose

Controls topology services options on a node or a control workstation.

Syntax

`hatsoptions [-s] [-d]`

Description

Before this command can be executed, environment variable HB_SERVER_SOCKET must be set to the location of the UNIX domain socket used by the topology services subsystem. The statement below can be used:

```
export HB_SERVER_SOCKET=/var/ha/soc/hats/server_socket.partition name
```

Alternatively, variable HA_SYSPAR_NAME can be set to the partition name.

The topology services daemon must be running in order for this command to be successful.

`hatsoptions` can be used to control a number of options in topology services. Option `-s` instructs the topology services daemon to reject messages that are apparently delayed. This can be used in very large system configurations, where messages are sometimes delayed in the network or in the sender and receiver nodes. Use this option only if the Time-Of-Day clocks are synchronized across all the nodes and the control workstation. Otherwise messages may be incorrectly discarded when the sender's Time-Of-Day clock is behind the receiver's.

Option `-d` instructs the topology services daemon not to reject messages that are apparently delayed. This is the default.

Flags

-s
Instructs the topology services daemon to reject messages that are apparently delayed.

-d
Instructs the topology services daemon not to reject messages that are apparently delayed (this is the default).

Security

You must have `root` privilege to run this command.

Exit Status

0
Indicates the successful completion of the command.

1
Indicates the command was unsuccessful.

Environment Variables

HB_SERVER_SOCKET
This environment variable should be set before this command can be executed. It must be set to the location of the UNIX domain socket used by topology services clients to connect to

the topology services daemon. This environment variable must be set to `/var/ha/soc/hats/server_socket.`*partition name*.

HA_SYSPAR_NAME
If HB_SERVER_SOCKET is not set, then HA_SYSPAR_NAME must be set to the partition name.

Restrictions

This command is valid in a peer domain only.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

This command writes error messages (as necessary) to standard error.

Examples

To instruct the topology services daemon on the local node to start discarding apparently delayed messages, enter:

```
export HA_SYSPAR_NAME=partition1

/opt/rsct/bin/hatsoptions -s
```

Location

/opt/rsct/bin/hatsoptions
Contains the `hatsoptions` command

Files

`/var/ha/soc/hats/server_socket.`*partition name*

head Command

Purpose

Displays the first few lines of a file.

Syntax

head [**-***Count* | **-c** *Number* | **-n** *Number*] [*File* ...]

Description

The **head** command writes to standard output a specified number of lines or bytes of each of the specified files, or of the standard input. If no flag is specified with the **head** command, the first 10 lines are displayed by default. The *File* parameter specifies the names of the input files. An input file must be a text file. When more than one file is specified, the start of each file will look like the following:

`==> `*filename*` <==`

To display a set of short files, identifying each one, enter:

`example% head -9999 `*filename1 filename2...*

Note: The standard input is used if you do not specify any file parameters or if you specify the '-' parameter.

Flags

Item	Description
-*Count*	Specifies the number of lines from the beginning of each specified file to be displayed. The *Count* variable must be a positive decimal integer. This flag is equivalent to the **-n** *Number* flag, but should not be used if portability is a consideration.
-c *Number*	Specifies the number of bytes to display. The *Number* variable must be a positive decimal integer.
-n *Number*	Specifies the number of lines from the beginning of each specified file to be displayed. The *number* variable must be a positive decimal integer. This flag is equivalent to the **-***Count* flag.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

To display the first five lines of the `Test` file, enter:

```
head -5 Test
```

OR

```
head -n 5 Test
```

help Command

Purpose

Provides information for new users.

Syntax

help

Description

The **help** command presents a one-page display of information for new users. Information is available for the following topics:

- Concatenating or displaying files.
- Editing lines interactively.
- Sending and receiving mail.
- Reading system messages.

- Changing password file information.
- Identifying current users of the system.
- Sending messages to the other users on the system.
- Displaying the contents of directories.
- Viewing information on the Source Code Control System.
- Setting terminal modes.

Examples

To obtain help, type `help` at the command line.

hfistat Command

Purpose

Displays the performance statistics of the host fabric interface.

Syntax

hfistat [-O options] [interval [count]]

hfistat [-h]

Description

The **hfistat** command displays performance statistics related to the host fabric interface.

The following information are the descriptions of the column headings in the output table.

Type	Column name with description
HFI	**HFI** Host fabric interface identifier (0, 1, ...) identifies the interface whose statistics are displayed. **packets - sent** Aggregate packet sent count (56-bit counter) is the number of packets sent to the cluster network, irrespective of the window it is sent from. **packets - imm_send** Aggregate immediate send packet sent count (56-bit counter) is the number of packets that must be sent immediately, irrespective of the window it is sent from. **packets - receive** Aggregate packet received count (56-bit counter) is the number of packets successfully received from the cluster network, irrespective of the window it is received from. The **packets - receive** counter increments whenever the packet received counter of any window increments. **sent_packets - fullRDMA** Aggregate full-RDMA packet sent count (56-bit counter) is the number of full-RDMA packets sent, irrespective of the window it is sent from. **sent_packets - halfRDMA** Aggregate half-RDMA packet sent count (56-bit counter) is the number of half-RDMA packets sent, irrespective of the window it is sent from. The **sent_packets - halfRDMA** counter does not increment for notifications.

Type	Column name with description
	sent_packets - smallRDMA Aggregate small-RDMA packet sent count (56-bit counter) is the number of small-RDMA packets sent, irrespective of the window it is sent from. The **sent_packets - smallRDMA** counter does not increment for notifications. **sent_packets - ip** Aggregate IP packet sent count (56-bit counter) is the number of IP packets sent, irrespective of the window it is sent from. **sent_packets - cau** Aggregate CAU packet sent count (56-bit counter) is the number of CAU packets sent, irrespective of the window it is sent from. **sent_packets - gups** Aggregate GUPS packet sent count (56-bit counter) specifies the number of GUPS packets sent, irrespective of the window it is sent from. **dropped_packets – sending** Aggregate packet dropped from sending count (56-bit counter) specifies the number of packets dropped, and not sent by the FIFO meant for sending packets, irrespective of the window. **dropped_packets – receiving** Aggregate packet dropped from received count (56-bit counter) specifies the number of packets from the ISR that are dropped and not received, irrespective of the window it is received from. **xlat – wait** Address xlat wait count (56-bit counter) specifies the number of missed translations and that are pending. The **xlat – wait** register resets when you write anything to this register.
ISR	**HFI** Host fabric interface identifier (0, 1, ...) identifies the interface whose statistics are displayed. **cycBlocked – sending** The cycles blocked from sending (64-bit) counter increments every 2 - 3 GHz chip cycle when the waiting flit cannot be sent over the link. **flits - sent** The flits sent (64-bit counter) is similar to the cycles blocked counter except that this counter increments each time a flit header passes over the corresponding ISR interface. **flits - dropped** The flits dropped (40-bit counter) is the number of flits that are dropped and counted whenever certain events happen such as the following items: 1. A port's link status bit is off. 2. The ISR ID is not valid. **link – retries** The link level retries (24-bit counter) counter increments each time a flit is removed from the link replay buffer and sent over the link again due to an error.
NMMU	**dyn_prot_cache – hits** Nest memory management unit dynamic protection cache hits. **dyn_prot_cache – misses** Nest memory management unit dynamic protection cache misses. **ATLB – hits** Nest memory management unit address translation buffer hits. **ATLB – misses** Nest memory management unit address translation buffer misses.
CAU	**cycles – waiting** Cycles waiting on credit (nonindexed counter).

Type	Column name with description
Window-based	**HFI** Host fabric interface identifier (0, 1, ...) identifies the interface whose statistics are displayed. **Win** Window number (0, 1, 2 ...) identifies the window whose statistics are displayed. **packet_indicated – send** The packet-indicated sent count (56-bit counter) increments whenever a packet is sent that has the Packet-Indicated-Count bit in the header set. **packet_indicated – receive** The packet-indicated received count (56-bit counter) increments whenever a packet is received that has the packet-indicated-count bit in the header set, but before the packet is written to memory. **packets – sent** The packet sent count (56-bit counter) is the number of packets successfully sent to the cluster network and includes packets sent to the same HFI by sending them to the cluster network where they are wrapped back. **packets – received** The packet received count (56-bit counter) is the number of packets successfully received from the cluster network and includes packets received from the same HFI, which were wrapped back by the cluster network. **packet_dropped – sending** The packet dropped from sending count (40-bit counter) is the number of packets from a send FIFO that were dropped and not sent. **packet_dropped – receiving** The packet dropped from receiving count (40-bit counter) is the number of packets from the ISR that were dropped and not received. **immediate – send_pkts** The immediate send packet count (56-bit counter).

Flags

Item	Description
Options	Specifies the content and the presentation of a report. Use the `Options` parameter with the **-O** flag. `-O option1=value1,option2=value2,option3="value3 value4 value5"` The list of options must be comma-separated and the list of values must be enclosed in quotation marks (“ ”) and separated by spaces. Following are the supported options with their values: • *type = [window nonwindow hfi isr nmmu cau all]* The **type** option specifies that the **hfistat** command displays only the specified types of register values to be displayed. `Default value: hfi` You can specify the following values to the type option: **window** Displays the window-based performance statistics. **nonwindow** Displays the nonwindow-based performance statistics. **hfi** Displays the performance counter values for HFI. **isr** Displays the integrated switch router (ISR) register values. **nmmu** Displays the Nest Memory Management Unit (NMMU) register values. **cau** Displays the Collectives Acceleration Unit (CAU) register values. **all** Displays all the register values. • *display = [raw | delta]* The display option is used to dump the register values. *Default value: none* You can specify the following values with the display option: **raw** Dump the raw register values collected. **delta** Dump the delta values of the registers. **Note:** 1. The **hfistat** tool displays formatted output when the display option is not provided. 2. The raw and delta option values are mutually exclusive. • *hfi = [0 1 ...]* The **hfi** option specifies the list of host fabric interfaces for which the register values are reported. *Default value: All available HFIs in the system* **Note:** 1. You can specify a range of host fabric interfaces in one of the following ways: `hfistat –O hfi=”0 1 2 3”` `hfistat –O hfi=0-3` 2. An empty end of the list can be used to signify the last available host fabric interface. This example of the **hfi** option indicates the range from 1 to the last available host fabric interface. `hfistat –O hfi=1-`

Item	Description
	• *window = [0 1 2...]* The **window** option specifies the list of HFI window numbers for which the register values are reported. `Default value: All available HFI windows for the specified HFIs` **Note:** 1. You can specify a range of HFI window numbers in one of the following ways: `hfistat –O window=”210 211 212 213 214 215 216”` `hfistat –O window=210-216` 2. 4. An empty end of the list can be used to signify the last available HFI window number. This example specifies the **window** option that indicates the range from 0 to the last available HFI window. `hfistat –O window=0-` • *output = <filename>* The **output** option specifies the output file to be used instead of stdout. `Default value: None` **Note:** The filename is mandatory.
interval	Specifies the interval in seconds for the **hfistat** command to collect and print statistics. If the **interval** parameter is not specified, the **hfistat** command runs with the 2-second interval.
Count	Specifies the number of repetitions for the **hfistat** command to collect and print statistics. Use the **Count** parameter along with the interval option. If both the **Count** and the **interval** parameters are not specified, the **hfistat** command runs 10 times. If the **interval** parameter is specified and not the **Count** parameter, the **hfistat** command runs indefinitely.

Examples

1. 1. To display the HFI-based performance statistics for all available HFIs by using a 2-second interval for 10 iterations, run the following command:

```
# hfistat
```

2. 2. To display all the performance statistics for available HFIs including window-based performance statistics of all windows in a formatted output, run the following command:

```
# hfistat –O type=all 2 5
```

3. 3. To display only the window-based performance statistics for fewer windows (0-15) in a formatted output, run the following command:

```
# hfistat –O type=window,window=0-15
```

4. 4. To display the CAU register values for the HFI-1 in a formatted output for 10 samples (default) and an interval of 2 seconds (default), run the following command:

```
# hfistat –O type=cau,hfi=1
```

5. 5. To display both CAU and nest memory management unit (NMMU) register values in a formatted output, run the following command:

```
# hfistat –O type=”cau nmmu”
```

6. 6. To dump the raw register values of all the HFIs including window-based performance statistics for the window 0 with an interval of 2 seconds and 5 iterations, run the following command:

```
# hfistat –O display=raw,type=all,window=0 2 5
```

7. 7. To dump delta values for only ISR performance counters with a 2-second interval and 5 iterations, run the following command:

```
# hfistat -O display=delta,type=isr 2 5
```

Files

Item	Description
`/usr/bin/hfistat`	Contains the **hfistat** command.

hdcryptmgr Command

Purpose

Provides the cryptographic management of logical volumes (LV).

Syntax

```
hdcryptmgr action [-h] [flags] devicename
```

Description

Starting from IBM AIX 7.2 with Technology Level 5, you can run the **hdcryptmgr** command by specifying the *action* parameter to perform one of the following operations:

Table 9. hdcryptmgr command operations

Operation	*action* parameter	Description
Display encryption settings	**showvg**	Displays the data encryption status of the volume group
	showlv	Displays the data encryption status of the logical volume
	showmd	Displays encryption metadata for a specific device
	showconv	Displays status of all active and stopped encryption conversions
Control authentication methods	**authinit**	Initializes a primary key for data encryption in the logical volume
	authunlock or **authunl**	Authenticates to the encrypted logical volume to unlock the primary key of the logical volume
	authadd	Adds additional authentication methods
	authcheck or **authchk**	Checks the validity of an authentication method
	authdelete or **authdel**	Removes an authentication method

Table 9. hdcryptmgr command operations (continued)		
Operation	***action* parameter**	**Description**
Manage platform keystore (PKS) keys	**pksimport**	Imports the platform keystore (PKS) keys
	pksexport	Exports the PKS keys
	pksclean	Removes a PKS key
	pksshow	Displays status of the PKS keys
Convert the encryption status of the logical volume	**plain2crypt**	Enables encryption in a logical volume and encrypts the logical volume data
	crypt2plain	Decrypts the logical volume data and disables encryption in a logical volume

Displaying encryption settings

You can run the following actions with the **hdcryptmgr** command to display encryption settings:

showvg

Syntax:

```
hdcryptmgr showvg [-h] [device]
```

Displays the data encryption status of the specified volume groups. If you do not specify a volume group, this command shows the encryption status of all the volume groups.

```
# hdcryptmgr showvg
VG NAME / ID          ENCRYPTION ENABLED
EVG1                      yes
INSTALLVG                 yes
rootvg                    no
```

showlv

Syntax:

```
hdcryptmgr showlv [-h] [-v] device
```

Displays the data encryption status of a logical volume. You must specify the device name of a volume group or a logical volume by using the **-v** flag. When you specify a volume group, this command displays the data encryption status of all the logical volumes in the volume group. When you specify a logical volume, this command displays the data encryption status of the specified logical volume. If the data encryption capability is not enabled for the volume group, a message, which indicates that encryption is not enabled on the volume group, is displayed.

```
# hdcryptmgr showlv vg00
NAME                CRYPTO_STATUS    %ENCRYPTED       NOTE
lv00                unlocked         100
lv01                unlocked         100
lv03                not_enabled      0
lv04                locked           100
lv02                uninitialized    0
lv06                uninitialized    n/a              not_accessible
lv07                locked           100
fslv00              locked           1                encrypting
```

showmd

Syntax:

```
hdcryptmgr showmd [-h] [-v] device
```

Displays encryption metadata for a specific logical volume, volume group, or physical volume. You must specify the device name of a logical volume, volume group, or a physical volume. When you

specify a volume group, only the header and trailer encryption metadata of the specified volume group are displayed. When you specify a physical volume, the metadata of encrypted logical volumes are displayed even if the corresponding volume group is not varied on. When you specify a logical volume, the entire encryption metadata of the specific logical volume is displayed.

```
# hdcryptmgr showmd ELV1
.....
.....    Wed Jun 17 13:25:46 2020
.....    Device type : LV
.....    Device name : ELV1
.....

=============== B: LV HEADER ================
Version                      : 0
MasterKey                    : Defined
MasterKey size               : 16 bytes
Encryption status            : Fully encrypted
Data crypto algorithm        : AES_XTS
=============== E: LV HEADER ================

============= B: LV AUTH METHODS ============
---- Index #0 -------------------------------
Method defined               : yes
Method name                  : initpwd
Authentication type          : Passphrase
Auto-auth method             : no
MasterKey crypto algorithm   : AES_GCM
---- Index #1 -------------------------------
Method defined               : no
---- Index #2 -------------------------------
Method defined               : no
---- Index #3 -------------------------------
Method defined               : no
---- Index #4 -------------------------------
Method defined               : no
---- Index #5 -------------------------------
Method defined               : no
============= E: LV AUTH METHODS ============
```

showconv

Syntax:

```
hdcryptmgr showconv [-h]
```

Displays the status of both active and stopped processes of logical volume that are being converted.

```
# hdcryptmgr showconv
NAME          TID/STATUS       %ENCRYPTED       DIRECTION        START_TIME
lv03          29557045         3                plain2crypt      Sun Feb 14 09:43:10 2021
fslv00        stopped/dirty    1                plain2crypt
```

Controlling authentication methods

The encryption function of the logical volume supports the following key-protection methods: passphrase, key file, key server management solution (such as IBM Security Key Lifecycle Manager), and Platform Keystore (PKS). The passphrase and key file protection methods require you to specify a password or a key file location manually. The key server management and PKS protection methods can be used to automatically unlock and activate the encrypted logical volume. For the key server authentication method to qualify as an automatic method, you must either store the client certificate password in PKS or choose no password for the client certificate. You can run the following actions with the **hdcryptmgr** command to control authentication methods:

authinit

Syntax:

```
hdcryptmgr authinit [-h] [-e algo_detail] [-n name] device
```

Initializes the primary key and encryption metadata for an encrypted logical volume. For each encrypted logical volume, the primary key and encrypted metadata must be initialized only once. A

first passphrase that is obtained from the key-protection method is added to the encryption metadata of the LV. You can specify the following flags or values for this *action* parameter:

-e
Specifies the data encryption algorithm, mode, and key length. The valid values of the **-e** flag are as follows:

prompt
Specifies that the encryption algorithm details will be prompted when the command runs.

[algorithm]:[b|B][key_len][:w]
Specifies the encryption algorithm details. The supported algorithms are Advanced Encryption Standard XTS mode (AES-XTS) 128 bits or 256 bits. The character **b** refers to bits (default) of the key, character **B** refers to bytes of the key, and the *key_len* variable refers to the length of the key. The *:w* parameter overwrites the default values of the volume group with the specified values. By default, when a volume group, in which encryption is enabled, is created, the default encryption algorithm is AES-XTS 128 bits.

-n
Specifies a name for the key-protection method.

device
Specifies the device name of the logical volume for which the key-protection method must be initialized.

authadd
Syntax:

```
hdcryptmgr authadd [-h] [-t type [-m method_detail]] [-n name] device
```

Adds an additional key-protection method to an encrypted logical volume in which a key-protection method is already initialized. To activate the authentication method that you added to an encrypted LV, the encrypted LV must be unlocked. This *action* parameter can be specified with the following flags or values:

-t
Specifies the key-protection type. The valid values are pwd, `keyfile`, `keyserv`, and `pks`.

-m
Specifies any additional information about the key-protection method that might include the following details:

- Input path to the authentication key file
- Key server ID in the `KeySvr` Object Data Manager (ODM) class

-n
Specifies a name for the key-protection method.

device
Specifies the device name of the logical volume for which the key-protection method must be added.

If you do not specify the required flags or values when you run the **hdcryptmgr authadd** command, you are prompted to specify the same. For information about registering key server information, see the **keysvrmgr** command.

authdelete or authdel
Syntax:

```
hdcryptmgr authdelete [-h] [-t type [-m method_detail]] [-i index] [-n name] [-f] device
```

Removes an initiated key-protection method. This *action* parameter can be specified with the following flags:

-t
Specifies the key-protection type. The valid values are pwd, `keyfile`, `keyserv`, and `pks`.

-m
Specifies any additional information about the key-protection method that might include the following details:

- Input path to the authentication key file
- Key server ID in the `KeySvr` ODM class

-i
Specifies the index of the key-protection method that must be deleted.

-n
Specifies the name of the key-protection method that must be deleted.

-f
Specifies the force option. This flag bypasses the authentication method checks to remove the key-protection method.

device
Specifies the device name of the logical volume for which the key-protection method must be deleted.

Only one key-protection method can be removed at a time. If you know the correct index or name of the key-protection method, you can specify the key-protection method by using the **-i** or **-n** flags. You can use the **-t** and **-m** flags to filter the list of existing key-protection methods. If multiple entries match the specified criteria, you are prompted to choose the key-protection method that must be removed.

Before the key-protection method is removed, the validity of the key-protection method is checked, unless the **-f** flag is used. You must authenticate to the LV with the selected key-protection method.

Note: Ensure that the logical volume has at least a passphrase key-protection method after performing the **authdelete** operation.

authunlock or authunl
Syntax:

```
hdcryptmgr authunlock [-h] [-t type [-m method_detail]] [-A] device
```

Authenticates to the encrypted LV and unlocks the encrypted logical volumes. This *action* parameter can be specified with the following flags or values:

-A
Authenticates to the encrypted LV by using the automatic key-protection methods that do not require any user inputs. You can use this flag at a volume group (VG) level only if the VG uses automatic key-protection methods, such as a key server management solution or PKS.

-t
Specifies the type of the key-protection method. The valid values are `pwd`, `keyfile`, `keyserv`, and `pks`.

-m
Specifies any additional information about the key-protection method that might include the following details:

- Input path to the authentication key file
- Key server ID in the `KeySvr` ODM class

device
Specifies the device name of the logical volume which must be authenticated and then the key-protection method must be unlocked. You must specify this value with the **-A** flag.

When you specify an LV device name, you can specify the key-protection method by using the **-t** and **-m** flags. If more than one key-protection methods meet the criteria, you are prompted to select a specific key-protection method.

authcheck or authchk

Syntax:

```
hdcryptmgr authcheck [-h] [-t <type> [-m <method_detail>]] [-i <index>] [-n <name>] <device>
```

Checks the validity of an authentication method. This *action* parameter can be specified with the following flags or values:

-h
Displays help information.

-t
Specifies the type of the key-protection method. The valid values are pwd, `keyfile`, `keyserv`, and pks.

-m
Specifies any additional information about the key-protection method that might include the following details:

- Input path to the authentication key file
- Key server ID in the `KeySvr` ODM class

-i
Checks the authentication of only the specified index. Authentication type is automatically forced according to the selected index.

-n
Specifies the name of the key-protection method that must be checked.

device
Specifies the device name of the logical volume that must be checked.

Managing PKS keys

The platform keystore (PKS) is a secure key-protection method that is available in IBM PowerVM® firmware of the IBM Power System E950. You can add the PKS key-protection method to an encrypted LV. You can use the following *action* parameters to manage the PKS keys for authentication.

pksshow

Syntax:

```
hdcryptmgr pksshow [-h]
```

Displays the logical volume IDs that are associated with the PKS keys and the status of the PKS keys. The LV IDs that are stored in both the PKS and in the LV metadata are displayed.

```
# hdcryptmgr pksshow

PKS uses 317 bytes on a maximum of 65536 bytes.
PKS_Label (LVid)                          Status
00fb293100004c0000000174c0a994b7.1        VALID
00fb293100004c0000000174c0a994b7.2        UNKNOWN
00fb293100004c0000000174c0a994b7.3        UNKNOWN

PKS_Label (objects)
ksvr:gpfs-pw-t2
```

pksclean

Syntax:

```
hdcryptmgr pksclean [-h] lvid
```

Removes an invalid key from the PKS. You must specify the logical volume ID that is associated with the invalid key that you want to remove. This command must be used to remove the keys that are listed in the **`hdcryptmgr pksshow`** command output with the status as O.

pksexport
Syntax:

```
hdcryptmgr pksexport [-h] -p ExportFile device
```

Exports the PKS keys into the specified file. If you specify an LV device name, the PKS key that is associated with the specified LV is exported. If you specify a VG device name, all PKS keys that are associated with the logical volumes in the volume group are exported.

pksimport
Syntax:

```
hdcryptmgr pksimport [-h] -p ExportFile [device]
```

Imports the PKS keys into the specified file. If you specify an LV device name, the PKS key that is associated with the specified LV is imported. If you specify a VG device name, all PKS keys that are associated with the logical volumes in the volume group are imported. If you do not specify a device name, all PKS keys are imported.

Converting the encryption status of the logical volume

You can convert a regular logical volume to an encrypted logical volume, and vice versa. You can perform this conversion operation only on the logical volume that is active and online.

Warning: You must back up your data before you run the following conversion commands.

You can use the following *action* parameters:

plain2crypt
Syntax:

```
hdcryptmgr plain2crypt [-h] [-e algo_detail] [-n name] [-f] device
```

Enables encryption in a logical volume, configures the encryption settings, and encrypts the LV data. This *action* parameter can be specified with the following flags and values:

-e
Specifies the data encryption algorithm, mode, and key length. The valid values of the **-e** flag are as follows:

prompt
Specifies that the encryption algorithm details will be prompted when the command runs.

[algorithm]:[b|B][key_len][:w]
Specifies the encryption algorithm details. The supported algorithms are Advanced Encryption Standard XTS mode (AES-XTS) 128 bits or 256 bits. The character **b** refers to bits (default) of the key, character **B** refers to bytes of the key, and the *key_len* variable refers to the length of the key. The *:w* parameter overwrites the default values of the volume group with the specified values. By default, when a volume group, in which encryption is enabled, is created, the default encryption algorithm is AES-XTS 128 bits.

-n
Specifies a name for the key-protection method.

-f
Specifies the force option. If you do not use this flag, the **hdcryptmgr** command prompts you to confirm that data have been backed up. The force option suppresses this prompt.

device
Specifies the device name of the logical volume for which the encryption status must be converted.

crypt2plain
Syntax:

```
hdcryptmgr crypt2plain [-h] [-f] device
```

Decrypts the encrypted data of the specified logical volume and disables the encryption status of the specified logical volume. This *action* parameter can be specified with the following flags and values:

-f
Specifies the force option. If you do not use this flag, the **hdcryptmgr** command prompts you to confirm that data have been backed up. The force option suppresses this prompt.

device
Specifies the device name of the logical volume for which the encryption status must be converted.

Commands and function restrictions for encrypted LV

For more information about the logical volume commands or functions that are not supported when the LV is encrypted, see the Limitations section in Encrypting logical volumes.

Examples

Scenario: Creating an encrypted logical volume with the passphrase key-protection method

1. Create a volume group in which encryption is enabled.

```
# mkvg -k y hdisk1 hdisk2
vg00
```

2. Create an encrypted LV with a size of 32 MB.

```
# mklv -k y vg00 32M
mklv: Please run :
# hdcryptmgr authinit lvname [..] to define LV encryption options.
lv00
```

3. Initialize the encryption configuration on the logical volume by using a primary key and the passphrase key-protection method.

```
# hdcryptmgr authinit -n default lv00
Enter Passphrase:
Confirm Passphrase:
Password authentication method added successfully
```

Scenario: Creating a file system in an encrypted LV

1. Create volume group in which encryption is enabled, and then create a logical volume with a size of 32 MB, and then initialize the encryption configuration for the logical volume.

```
# mkvg -k y hdisk1 hdisk2
vg00
# mklv -t jfs2 -k y vg00 32M
mklv: Please run :
# hdcryptmgr authinit lvname [..] to define LV encryption options.
fslv00
# hdcryptmgr authinit -n default fslv00
Enter Passphrase:
Confirm Passphrase:
Password authentication method added successfully
```

2. Create a file system in the encrypted logical volume similar to creating it in a regular logical volume.

```
# crfs -v jfs2 -d fslv00 -m /mnt/myfs -A no
File system created successfully.
32560 kilobytes total disk space.
New File System size is 65536
```

Scenario: Authenticating to a logical volume in which encryption is enabled

When the volume group is varied off or the system is restarted, the authentication to the encrypted LV expires. You must authenticate to the encrypted LV to access its data. You must use the configured key-protection method for the encrypted LV. To authenticate an encryption-enabled LV, complete the following steps:

1. Vary on the VG.

```
# varyonvg vg00
```

2. Authenticate by using the passphrase key-protection method.

```
# hdcryptmgr authunlock -t pwd fslv00
Enter Passphrase:
Password authentication succeeded
```

Files

/usr/sbin/hdcryptmgr
Contains the **hdcryptmgr** command.

hmcauth Command

Purpose

The **hmcauth** command is used to authenticate with a Hardware Management Console (HMC) and get a token to use the HMC services for a AIX Live Update operation. It can also be used to invalidate a token.

Syntax

To authenticate with an HMC and get a token, use the following syntax:

```
hmcauth [ -u user_name ] [ -p password ] [ -a hmc ] [ -P port ]
```

To invalidate and remove a previously generated token, use the following syntax:

```
hmcauth -r [ -a hmc ] [ -u user_name ]
```

To list all the known HMC authentication tokens, use the following syntax:

```
hmcauth -l
```

To display the command usage statement, use the following syntax:

```
hmcauth -h
```

Description

You can use the **hmcauth** command if you have all object access and appropriate HMC administrative authority. The **hmcauth** command generates a token that can be used by an AIX partition administrator to perform the Live Update operation. If the command succeeds, a token is stored in the kernel so that the **geninstall** interface can perform the Live Update operation.

To use this command, you must have authority to perform the following tasks:

- Power on a managed partition.
- Shut down a managed partition.
- Remove a managed partition (automatic mode only).
- Create a managed partition based on the current profile (automatic mode only).
- Set the boot device of a managed partition.

- Manage the virtual Ethernet adapters.

The `hmcclientliveupdate` HMC role has all the privileges that are required for the Live Update operation. If a user is defined on the HMC with this role, the authentication can be done with this user rather than the `hscroot` user.

The **hmcauth** command can also be used without any flags. If you do not specify any flags, the **hmcauth** command prompts for all the required information such as *user_name*, *hmc*, and *password*.

Note: If the LPAR is restarted, the HMC authentication token is not preserved. Therefore, you must authenticate with the HMC again before attempting a Live Update operation.

Parameters

Item	Description
user_name	A string of up to 64 characters that specifies the HMC user name.
password	A string of up to 64 characters that specifies a password.
hmc	A string of up to 64 characters that specifies either the host name or the IP address of the HMC to authenticate with.
port	A string of up to 16 characters that specifies a port number to contact the HMC.

Flags

Item	Description
-a *hmc*	Specifies the host name or the IP address of the HMC to authenticate with. If the *hmc* variable is not specified, the command prompts for it.
-h	Writes the command usage statement to standard output.
-r	Removes the token that is generated by the HMC.
-P *port*	Specifies a port number to be used to contact the HMC. The **-P** flag is optional. Therefore, if the port number is not specified, the port number is defaulted to the value of 12443. The HMC always uses port 12443, but if any proxy setup is used, you can use the **-P** option to allow the proxy to use a port other than 12443.
-p *password*	Specifies the password for authentication. If the password is not specified on the command line, you are prompted for the password.
-u *user_name*	Specifies the HMC user name to authenticate as. You must have all object access and appropriate task authority on the HMC.

Examples

1. To authenticate with the HMC called `apollo`, enter the following command:

   ```
   # hmcauth -a apollo -u hscroot -p T2x6z42p
   ```

2. To authenticate with an HMC at IP 5.5.55.121 with password prompt, enter the following command:

   ```
   # hmcauth -a 5.5.55.121 -u hscroot
   Enter password for hscroot:
   ```

3. To invalidate a previous authentication with an HMC at IP 5.5.55.121, enter the following command:

```
# hmcauth -r -a 5.5.55.121
```

4. To authenticate with an HMC called `apollo` that has a firewall, where the HMC port 12443 is not accessible, a rebound proxy node can be set up to use a different open port. To use the SSH client with port 14111 on a proxy node that is called `proxy1` to authenticate from a logical partition `mylpar`, enter the following commands:

```
(0) root @ proxy1: /
# ssh -R localhost:14111:apollo:12443 root@mylpar

(0) root @ mylpar: /
# hmcauth -a localhost -u hscroot -P 14111
Enter HMC password:
```

   You can specify `localhost` as the `management_console` attribute in the hmc stanza of the `lvupdate.data` file to initiate the Live Update operation.

host Command

Purpose

Resolves a host name into an Internet Protocol (IP) address or an IP address into a host name.

Syntax

`host` [**-n** [**-a**] [**-c** *Class*] [**-d**] [**-r**] [**-t** *Type*] [**-v**] [**-w**]] *Hostname* | *Address* [*Server*]

`hostnew` [**-a**] [**-c** *Class*] [**-d**] [**-r**] [**-t** *Type*] [**-v**] [**-w**] *Hostname* | *Address* [*Server*]

Description

The `/usr/bin/host` command returns the IP address of a host machine when the *HostName* parameter is specified and the name of the host when the *Address* parameter is specified. Depending on the configuration of name resolution service, the **host** command might also display any aliases that are associated with the *HostName* parameter. Examples of name resolution services include `local`, `nis`, and `bind`.

If the local host is using the Domain Name Protocol, the local or remote name server database is queried before it searches the local **/etc/hosts** file.

Flags

Item	Description
-a	Equivalent to using "`-v -t *`"
-c *Class*	Specifies the class to look in when it searches non-Internet data. Valid classes follow: **IN** Internet class **CHAOS** Chaos class **HESIOD** MIT Althena Hesiod class **ANY** Wildcard (any of the above)
-d	Turns on debugging mode.
-n	Equivalent to issuing the `/usr/bin/hostnew` command. The **hostnew** command performs `bind` resolution service.

Item	Description
-r	Disables recursive processing.
-t *Type*	Specifies the type of record to query for. Valid types follow: **A** Host's IP address **CNAME** Canonical name for an alias **HINFO** Host processor and operating system type **KEY** Security Key Record **MINFO** Mailbox or mail list information **MX** Mail exchanger **NS** Nameserver for the named zone **PTR** Host name if the query is an IP address; otherwise, the pointer to other information **SIG** Signature Record **SOA** Domain's "start-of-authority" information **TXT** Text information **UINFO** User information **WKS** Supported well-known services
-v	Verbose mode.
-w	Waits forever for a reply from the DNS server.

Parameters

Item	Description
Address	Specifies the IP address of the host machine to use in resolving the host name. The *Address* parameter must be a valid IP address in dotted decimal format.
HostName	Specifies the name of the host machine to use in resolving the IP address. The *HostName* parameter can be either a unique host name or a well-known host name (such as *nameserver*, *printserver*, or *timeserver*, if these names exist).
Server	Specifies the nameserver to query.

Examples

1. To display the address of a host machine named `mephisto`, enter the following command:

```
host mephisto
```

   The output is similar to the following information:

```
mephisto is 192.100.13.5, Aliases: engr, sarah
```

2. To display the host whose address is `192.100.13.1`, enter the following command:

```
host 192.100.13.1
```

The output is similar to the following information:

```
mercutio is 192.100.13.1
```

3. To display the MX records for the domain named `test.ibm.com`, enter:

```
host -n -t mx test.ibm.com
```

or

```
hostnew -t mx test.ibm.com
```

The output is similar to the following information:

```
test.ibm.com mail is handled (pri=10) by test1.tt.ibm.com
test.ibm.com mail is handled (pri=10) by test2.aix.ibm.com
```

Files

Item	Description
/etc/hosts	Contains the Internet Protocol (IP) name and addresses of hosts on the local network.

host9 Command

Purpose

Performs DNS lookups.

Syntax

host9 [**-aCdlrsTwv**] [**-c** *class*] [**-N** *ndots*] [**-R** *number*] [**-t** *type*] [**-W** *wait*] [**-m** *flag*] [**-4**] [**-6**] *name* [*server*]

Description

The **host9** command is a simple utility for performing DNS lookups. You can use this command to convert names to IP addresses and vice versa. When you specify no arguments or options, the **host9** command prints a short summary of its command line arguments and options.

Flags

Item	Description
-a	Equivalent to using the flags of **-v -t ***.
-c *class*	Instructs to make a DNS query of the specified class. You can use this flag to look up Hesiod or Chaosnet class resource records. The default class is IN (Internet).
-C	Attempts to display the SOA records for the zone name from all of the listed authoritative name servers for that zone. The NS records found for the zone defines the list of name servers.
-d	Generates the verbose output. This flag is equivalent to the **-v** flag.

Item	Description
-l	Specifies the list mode. This makes the **host9** command perform a zone transfer for the zone name. Transfers the zone printing out the NS, PTR and address records (A/AAAA). If you use the **-l** flag with the **-a** flag, the **host9** command prints all records.
-m *flag*	Sets the memory usage debugging flags record, usage, and trace.
-N *ndots*	Sets the number of dots that have to be in the name for it to be considered absolute. The default value is that defined using the ndots statement in the **/etc/resolv.conf** file, or 1 if no ndots statement is present. Names with fewer dots are interpreted as relative names and will be searched for in the domains listed in the search or domain directive in the **/etc/resolv.conf** file.
-r	Enables the **host9** command to mimic the behavior of a name server by making non-recursive queries and expecting to receive answers to those queries that are usually referrals to other name servers.
-R *number*	Changes the number of UDP retries for a lookup. The *number* value indicates how many times the **host9** command repeats a query that does not get answered. The default number of retries is 1. If the number is negative or zero, the number of retries defaults to 1.
-s	Informs the **host9** command not to send the query to the next name server if any server responds with a SERVFAIL response.
-t *type*	Selects the query type. The type can be any recognized query type: CNAME, NS, SOA, and so on. When no query type is specified, the **host9** command automatically selects an appropriate query type. By default, it looks for A records, but if you specify the **-C** flag, queries are made for SOA records, and if name is a dotted-decimal IPv4 address or colon-delimited IPv6 address, the **host9** command queries for PTR records. If a query type of IXFR is chosen, you can specify the starting serial by appending an equal sign, followed by the starting serial number (for example, **-t** IXFR=12345678).
-T	Uses a TCP connection when querying the name server. TCP is automatically selected for queries that require it, such as zone transfer (AXFR) requests.
-v	Generates the verbose output. This flag is equivalent to the **-d** flag.
-w	Waits forever for a reply. The time to wait for a response is set to the number of seconds given by the hardware's maximum value for an integer quantity.
-W *wait*	Waits for the *wait* seconds. If the *wait* value is less than one, the wait interval is set to 1 second.
-4	Forces the **host9** command to only use IPv4 query transport.
-6	Forces the **host9** command to only use IPv6 query transport.
name	Specifies the domain name that is to be looked up. It can also be a dotted-decimal IPv4 address or a colon-delimited IPv6 address, in which case the **host9** command performs a reverse lookup for that address.
server	Specifies an optional argument, which is either the name or IP address of the name server that the **host9** command queries instead of the server or servers listed in the **/etc/resolv.conf** file.

IDN SUPPORT

If the **host9** command has been built with internationalized domain name (IDN) support, it can accept and display non-ASCII domain names. The **host9** command appropriately converts character encoding of domain names before sending a request to the DNS server or displaying a reply from the server. If you'd like to turn off the IDN support for some reason, define the IDN DISABLE environment variable; the IDN support is disabled if the variable is set when the **host9** command runs.

Files

Item	Description
/etc/resolv.conf	

Examples

1. To display the address of a host machine named `mephisto`, enter the following command:

   ```
   host9 mephisto
   ```

   This command displays information similar to the following:

   ```
   mephisto is 192.100.13.5, Aliases: engr, sarah
   ```

2. To display the host machine with an address of 192.100.13.1, enter the following command:

   ```
   host9 192.100.13.1
   ```

   This command displays information similar to the following:

   ```
   mercutio is 192.100.13.1
   ```

3. To display the MX records for the domain named `test.ibm.com`, enter the following command:

   ```
   host9 -n -t mx test.ibm.com
   ```

   This command displays information similar to the following:

   ```
   test.ibm.com mail is handled (pri=10) by test1.tt.ibm.com
   test.ibm.com mail is handled (pri=10) by test2.aix.ibm.com
   ```

hostent Command

Purpose

Directly manipulates address-mapping entries in the system configuration database.

Syntax

To Add an Address-to-Host Name Mapping

hostent **-a** *IPAddress* **-h** "*HostName...*"

To Delete an Address-to-Host Name Mapping

hostent **-d** *IPAddress*

To Delete All Address-to-Host Name Mappings

hostent **-X**

To Change an Address-to-Host Name Mapping

hostent **-c** *IPAddress* **-h** "*HostName...*" [**-i** *NewIPAddress*]

To Show an Address or Host Name in Colon Format

hostent -s { *IPAddress* | **"***HostName***"** } [**-Z**]

To Show all Address-to-Host Name Mappings in Colon Format

hostent -S [**-Z**]

Description

The **hostent** low-level command adds, deletes, or changes address-mapping entries in the system configuration database. Entries in the database are used to map an Internet Protocol (IP) address (local or remote) to its equivalent host names.

The **hostent** command can show one or all address-to-host name mapping entries in the **/etc/hosts** file. An Internet Protocol (IP) address of a given local or remote host might be associated with one or more host names. Represent an IP address in dotted decimal format. Represent a host name as a string with a maximum length of 255 characters, and use no blank characters. Each entry must be contained on one line. Multiple *HostNames* (or aliases) can be specified.

Note: Valid host names or alias host names must contain at least one alphabetic character. If you choose to specify a host name or alias that begins with an `x` followed by any hexadecimal digit (`0-f`), the host name or alias must also contain at least one additional letter that cannot be expressed as a hexadecimal digit. The system interprets a leading `x` followed by a hexadecimal digit as the base 16 representation of an address unless there is at least one character in the host name or alias that is not a hexadecimal digit. Thus, `xdeer` would be a valid host name, whereas `xdee` would not.

You can use the System Management Interface Tool (SMIT) **`smit hostent`** fast path to run this command.

Flags

Note: The **-a**, **-d**, **-c**, and **-s** flags cannot be used together.

Item	Description
-a *IPAddress*	Adds an IP address-to-host name mapping entry for the Internet Protocol address in the database. Specify the host names with the **-h** flag.
-c *IPAddress*	Changes an IP address-to-host name mapping entry in the database that corresponds to the address that is specified by the *IPAddress* variable. Specify the changed host names with the **-h** flag. If you want to change the current IP address to a new address (*IPAddress*), use the **-i** flag.
-d *IPAddress*	Deletes the IP address-to-host name mapping entry in the database that corresponds to the address that is specified by the *IPAddress* variable.
-h"*HostName...***"**	Specifies a list of host names. Entries in the list are to be separated by blanks. The **-h"***HostName...***"** flag should be used with the **-a** flag. The **-c** flag might also require the **-h"***HostName...***"** flag.
-i *NewIPAddress*	Specifies a new IP address. This flag is required by the **-c** flag if an existing IP address is to be replaced by the *NewIPAddress* variable.
-S	Shows all entries in the database.
-s"*HostName***"**	Shows an IP address-to-host name mapping entry matching the host name specified by the **"***HostName***"** variable.
-s *IPAddress*	Shows an IP address-to-host name mapping entry matching the entry specified by the *IPAddress* variable.
-X	Deletes all IP address-to-host name mapping entries in the database.
-Z	Generates the report of the query in colon format. This flag is used when the **hostent** command is started from the SMIT usability interface.

Note: The **hostent** command does recognize the following addresses: .08, .008, .09, and .009. Addresses with leading zeros are interpreted as octal, and numerals in octal cannot contain 8s or 9s.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To add an entry in the database associating an address with a series of host names, enter the command in the following format:

   ```
   hostent -a 192.100.201.7 -h "alpha bravo charlie"
   ```

   In example 1, the IP address `192.100.201.7` is specified as the address of the host that has a primary host name of `alpha` with synonyms of `bravo` and `charlie`.
2. To show an entry in the database matching a host name, enter the command in the following format:

   ```
   hostent -s alpha
   ```

   In example 2, the entry to be shown matches the host name `alpha`.
3. To change the IP address of an entry to a new IP address, enter the command in the following format:

   ```
   hostent -c 192.100.201.7 -i 192.100.201.8
   ```

   In example 3, the old IP address is `192.100.201.7` and the new address is `192.100.201.8`.

Files

Item	Description
/etc/hosts	Contains host names and addresses for the network.

hostid Command

Purpose

Sets or displays the identifier of the current local host.

Syntax

/usr/sbin/hostid [*HexNumber* | *InternetAddress* | *HostName*]

Description

The **/usr/sbin/hostid** command displays the identifier (either a unique host name or a numeric argument) of the current local host as a hexadecimal number. This numeric value is expected to be unique across all hosts and is commonly set to the address of the host specified by the *InternetAddress* or *HostName* parameter. The root user can set the **hostid** command by specifying a hexadecimal number for the *HexNumber*, *InternetAddress*, or *HostName* parameter. The host identifier is set to the hostname by the **/etc/rc.net** file.

Parameters

Item	Description
HexNumber	Specifies a unique hexadecimal number representing the current local host.
InternetAddress	Specifies an Internet address representing the current local host.
HostName	Specifies a symbolic name that maps to a unique host.

Examples

1. To set the identifier of the local host to the local Internet address with the **hostid** command, enter the command in the following format:

   ```
   hostid   192.9.200.3
   0xc009c803
   ```

   The **hostid** command converts the Internet address `192.9.200.3` into the hexadecimal representation `0xc009c803`, and then sets the local host (your workstation connected to a network) to this address.
2. To display the identifier of the local host, enter:

   ```
   hostid
   0xc009c803
   ```

   The **hostid** command displays the identifier of the host as a hexadecimal number.

hostmibd Daemon

Purpose

Starts the **hostmibd** dpi2 sub-agent daemon as a background process.

Syntax

hostmibd [**-f** *File*] [**-d** [*Level*]] [**-h** *Hostname*] [**-c** *Community*]

Description

The **hostmibd** command starts the **hostmibd** dpi2 sub-agent. This command may only be issued by a user with root privileges or by a member of the system group.

The **hostmibd** daemon complies with the standard Simple Network Management Protocol Distributed Protocl Interface Version 2.0 defined by RFC 1592. It is acting as a dpi2 sub-agent to communicate with the dpi2 agent through dpiPortForTCP.0 (1.3.6.1.4.1.2.2.1.1.1.0) which is defined in RFC1592 section 3.1.

The Management Information Base (MIB) is defined by RFC 1155. The specific MIB variables `hostmibd` is managing are defined by RFC 2790. The actual MIB variables managed by **hostmibd** are the following six subtrees:

- hrSystem (1.3.6.1.2.1.25.1)
- hrStorage (1.3.6.1.2.1.25.2)
- hrDevice (1.3.6.1.2.1.25.3)
- hrSWRun (1.3.6.1.2.1.25.4)
- hrSWRunPerf (1.3.6.1.2.1.25.5)
- hrSWInstalled (1.3.6.1.2.1.25.6)

The **hostmibd** daemon is normally executed during system startup when **/etc/rc.tcpip** shell script is called.

The **hostmibd** daemon should be controlled using the System Resource Controller(SRC). Entering **hostmibd** at the command line is not recommended.

Use the following SRC commands to manipulate the **hostmibd** daemon:

startsrc
Starts a subsystem, group of subsystems, or a subserver.

stopsrc
Stops a subsystem, group of subsystems, or a subserver.

refresh
Causes a subsystem or group of subsystems to reread the appropriate configuration file.

lssrc
Gets the status of a subsystem, group of subsystems, or a subserver. If the user issuing the long status form of the **lssrc** command is not the root user, no community name information is displayed.

Flags

Item	Description
-c *Community*	Use specified community name. If **-c** flag is not specified, the default community name is 'public'.
-d *Level*	Specifies tracing/debug level. The levels are: • 0 = Least level • 8 = DPI level 1 • 16 = DPI level 2 • 32 = Internal level 1 • 64 = Internal level 2 • 128 = Internal level 3 Add the numbers for multiple trace levels. The default level is 56 if the **-d** flag is specified but *Level* is not specified. If the **-d** flag is not specified, the default level is 0.
-f *File*	Specifies a non-default configuration file. If the **-f** flag is not specified, the default configuration file is **/etc/hostmibd.conf**. See **/etc/hostmibd.conf** file for information on this file format.
-h *Host*	Send request to specified host. The *Host* value can be an IPv4 address, an IPv6 address, or a host name. If **-h** flag is not specified, the default destination host is 'loopback' (127.0.0.1).

Examples

1. To start the **hostmibd** daemon, enter a command similar to the following:

   ```
   startsrc -s hostmibd -a "-f /tmp/hostmibd.conf"
   ```

   This command starts the **hostmibd** daemon and reads the configuration file from **/tmp/hostmibd.conf**.
2. To stop the **hostmibd** daemon, normally enter:

   ```
   stopsrc -s hostmibd
   ```

   This command stops the **hostmibd** daemon. The **-s** flag specified the subsystem that follows to be stopped.
3. To get the short status from the **hostmibd**, enter:

   ```
   lssrc -s hostmibd
   ```

This command returns the name of the daemon, the process ID of the daemon, and the state of the daemon (active or inactive).

4. To get long status from the **hostmibd** daemon, enter:

```
lssrc -ls hostmibd
```

If you are the root user, this long form of the status report lists the configuration parameters in **/etc/hostmibd.conf**.

Files

Item	Description
/etc/hostmibd.conf	Defines the configuration parameters for **hostmibd** command.
/etc/mib.defs	Defines the Management Information Base (MIB) variables the SNMP agent and manager should recognize and handle.

hostname Command

Purpose

Sets or displays the name of the current host system.

Syntax

/usr/bin/hostname [*HostName*] [**-s**]

Description

The **/usr/bin/hostname** command displays the name of the current host system. Only users with root user authority can set the host name. The **mkdev** command and the **chdev** commands also set the host name permanently. Use the **mkdev** command when you are defining the TCP/IP instance for the first time.

You can use the System Management Interface Tool (SMIT) **smit mkhostname** fast path to run this command.

Flags

Item	Description
-s	Trims any domain information from the printed name.

Parameters

Item	Description
HostName	Sets the primary name of the host.

Note: You must have root user authority to use the *HostName* parameter.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

hosts2ldif Command

Purpose

Creates an LDAP Data Interchange Format (LDIF) file from a hosts file.

Syntax

hosts2ldif [**-i** *InputFile*] [**-o** *OutputFile*] [**-s** *SearchBase*]

Description

The **/usr/sbin/hosts2ldif** command creates a LDAP Data Interchange Format (LDIF) file from **/etc/hosts** or another file that looks like **/etc/hosts**. With no flags, the **/etc/hosts** file is used to create the **/tmp/hosts.ldif** LDIF file using `cn=hosts` as the baseDN.

The LDIF file created by this command is compliant with SecureWay Directory Schema and is used for setting up the **ldap** mechanism. The **ldap** mechanism is supported, but the use of the **nis_ldap** mechanism rather than the **ldap** mechanism is recommended.

Flags

Item	Description
-i *InputFile*	Specifies the hosts file used for input.
-o *OutputFile*	Specifies the LDIF file used for output.
-s *SearchBase*	Specifies the baseDN of the host table on the LDAP server.

Examples

1. To create **/home/ldifhosts** from the **/etc/hosts** file, type:

   ```
   hosts2ldif -o /home/ldifhosts
   ```

2. To create **/tmp/hosts.ldif** from the **/home/hosts.bak** file, type:

   ```
   hosts2ldif -i /home/hosts.bak
   ```

3. To create **/home/ldifhosts** from the **/etc/hosts** file using `cn=hosttab` as the baseDN, type:

   ```
   hosts2ldif -o /home/ldifhosts -s cn=hosttab
   ```

Files

Item	Description
/etc/hosts	Contains the Internet Protocol (IP) name and addresses of hosts on the local network.

hp Command

Purpose

Handles special functions for the HP2640- and HP2621-series terminals.

Syntax

hp [**-e**] [**-m** ...]

Description

The **hp** command reads standard input (usually output from the **nroff** command), and writes to standard output, which is usually Hewlett-Packard 2640- and 2621-series terminal displays.

If your terminal has the display enhancement feature, you can display subscript characters and superscript characters. With the mathematical-symbol feature, you can display Greek characters and other special characters, with two exceptions. The **hp** command approximates the logical operator NOT with a right arrow and shows only the top half of the integral sign.

Overstrike characters are characters followed by a backspace and another character. They appear underlined or in inverse video (depending on terminal enhancements) if either the overwritten character or the character typed after the backspace is an underscore character.

Note: Some sequences of control characters (reverse line-feeds and backspaces) can make text disappear from the display. Tables with vertical lines generated by the **tbl** command may be missing lines of text containing the bottom of a vertical line. You may be able to avoid these problems by first piping the input through the **col** command and then through the **hp** command.

Flags

Item	Description
-e	Shows overstruck characters underlined, superscript characters in half-bright, and subscript characters in half-bright underlined. Otherwise, all overstruck characters, subscript characters, and superscript characters appear in inverse video (dark-on-light). Use this flag only if your display has the display enhancements feature.
-m	Produces only one blank line for any number of successive blank lines in the text.

hplj Command

Purpose

Postprocesses the **troff** command output for the HP LaserJet Series printers.

Syntax

hplj [**-F** *Directory*] [**-quietly**] [**-landscape**] [*File* ...]

Description

The **hplj** command processes the output of the **troff** command for output to Hewlett-Packard LaserJet Series printers.

If given one or more files as options, the **hplj** command processes those files. If no files are specified, it acts as a filter interpreting standard input. The parameter *File* specifies files the **hplj** command processes to output on an HP Laser Jet Series printer.

Note: The **hplj** command can use the K cartridge or Text-Equations cartridge if installed in the printer. (The Text-Equations cartridge, HP part number C2053A #C07, supersedes the K cartridge.) The default font files assume one of the cartridges is installed. If you do not have a K cartridge, use the downloaded bit-mapped fonts instead. To do this, run the **no_cart** shell script in the font directory for the HP printer (**/usr/lib/font/devhplj**).

Incorrect output can occur if your font files assume either cartridge is mounted when it is not. Incorrect output can also occur if other cartridges or soft fonts are installed, in addition to the K cartridge or Text-Equations cartridge.

The **hplj** command depends on the files with names ending in **.out** in the **/usr/lib/font/devhplj** file. This command does not produce reasonable output unless these files have been properly set up. See the **troff** font file format document for more information.

Flags

Item	Description
-F*Directory*	Identifies the specified directory as the place to find the font file. By default, the **hplj** command looks for font files in the **/usr/lib/font/devhplj** directory.
-quietly	Suppresses all nonfatal error messages.

Item	Description
-landscape	Prints the specified file in landscape format. A landscape page is oriented so that for normal reading, the width of the page is greater than its length. By default, the **hplj** command prints in portrait orientation. **Note:** Landscape is only available in the Courier font on the Hewlett-Packard Jet II printer. Therefore, **troff** documents must be formatted in the Courier font. To accomplish this, insert the following lines at the beginning of the **troff** input file: `.fp 1 C` `.fp 2 C` `.fp 3 CB` The Courier font is loaded onto font positions #1 & #2 and Courier-Bold onto position #3.

Examples

1. To print a **troff** file named `foo` on the printer called hp using the **lp** command, enter:

```
troff -mm -Thplj foo | hplj | lp -dhp -o -dp
```

2. To print a **troff** file named boo on printer called hp using the **qprt** command, enter:

```
troff -mm -Thplj boo | hplj | qprt -dp -Php
```

Note: The **-dp** flag in both examples sends the printer data to the print device in pass-through (unmodified) mode.

File

Item	Description
/usr/lib/font/devhpl/*.out	Contains font files.

hpmcount Command

Purpose

Measures application performance.

Syntax

`hpmcount` [`-a`] [**-b** *time_base*] [`-d`] [**-D** *metrics*] [**-g** *event_groups*] [`-H`] [`-k`] [**-m** *metrics_groups*] [`-o` *file*] [`-s` *set*] [**-x**] *command*

`hpmcount [-h]`

Description

The `hpmcount` command provides the execution wall clock time, hardware performance counters information, derived hardware metrics, and resource utilization statistics (obtained from the `getrusage()` system call) for the application named by *command*.

Event types to be monitored and the associated hardware performance counters are specified by setting the `-s` option, by specifying an event group name, set number, or a comma-separated list of set numbers in the `HPM_EVENT_SET` environment variable, or by specifying counter/event pairs POWER3 / PowerPC 604 RISC Microprocessor) or an event group name (POWER4 and later) in the `libHPM_events` input file (takes precedence over `HPM_EVENT_SET`). Each set can be qualified by a counting mode. An event group number or name can be specified by setting the **-g** option or specifying a comma-separated list of event groups in the **HPM_EVENT_GROUP** environment variable. In the same manner, each event group can be qualified by a counting mode.

Valid event set numbers run from 1 to an upper limit dependent upon the processor type, which can be listed using the `pmlist` command. A comma-separated list of event sets can be specified instead of a set number, in which case the counter multiplexing mode is selected. To select all event sets, set the number value to `0`.

A comma-separated list of derived metrics can be specified by setting the **-D** option. Each derived metric can be qualified by a counting mode.

A list of derived metric groups can be specified by setting the **-m** option or by specifying a comma-separated list of derived metric groups in the **HPM_PMD_GROUP** environment variable. This allows the selection of all of the derived metrics pertaining to the specified groups. Each metric group can be qualified by a counting mode.

System and hypervisor (for processors supporting hypervisor mode) activity can be included in counting by specifying the `-k` and `-H` options.

When counting in the multiplexing mode, the results must be normalized to be used. The default base used for the data normalization is the timebase. The **-b** option allows for the use of the PURR time or the SPURR time (when supported by the processor) for the data normalization. The base for the data normalization can also be defined using the **HPM_NORMALIZE** environment variable.

Results can be output in XML format using the **-x** option.

Flags

Item	Description
`-a`	Aggregates the counters on POE runs.
-b *time_base*	Selects a base for the data normalization. The available bases are as follows: **time** timebase **purr** PURR time (when available) **spurr** SPURR time (when available) The default value is **time**.
`-d`	Adds detailed set counts for counter multiplexing mode.

Item	Description
-D *metrics*	Selects a list of derived metrics to be evaluated. Each derived metric can be qualified by a counting mode as follows: `metric:counting_modes` (See the **-m** option for available counting modes.)
-g *event_groups*	Lists a predefined group of events or a comma-separated list of event group names or numbers. When a comma-separated list of groups is used, the counter multiplexing mode is selected. Each event group can be qualified by a counting mode as follows: `event_group:counting_modes` (See the **-m** option for available counting modes.)
-H	Adds hypervisor activity on behalf of the process.
-h	Displays help message.
-k	Adds system activity on behalf of the process.
-m *metrics_groups*	Selects a list of derived metric groups to be evaluated. The derived metric group refers to all derived metrics that do not belong to a specific derived metric group. Each metric group can be qualified by a counting mode as follows: `metric_group_name:counting_modes` The available counting modes are as follows: **u** user mode **k** kernel mode **h** hypervisor mode **r** runlatch mode **n** nointerrupt mode
-o *file*	Output file name.
-s *set*	Lists a predefined set of events or a comma-separated list of sets (1 to *N*, or `0` to select all. See the `pmlist` command.) When a comma-separated list of sets is used, the counter multiplexing mode is selected. Each set can be qualified by a counting mode as follows: `event_set:counting_modes` (See the **-m** option for available counting modes.)
-x	Displays results in XML format.

Parameters

Item	Description
command	Specifies the executed program for which performance measurements are made.

Environment Variables

The following environment variables directly affect the execution of the hpmcount command (there are additional MP_* environment variables that influence the execution of parallel programs).

Item	Description
HPM_EVENT_SET	Selects one of the event sets. The value can be an integer from 1 to 6 on POWER3 systems, 1 to 4 on PowerPC 604 RISC Microprocessor systems, or 1 to a processor-dependent upper limit on POWER4 and later systems. This environment variable is also used to select an event group name on POWER4 and later systems. A comma-separated list of event sets can be specified. In this case, the counter multiplexing mode is selected. Each event set can be qualified by a counting mode as follows: `event_set_number:counting_modes` The **-g** or **-s** option takes precedence over this variable. The **HPM_EVENT_GROUP** environment variable takes precedence over this variable.
HPM_EVENT_GROUP	Selects the event groups. A comma-separated list of event groups can be specified. In this case, the counter multiplexing mode is selected. Each event group can be qualified by a counting mode as follows: `event_group_number:counting_modes` The **-g** or **-s** option takes precedence over this variable. The **HPM_EVENT_GROUP** environment variable takes precedence over the **HPM_EVENT_SET** variable.
HPM_NORMALIZE	Provides the base to be used for the data normalization. The **-b** option takes precedence over this variable.
HPM_PMD_GROUP	Specifies a comma-separated list of derived metric groups. Each metric group can be qualified by a counting mode. The **-m** option takes precedence over this variable.
HPM_PMD_METRIC	Specifies a comma-separated list of derived metrics. Each derived metric can be qualified by a counting mode. The **-D** option takes precedence over this variable.
HPM_DIV_WEIGHT	Provides a weight (an integer greater than 1) to be used to compute weighted flips on POWER4 systems.
MP_CHILD	Used in a parallel environment when aggregate counts are specified to complement the output results file name (*myID*), synchronize collation of results, and identify verbose/debug diagnostic messages more closely.
MP_PROCS	The number of program tasks.
HPM_AGGREGATE_OUTPUT	Aggregates counts on POE applications (forces the command line argument -a). With this flag, a single file performance file is generated for all tasks. This only works with POE or Load Leveller, and it requires the availability of a parallel file system (such as GPFS) on the system.
HPM_LOG_DIR	When this flag is set, hpmcount writes a hpm_log.*id* file with the performance data in the provided directory. This is in addition to the regular output.

Item	Description
MP_PARTITION	On POE applications, *id* is a POE ID provided by MP_PARTITION. Otherwise, it is the *pid*. Also names internal lock and data files.
HPM_MX_DURATION	When counting in counter multiplexing mode, this flag specifies the duration of each slice of time. It is expressed in ms, and must lie in the range of 10 ms - 30 s. When this flag is not set, the default value used for the time slice duration is 100 ms.

In addition, the following environment variables, supplied by the user, specify estimations of memory, cache, and TLB miss latencies for the computation of derived metrics. These environment variables do not take precedence over the same estimations eventually provided in the file HPM_flags.env, if present.

- HPM_MEM_LATENCY
- HPM_L3_LATENCY
- HPM_L35_LATENCY
- HPM_AVG_L3_LATENCY
- HPM_AVG_L2_LATENCY
- HPM_L2_LATENCY
- HPM_L25_LATENCY
- HPM_L275_LATENCY
- HPM_L1_LATENCY
- HPM_TLB_LATENCY

Exit Status

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

1. To run the ls command and write information concerning events in set 5 from hardware counters, enter:

   ```
   hpmcount -s 5 ls
   ```

2. To run the ls command and write information concerning events in sets 5, 2, and 9 from hardware counters using the counter multiplexing mode, enter:

   ```
   hpmcount -s 5,2,9 ls
   ```

3. To run the **ls** command and report derived metrics pertaining to the default and cpi_breakdown metric groups counted respectively in kernel+user+hypervisor mode and user mode, enter:

   ```
   hpmcount -m default:kuh,cpi_breakdown:u ls
   ```

Implementation Specifics

The hpmcount command uses the PMAPI thread-level API.

The hpmcount *command* parameter is not parsed as a command line for an application name with options. Instead, a shell script must be created that contains the command line.

Location

/usr/bin/perf/pmapi/hpmcount

Standard Input

Not used.

Standard Output

Performance monitoring results are written to stdout, unless the -o *file* option is specified on the command line.

Standard Error

Used only for diagnostic messages.

Files

The following input files are used if present.

Item	Description
libHPM_events	User-supplied event set file. This file does not take precedence over the command lines specified with the -s option. The format for a POWER3/PowerPC 604 RISC Microprocessor counter/event pair is *counternumber eventname*. For example: `0 PM_LD_MISS_L2HIT` `1 PM_TAG_BURSTRD_L2MISS` `2 PM_TAG_ST_MISS_L2` `3 PM_FPU0_DENORM` `4 PM_LSU_IDLE` `5 PM_LQ_FULL` `6 PM_FPU_FMA` `7 PM_FPU_IDLE` A comma-separated list of events can also be specified. This turns on the counter multiplexing mode: `0 PM_CYC,PM_FPU_FIN,PM_IC_MISS` `1 PM_LD_CMPL,PM_INST_CMPL,PM_DC_MISS` `2 PM_INST_CMPL,PM_FPU_WT,PM_INST_CMPL` `3 PM_LD_MISS_DC_XU,PM_CYC,PM_CYC` For a POWER4 event group name, the format is *event_group_name*. For example: `pm_hpmcount1` A comma-separated list of events can also be specified. This turns on the counter multiplexing mode: `pm_hpmcount1,pm_hpmcount2,pm_basic`
HPM_flags.env	File containing environment variable/value pairs used for the computation of derived metrics. For example: `HPM_L2_LATENCY 12` `HPM_EVENT_SET  5`
./.hpm_lockfile_*mp_partition*	Lock file. This file is reserved for the hpmcount command's internal use.
./.hpm_datafile_*mp_partition*	Accumulative results file. This file is reserved for the hpmcount command's internal use.

The following output files are used.

Item	Description
file_*myID*.*pid*	File specified with the -o option for hpmcount output results, where *myID* is taken from the MP_CHILD environment variable, with a default value of 0000.
HPM_LOG_DIR/hpm_log.MP_PARTITION or HPM_LOG_DIR/hpm_log.*pid*	Log file specified for aggregate counters on POE runs.
./.hpm_lockfile_*mp_partition*	Lock file. This file is reserved for the hpmcount command's internal use.

Item	Description
./.hpm_datafile_*mp_partition*	Accumulative results file. This file is reserved for the hpmcount command's internal use.

hpmstat Command

Purpose

Provides system-wide hardware performance counter information.

Syntax

hpmstat [**-b** *time_base*] [-d] [**-D** *metrics*] [**-g** *event_groups*] [-H] [-k] [**-m** *metrics_groups*] [-o *file*] [-r] [-s *set*] [-T] [-U] [-u] [**-x**] [**-@ ALL** | *WparName*] *interval count*

hpmstat [-h]

Description

The hpmstat command provides the execution wall clock time, hardware performance counters information, and derived hardware metrics. It can only be used by a user with root privilege.

When specified without command line options, hpmstat counts the default 1 iteration of user, kernel, and hypervisor (for processors supporting hypervisor mode) activity for 1 second for the default set 1 of events. It then writes the raw counter values and derived metrics to standard output. By default, runlatch is disabled so that counts can be performed while executing in idle cycle.

When the -U option is specified, *interval* is in microseconds, the iteration *count* is infinity, and derived metrics are not calculated and written to standard output. This option is ignored if the counter multiplexing mode is specified.

When the -T option is specified, output information is preceded by the time stamp (seconds plus microseconds) and timing information is written as time stamps instead of time in seconds.

Event types to be monitored and the associated hardware performance counters are specified using either the set -s option or by specifying an event group name or set number in the HPM_EVENT_SETone environment variable. Alternatively, specify counter/event pairs (POWER3 / PowerPC 604 RISC Microprocessor) or an event group name (POWER4 and later) in the libHPM_events input file (takes precedence over HPM_EVENT_SET). Each set can be qualified by a counting mode. An event group number or name can be specified by setting the **-g** option or specifying a comma-separated list of event groups in the **HPM_EVENT_GROUP** environment variable. In the same manner, each event group can be qualified by a counting mode.

A comma-separated list of event sets can be specified instead of a set number, in which case the counter multiplexing mode is selected. To select all event sets, set the set number value to 0.

Valid event set numbers run from 1 to an upper limit dependent upon the processor type, which can be listed using the pmlist command.

A comma-separated list of derived metrics can be specified by setting the **-D** option. Each derived metric can be qualified by a counting mode.

A list of derived metric groups can be specified by setting the **-m** option or by specifying a comma-separated list of derived metric groups in the **HPM_PMD_GROUP** environment variable. This allows to select all the derived metrics pertaining to the specified groups. Each metric group can be qualified by a counting mode.

When counting in the multiplexing mode, the results must be normalized to be used. The default base used for the data normalization is the timebase. The **-b** option allows for the use of the PURR time or the SPURR time (when supported by the processor) for the data normalization. The base for the data normalization can also be defined using the **HPM_NORMALIZE** environment variable.

When you run the **hpmstat** command from the global workload partition (WPAR), it is possible to monitor a specific WPAR using the **-@** *WparName* option. You can use the **-@ ALL** option to monitor all active WPARs in the system and to retrieve per-WPAR data.

Results can be output in XML format using the **-x** option.

Flags

Item	Description
-@ ALL | *WparName*	Selects the target WPAR in which the activity is to be measured. The **ALL** value means that the **hpmstat** command measures all active WPARs in the system and reports the activity for each WPAR. This option is only available when you run the **hpmstat** command from the global WPAR; it is ignored otherwise.
-b *time_base*	Selects a base for the data normalization. The available bases are as follows: **time** timebase **purr** PURR time (when available) **spurr** SPURR time (when available) The default value is **time**.
-d	Adds detailed set counts for counter multiplexing mode.
-D *metrics*	Selects a list of derived metrics to be evaluated. Each derived metric can be qualified by a counting mode as follows: `metric:counting_modes` (See the **-m** option for available counting modes.)
-g *event_groups*	Lists a predefined group of events or a comma-separated list of event group names or numbers. When a comma-separated list of groups is used, the counter multiplexing mode is selected. Each event group can be qualified by a counting mode as follows: `event_group:counting_modes` (See the **-m** option for available counting modes.)
-H	Counts hypervisor activity only.
-h	Displays help message.
-k	Counts system activity only.

Item	Description
-m *metrics_groups*	Selects a list of derived metric groups to be evaluated. The default derived metric group refers to all derived metrics that do not belong to a specific derived metric group. Each metric group can be qualified by a counting mode as follows: `metric_group_name:counting_modes` The available counting modes are as follows: **u** user mode **k** kernel mode **h** hypervisor mode **r** runlatch mode **n** nointerrupt mode
`-o` *file*	Output file name.
`-r`	Enables `runlatch` and disables counts while executing in idle cycle.
`-s` *set*	Lists a predefined set of events or a comma-separated list of sets (`1` to *N*, or `0` to select all. See the `pmlist` command.) When a comma-separated list of sets is used, the counter multiplexing mode is selected. Each set can be qualified by a counting mode as follows: `event_set:counting_modes` (See the **-m** option for available counting modes.)
`-T`	Writes time stamps instead of time in seconds.
`-U`	Puts counting time interval in microseconds. This option is ignored if the counter multiplexing mode is specified.
`-u`	Counts user activity only.
-x	Displays results in VPA XML format.

Parameters

Item	Description
interval	Displays the counting time interval in seconds or microseconds, with a default value of `1`.
count	Shows the number of iterations to count. The default is `1` with an interval in seconds, and infinity when the option `-U` is specified.

Environment Variables

The following environment variables directly affect the execution of the `hpmstat` command (there are additional `MP_*` environment variables that influence the execution of parallel programs).

Item	Description
HPM_EVENT_SET	Selects one of the event sets. The value can be an integer from 1 to 6 on POWER3 systems, 1 to 4 on PowerPC 604 RISC Microprocessor systems, or 1 to a processor-dependent upper limit on POWER4 and later systems. This environment variable is also used to select an event group name on POWER4 and later systems. Each event set can be qualified by a counting mode as follows: `event_set_number:counting_modes` The **-g** or **-s** option takes precedence over this variable. The **HPM_EVENT_GROUP** environment variable takes precedence over this variable.
HPM_EVENT_GROUP	Selects the event groups. A comma-separated list of event groups can be specified. In this case, the counter multiplexing mode is selected. Each event group can be qualified by a counting mode as follows: `event_group_number:counting_modes` The **-g** or **-s** option takes precedence over this variable. The **HPM_EVENT_GROUP** environment variable takes precedence over the **HPM_EVENT_SET** variable.
HPM_NORMALIZE	Provides the base to be used for the data normalization. The **-b** option takes precedence over this variable.
HPM_PMD_GROUP	Specifies a comma-separated list of derived metric groups. Each metric group can be qualified by a counting mode. The **-m** option takes precedence over this variable.
HPM_PMD_METRIC	Specifies a comma-separated list of derived metrics. Each derived metric can be qualified by a counting mode. The **-D** option takes precedence over this variable.
HPM_DIV_WEIGHT	Provides a weight (an integer greater than 1) to be used to compute weighted flips on POWER4 systems.
HPM_MX_DURATION	When counting in counter multiplexing mode, this flag specifies the duration of each slice of time. It is expressed in ms, and must lie in the range of 10 ms - 30 s. When this flag is not set, the default value used for the time slice duration is 100 ms.

In addition, the following environment variables, supplied by the user, specify estimations of memory, cache, and TLB miss latencies for the computation of derived metrics. These environment variables do not take precedence over the same estimations eventually provided in the file `HPM_flags.env`, if present.

- HPM_MEM_LATENCY
- HPM_L3_LATENCY
- HPM_L35_LATENCY
- HPM_AVG_L3_LATENCY
- HPM_AVG_L2_LATENCY
- HPM_L2_LATENCY
- HPM_L25_LATENCY
- HPM_L275_LATENCY
- HPM_L1_LATENCY

- HPM_TLB_LATENCY

Exit Status

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To write information for system, user, and hypervisor activity over a 1 second interval concerning events in set 2 from hardware counters, enter the following command:

   ```
   hpmstat -s 2
   ```

2. To write information for the user activity concerning events of group 0 and the system activity concerning events of group 1 for the wpar1 WPAR over a five-second interval, enter the following command:

   ```
   hpmstat -@ wpar1 -g 0:u,1:k 5
   ```

Location

/usr/bin/perf/pmapi/hpmstat

Standard Input

Not used.

Standard Output

Performance monitoring results are written to stdout, unless the -o *file* option is specified on the command line.

Standard Error

Used only for diagnostic messages.

Files

The following input files are used if present.

Item	Description
`libHPM_events`	User-supplied event set file. This file does not take precedence over the command lines specified with the `-s` option. The format for a POWER3/PowerPC 604 RISC Microprocessor counter/event pair is *countername eventname*. For example: `0 PM_LD_MISS_L2HIT` `1 PM_TAG_BURSTRD_L2MISS` `2 PM_TAG_ST_MISS_L2` `3 PM_FPU0_DENORM` `4 PM_LSU_IDLE` `5 PM_LQ_FULL` `6 PM_FPU_FMA` `7 PM_FPU_IDLE` For a POWER4 event group name, the format is *event_group_name*. For example: `pm_hpmcount1`
`HPM_flags.env`	File containing environment variable/value pairs used for the computation of derived metrics. For example: `HPM_L2_LATENCY 12` `HPM_EVENT_SET  5`

The following output files are used.

Item	Description
file	File specified with the `-o` option for `hpmstat` output results.

hps_dump Command

Purpose

Dumps contents of Network Terminal Accelerator (NTX) adapter memory to a host file.

Syntax

hps_dump [**-f** *Name*] [**-d** *Device*]

Description

The **hps_dump** command uses the loader interface to upload all of the memory from the adapter board into a file. It produces a snapshot of a system for later analysis and debugging. The first 1024 bytes of the file contains the following items:

Item	Description
80	Identification string, includes version.
80	Time and date of memory dump from host system.
80	Comments.

Item	Description
268	Log table from the host adapter.
32	System address table.
8	Starting and ending address range of memory dump.
476	Padding to 1024 bytes total.

Flags

Item	Description
-f *Name*	Specifies the name of the memory dump. Use this option to override the default file name **./hpscore**.
-d *Device*	Specifies the raw device file name of the adapter. Use this option to override the default device name **/dev/rhp0**.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Auditing Events: Not applicable.

Examples

1. To get a memory dump of the default adapter to the file **hpscore** in the current directory, enter:

   ```
   hps_dump
   ```

2. To get a memory dump of the default adapter to the file hpsdebug in the current directory of the default adapter, enter:

   ```
   hps_dump -f hpsdebug
   ```

3. To get a memory dump of memory of the adapter **/dev/rhp1** to the file **hpsdebug** in the current directory of the default adapter, enter:

   ```
   hps_dump -f hpsdebug -d /dev/rhp1
   ```

Files

Item	Description
/usr/bin/hps_dump	Contains the **hps_dump** command.

Item	Description
/dev/rhp0	Default NTX raw device file name.

htable Command

Purpose

Converts host files to the format used by network library routines.

Syntax

/usr/sbin/htable [**-c** *connected-nets*] [**-l** *local-nets*] *input-file*

Note: Do not put a space on either side of the comma.

Description

The **htable** command converts host files in the format specified in RFC 810 to the format used by the network library routines. The conversion creates three files: the **/etc/hosts** file, the **/etc/networks** file, and the **/etc/gateways** file.

The **gethostbyname** subroutine uses the **hosts** file for mapping host names to addresses when the **named** daemon is not used. The **getnetent** subroutine uses the **networks** file for mapping network names to numbers.

The **gateways** file may be used by the **routed** daemon in identifying passive Internet gateways.

If any local **hosts**, **networks**, or **gateways** files (**localhosts**, **localnetworks**, or **localgateways** respectively) exist in the current directory, that file's contents are added as a prefix to the output file. Of these, the **htable** program only interprets the **gateways** file. Adding the prefix to the contents allows sites to maintain local entries that are not normally present in the master database.

Flags

Item	Description
-c *connected-nets*	Specifies a list of networks to which the host is directly connected if the network routing daemons use the **gateways** file. Separate the networks with commas, and use the network name or standard Internet dot notation (for example, `-c arpanet,128.32,LocalEthernet`). The **htable** command only includes gateways that are directly connected to one of the networks specified or that can be reached from another gateway on a connected network.
-l *local-nets*	Specifies a list of networks for the **htable** command to treat as local. Take information about hosts on local networks only from the **localhosts** file. Separate the networks with commas, and use the network name or standard Internet dot notation (for example, `-l 128.32,local-ether-net`). Entries for local hosts from the main database are omitted so that the **localhosts** file can override entries in the input file (the file you specify on the command line).

Files

Item	Description
/*CurrentDirectory***/localgateways**	Contains local gateway information.
/*CurrentDirectory***/localhosts**	Contains local host name information.

Item	Description
/CurrentDirectory/**localnetworks**	Contains local network information.

hty_load Command

Purpose

Displays or downloads Network Terminal Accelerator (NTX) adapter configurations.

Syntax

hty_load [**-d** *Device*] [**-f** *ConfigFileName*]

Description

The **hty_load** command displays or downloads adapter configurations. If you issue this command without any flags, the system displays the current adapter configuration for the **/dev/rhp0** device file. Given a *Device* parameter, the **hty_load** command loads a configuration file into the tty driver. The tty driver uses the file to configure both the host presentation services (HPS) and the adapters.

Typically, the **hty_load** command is invoked from the **/etc/rc.ntx** file.

The Configuration File

The **hty_load** command uses a single configuration file to configure the adapters. Each entry is on a separate line. Entries are separated by new-line characters. Fields in an entry are separated by tabs or space characters. Entries in the configuration file have the following fields.:

```
MinorNumber Cluster NumberOfPorts
```

These fields have the following values:

Item	Description
MinorNumber	Specifies the board's minor device number.
Cluster	This field is always 1.

Item	Description
NumberOfPorts	Specifies the number of hty devices. The number depends on the model of adapter you are using. The number of available channels is from 1 to 256 for a 2MB board or from 1 to 2048 for an 8MB board.

The configuration file also supports comments. Comment lines begin with a # (pound sign). Everything to the right of the comment character is ignored. Comment lines end with new-line characters.

Flags

Item	Description
-d *Device*	Specifies the raw device file name of the adapter. Use this option to override the default device name **/dev/rhp0**.
-f *ConfigFileName*	Specifies the driver configuration file name. The default configuration file is the **/etc/hty_config** file.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Auditing Events: N/A

Examples

To load the system configuration and use the default driver configuration file, enter:

```
hty_load -d /dev/rhp0
```

Files

Item	Description
/usr/bin/hty_load	Contains the **hty_load** command.
/etc/rc.ntx	Invokes the **hty_load** command.
/etc/hty_config	Default NTX driver configuration file name.
/dev/rhp0	Default NTX raw device file name.

hyphen Command

Purpose

Finds hyphenated words.

Syntax

hyphen [*File* ...]

Description

The **hyphen** command reads one or more English-language files, finds all the lines ending with hyphenated words, and writes those words to standard output. The parameter *File* specifies English-language files to be read by the **hyphen** command. The default is standard input. If no file is specified or if the - (hyphen) is specified as the last file name, the **hyphen** command reads standard input. The **hyphen** command can be used as a filter.

Note: The **hyphen** command cannot read hyphenated words that are italic or underlined. The **hyphen** command sometimes gives unnecessary output.

Examples

To check the hyphenation performed by a text-formatting program on a file, enter:

```
mm [Flag...] [File...] | hyphen
```

i

The following AIX commands begin with the letter *i*.

ibm3812 Command

Purpose

Postprocesses the **troff** command output for the IBM 3812 Model 2 Pageprinter.

Syntax

ibm3812 [**-altpaper**] [**-landscape**] [**-quietly**] [**-F***Directory*] [**-i**] [*File*...]

Description

The **ibm3812** command is a postprocessor that can be used on intermediate output produced by the **troff** command.

Note: An entire page is placed in memory before it is printed.

If given one or more file names as options, the **ibm3812** command processes those files. If no file names are specified, this command acts as a filter interpreting standard input.

The **ibm3812** command's font files allow the postprocessor to send characters of more than one byte to the printer. These can be characters that require multiple bytes to represent them, such as code page and point; or, they can be characters that are composed of two or more concatenated glyphs.

For example, the character code for the \(ib (improper subset) special character is:

```
"\001\125\xe2\xff\xe8\xe3%\x00\x16\001\074\xe3\xff\xea"
```

The printer is in Page Map Primitive (PMP) mode when these bytes are sent, so you must use the 001 directive to introduce a character. For single-byte codes, this Generic Font Patterns command is automatically handled by the postprocessor. The % (percent sign) characters escape the bytes containing 0, which would otherwise terminate the code sequence. To obtain a literal % character, escape it with another % character so that a percent sign is displayed as %%. A single-byte % code is assumed to be a literal percent sign, so that the single-byte % character needs no special handling in the font file.

Notes:

1. The **ibm3812** command depends on the files with names ending in **.out** in the **/usr/lib/font/devibm3812** directory. It does not produce usable output unless these files have been properly set up.
2. The postprocessor requires additional font information to be stored in the **/usr/lib/font/devibm3812/fonts** file. If new fonts are added to this file, make sure that the **DESC** file is also updated to reflect the additional fonts and special characters.

The format of the file must be preserved. The file contains the following four fields:

- The one- or two-letter name of the font
- The full name of the font on the printer-font diskette
- The one- or two-letter name of the substitute font
- An array of five available sizes.

Flags

Item	Description
-altpaper	Specifies that the file should be printed from the alternate paper drawer. By default, the **ibm3812** command prints from the primary paper drawer.
-landscape	Specifies that the file should be printed in landscape orientation, so that the wider part of the paper is horizontally oriented. This flag rotates the page to the right by 90 degrees. By default, the **ibm3812** command prints in portrait orientation.
-quietly	Suppresses all non-fatal error messages.
-F*Directory*	Specifies the directory holding the font files. The default file is **devibm3812**. The command looks for font files in the **/usr/lib/font** directory by default.
-i	Suppresses initialization of the printer that runs the PMP.init macro, after the job has printed.

Example

Following is an example of the **troff** command used with the **ibm3812** command:

```
troff file|ibm3812|qprt-dp
```

Files

Item	Description
/usr/lib/font/devibm3812/*.out	Contains font files for the **ibm3812** command.
/usr/lib/font/devibm3812/fonts	Contains information about the available fonts for the **ibm3812** command.

ibm3816 Command

Purpose

Postprocesses the **troff** command output for the IBM 3816 Pageprinter.

Syntax

ibm3816 [**-altpaper**] [**-landscape**] [**-quietly**] [**-F***Directory*] [**-i**] [*File*...]

Description

The **ibm3816** command is a postprocessor that can be used on intermediate output produced by the **troff** command.

Note: An entire page is placed in memory before it is printed.

If given one or more file names as options, the **ibm3816** command processes those files. If no file names are specified, this command acts as a filter interpreting standard input.

The **ibm3816** command's font files allow the postprocessor to send characters of more than one byte to the printer. These can be characters that require multiple bytes to represent them, such as code page and point; or, they can be characters that are composed of two or more concatenated glyphs.

For example, the character code for the \(ib (improper subset) special character is:

```
"\001\125\xe2\xff\xe8\xe3%\x00\x16\001\074\xe3\xff\xea"
```

The printer is in Page Map Primitive (PMP) mode when these bytes are sent, so you must use the 001 directive to introduce a character. For single-byte codes, this Generic Font Patterns command is automatically handled by the postprocessor. The % (percent sign) characters escape the bytes containing 0, which would otherwise terminate the code sequence. To obtain a literal % character, escape it with another % character so that a percent sign is displayed as %%. A single-byte % code is assumed to be a literal percent sign, so that the single-byte % character needs no special handling in the font file.

Notes:

1. The **ibm3816** command depends on the files with names ending in **.out** in the **/usr/lib/font/devibm3816** directory. It does not produce usable output unless these files have been properly set up.
2. The postprocessor requires additional font information to be stored in the **/usr/lib/font/devibm3816/fonts** file. If new fonts are added to this file, make sure that the **DESC** file is also updated to reflect the additional fonts and special characters.

The format of the file must be preserved. The file contains the following four fields:

- The one- or two-letter name of the font
- The full name of the font on the printer-font diskette
- The one- or two-letter name of the substitute font
- An array of five available sizes.

Flags

Item	Description
-altpaper	Specifies that the file should be printed from the alternate paper drawer. By default, the **ibm3816** command prints from the primary paper drawer.
-landscape	Specifies that the file should be printed in landscape orientation, so that the wider part of the paper is horizontally oriented. This flag rotates the page to the right by 90 degrees. By default, the **ibm3816** command prints in portrait orientation.
-quietly	Suppresses all non-fatal error messages.
-F*Directory*	Specifies the directory holding the font files. The default file is **devibm3816**. The command looks for font files in the **/usr/lib/font** directory by default.
-i	Suppresses initialization of the printer that runs the PMP.init macro, after the job has printed.

Example

Following is an example of the **troff** command used with the **ibm3816** command:

```
troff file|ibm3816|qprt-dp
```

Files

Item	Description
/usr/lib/font/devibm3816/*.out	Contains font files for the **ibm3816** command.
/usr/lib/font/devibm3816/fonts	Contains information about the available fonts for the **ibm3816** command.

ibm5585H-T Command

Purpose

Processes **troff** command output for the IBM 5585H-T printer.

Syntax

ibm5585H-T [**-F***Directory*] [*File*]

Description

The **ibm5585H-T** command processes the output of the **troff** command for output to the IBM 5585H-T printer for traditional Chinese language. This command is provided exclusively for traditional Chinese language support.

The **ibm5585H-T** command processes one or more files specified by the *File* parameter. If no file is specified, the **ibm5585H-T** command reads from standard input.

The **ibm5585H-T** command uses font files in the **/usr/lib/font/devibm5585H-T** directory that have command names ending with **.out**. The **ibm5585H-T** command does not produce correct output unless these files are provided.

Flag

Item	Description
-F*Directory*	Specifies a directory name as the place to find font files. By default, the **ibm5585H-T** command looks for font files in the **/usr/lib/font/devibm5585H-T** directory.

Example

To process the `reports` file for the IBM 5585H-T printer, enter:

```
troff reports |ibm5585H-T | qprt -dp
```

The **ibm5585H-T** command first processes the output of the **troff** command, then sends the file to a print queue.

File

Item	Description
/usr/lib/font/devibm5585H-T/*.out	Contains font files.

ibm5587G Command

Purpose

Postprocesses **troff** command output for the IBM 5587-G01, 5584-H02, 5585-H01, 5587-H01, and 5589-H01 printers with the (32x32/24x24) cartridge installed. This command is used exclusively for Japanese Language Support.

Syntax

ibm5587G [**-F***Directory*] [**-quietly**] [*File* ...]

Description

The **ibm5587G** command processes the output of the **troff** command for output to the 5587-G01, 5584-H02, 5585-H01, 5587-H01, and 5589-H01 printers.

If given one or more files as options, the **ibm5587G** command processes those files. If no files are specified, it acts as a filter interpreting standard input.

> **Note:** The **ibm5587G** command assumes that the (32x32/24x24) cartridge is installed in the printer. Incorrect output from the printer may result if the wrong cartridge is installed in the printer.

The **ibm5587G** command depends on the files with names ending in **.out** in the **/usr/lib/font/devibm5587G** directory. It does not produce reasonable output unless these files have been properly set up.

Flags

Item	Description
-F*Directory*	Specifies a directory name as the place to find the font files. By default, the **ibm5587G** command looks for font files in the **/usr/lib/font/devibm5587G** directory.
-quietly	Suppresses all nonfatal error messages.

Files

Item	Description
/usr/lib/font/devibm5587G/*.out	Contains font files.

ibstat Command

Purpose

Displays operational information about one or more InfiniBand network devices.

Syntax

`ibstat` [`-d`, `-h`, `-i`, `-n`, `-p`, `-v`] [*DeviceName*]

Description

This command displays InfiniBand operational information pertaining to a specified Host Channel Adapter Device (HCAD). If an HCAD device name is not entered, status for all available HCADs are displayed. Select a flag to narrow down your search results. You can display specific categories of information, including Node, Port, Interface, and Debug information. You can also choose to display all of the information categories.

Flags

Item	Description
`-d`	Displays current debug setting.
`-h`	Displays `ibstat` command usage.
`-i`	Displays network interface information.
`-n`	Displays only IB node information.

Item	Description
-p	Displays only IB port information.
-v	Displays all IB device information.

The following fields display information for all valid calls:

Device Name
Displays the name of an available HCAD (for example, `iba0`).

Port State
Displays the current state of each HCAD port.

Down
Port is disabled.

Initialized
Port is enabled and issuing training sequences.

Armed
Port is trained and attempting to configure to the active state.

Active
Port is in a normal operational state.

Unknown
Port is in an invalid or unknown state.

Parameters

Item	Description
DeviceName	Specifies the name of the HCAD device (for example, `iba0`) **Tip:** The device name is optional. If you do not specify a device name, all InfiniBand devices are queried for control or information.

Exit Status

When you specify an invalid *DeviceName*, the `ibstat` command produces error messages stating that it could not connect to the device. For example:

```
IBSTAT: No device iba2 configured.
```

or:

```
IBSTAT: Device iba3 is not available.
```

Examples

1. To request node and port information, enter:

   ```
   ibstat -n -p
   ```

   Information similar to the following is displayed:

   ```
   ================================================================================
    INFINIBAND DEVICE INFORMATION (iba0)
   ================================================================================

   --------------------------------------------------------------------------------
    IB NODE INFORMATION (iba0)
   --------------------------------------------------------------------------------
   ```

```
Number of Ports:                         2
Globally Unique ID (GUID):               00.02.55.00.00.00.46.00
Maximum Number of Queue Pairs:           1023
Maximum Outstanding Work Requests:       32768
Maximum Scatter Gather per WQE:          252
Maximum Number of Completion Queues:     1023
Maximum Multicast Groups:                256
Maximum Memory Regions:                  3836
Maximum Memory Windows:                  3836

------------------------------------------------------------------------------
 IB PORT 1 INFORMATION (iba0)
------------------------------------------------------------------------------
Global ID Prefix:                        fe.80.00.00.00.00.00.00
Local ID (LID):                          0012
Port State:                              Active
Maximum Transmission Unit Capacity:      2048
Current Number of Partition Keys:        1
Partition Key List:
  P_Key[0]:                              ffff
Current Number of GUID's:                1
Globally Unique ID List:
  GUID[0]:                               00.02.55.00.00.00.46.12

------------------------------------------------------------------------------
 IB PORT 2 INFORMATION (iba0)
------------------------------------------------------------------------------
Global ID Prefix:                        fe.80.00.00.00.00.00.00
Local ID (LID):                          0011
Port State:                              Active
Maximum Transmission Unit Capacity:      2048
Current Number of Partition Keys:        1
Partition Key List:
  P_Key[0]:                              ffff
Current Number of GUID's:                1
Globally Unique ID List:
  GUID[0]:                               00.02.55.00.00.00.46.52
```

Location

`/usr/sbin/ibstat`

iconv Command

Purpose

Converts the encoding of characters from one code page encoding scheme to another.

Syntax

iconv [`-cs`] **-f** *FromCode* **-t** *ToCode* [*FileName*...]

`iconv -l`

Description

The **iconv** command converts the encoding of characters read from either standard input or the specified file from one coded character set to another and then writes the results to standard output. The input and output coded character sets are identified by the *FromCode* and *ToCode* parameters. The input data should consist of characters in the code set specified by the *FromCode* parameter. If the *FileName* parameter is not specified on the command line, the **iconv** command reads from standard input.

You can use the System Management Interface Tool (SMIT) **smit iconv** fast path to run this command. The `iconv` command uses the LOCPATH environment variable to search for code-set converters of the form `iconv/`*FromCodeSet_ToCodeSet*. The default value of LOCPATH is `/usr/lib/nls/loc`.

Flags

Item	Description
-c	Omits characters that cannot be converted in the input file from the output. Characters that cannot be converted include characters that are invalid in the *FromCode* of the input or that have no corresponding character in the *ToCode* of the output. After omitting an unconvertible character, `iconv` advances to the next byte of the input to convert the next character. If `-c` is not used, `iconv` exits upon encountering a character that cannot be converted in the input. The presence or absence of `-c` does not affect the exit status of `iconv`.
-f *FromCode*	Specifies the code set in which the input data is encoded. The space between the **-f** flag and the *FromCode* parameter is optional.
-l	Writes all supported *FromCode* and *ToCode* values to standard output.
-s	Suppresses any messages written to standard error concerning invalid characters. When -s is not used, an error message is written to standard error for each unconvertible or truncated character. The presence or absence of `-s` does not affect the exit status of `iconv`.
-t *ToCode*	Specifies the code set to which the output data is to be converted. The space between the **-t** flag and the *ToCode* parameter is optional.
FileName	Specifies a file to be converted.

Exit Status

This command returns the following exit values:

Item	Description
0	Input data was successfully converted.
1	The specified conversions are not supported; the given input file cannot be opened for read; or there is a usage-syntax error.
2	An unusable character was encountered in the input stream.

Examples

1. To convert the contents of the **mail.x400** file from code set IBM-850 and store the results in the **mail.local** file, enter:

```
iconv -f IBM-850 -t ISO8859-1 mail.x400 > mail.local
```

2. To convert the contents of the **mail.japan** file from the 7-bit interchange (ISO2022) encoding to the Japanese EUC code set (IBM-eucJP), enter:

```
iconv -f fold7 -t IBM-eucJP mail.japan > mail.local
```

3. To convert the contents of a local file to the mail-interchange format and send mail, enter:

```
iconv -f IBM-943 -t fold7 mail.local | mail fxrojas
```

id Command

Purpose

Displays the system identifications of a specified user.

Syntax

id [user]

id **- G** [**-n**] [*User*]

id **-g** [**-n l** | [**-n -r**] [*User*]

id **-u** [**-n l** | [**-n r**] [*User*]

Description

The **id** command writes to standard output a message containing the system identifications (ID) for a specified user. The system IDs are numbers which identify users and user groups to the system. The **id** command writes the following information, when applicable:

- User name and real user ID
- Name of the user's group and real group ID
- Name of user's supplementary groups and supplementary group IDs

  Supplementary group information is written only for systems supporting multiple-user groups and only if the specified user belongs to a supplementary group.

The **id** command also writes effective user and group IDs, but only for the user that invoked the **id** command. (If the *User* parameter is specified with the **id** command, the effective IDs are assumed to be identical to real IDs.) If the effective and real IDs for the invoking user are different, the **id** command writes the following effective ID information, when applicable:

- Effective user name and effective user ID
- Name of effective user's group and effective group ID

The id command, when specified with the -l option, displays login UID. Login ID indicates the system credentials at the time of logging in to the session. Login UID indicates the user ID (numeric value) of the user, who actually logged in. The login UID is equal to the UID for a user who has logged in to the system and whose credentials remain unchanged. For example, when the user runs the su command, the UID for the user changes and the login UID remains the same.

The **id** command will fail if the specified user does not exist or if the command cannot read the user or group information.

Flags

The contents and format of the message written by the **id** command can be altered with the following flags:

Item	Description
-G	Specifies that the **id** command write the effective, real, and supplementary group IDs only. If there are multiple entries for the effective, real, or supplementary IDs, they are separated by a space and placed on the same line.
-g	Specifies that the **id** command write only the effective group ID.
-u	Specifies that the **id** command write only the effective user ID.
-r	Specifies that the **id** command write the real ID instead of the effective ID. This flag can be invoked with either the **-g** flag to write the real group ID, or the **-u** flag to write the real user ID.
-n	Specifies that the **id** command outputs the name, instead of the ID number, when it is specified with the **-G**, **-g**, and **-u** flags.

Item	Description
-l	Specifies that the `id` command write the login ID instead of the real or effective ID. This flag can be invoked with either the `-u` flag to write the login UID or the `-g` flag to write the primary group ID for the login user. When *username* is passed with the `-l` option, the `id` command displays the ID details of the user name instead of the login ID details.
User	Specifies the login name of a user for the **id** command. If no user is specified, the user invoking the **id** command is the default.

Security

Access Control: This program should be installed as a normal user program in the Trusted Computing Base.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To display all system identifications for the current user, enter:

   ```
   id
   ```

   Output for the **id** command is displayed in the following format:

   ```
   uid=1544(sah) gid=300(build) euid=0(root) egid=9(printq) groups=0(system),10(audit)
   ```

   In this example, the user has user name `sah` with an ID number of `1544`; a primary group name of `build` with an ID number of `300`; an effective user name of `root` with an ID number of `0`; an effective group name of `printq` with an ID number of `9`; and two supplementary group names of `system` and `audit`, with ID numbers `0` and `10`, respectively.
2. To display all group ID numbers for the current user, enter:

   ```
   id -G
   ```

   Output is displayed in the following format:

   ```
   0 10 300 9
   ```

   The **-G** flag writes only the group IDs for a user. In this example, user `sah` is a member of the system (`0`), audit (`10`), build (`300`), and printq (`9`) groups.
3. To display all group names for the current user, enter:

   ```
   id -Gn
   ```

   Output is displayed in the following format:

   ```
   system audit build printq
   ```

   The **-n** flag writes only the names instead of the ID numbers.
4. To display the real group name for the current user, enter:

```
id -gnr
```

Output is displayed in the following format:

```
build
```

5. To display the login UID after logging in as root and running the su command to user sah, type:

```
id -lu
```

Output is displayed in the following format:

```
0
```

6. To display the primary group name of the user who actually logged in, type:

```
id -lgn
```

Output is displayed in the following format:

```
system
```

7. To display the primary group ID of the user who actually logged in, type:

```
id -lg
```

Output is displayed in the following format:

```
0
```

Files

Item	Description
/usr/bin/id	Contains the **id** command.

ifconfig Command

Purpose

Configures or displays network interface parameters for a network by using TCP/IP.

Syntax

ifconfig *interface* [*addressfamily* [*address* [*destinationaddress*]] [*parameters*...]]

ifconfig *interface* [*protocolfamily*] *interface protocolfamily*

ifconfig **-a** [**-l**] [**-d**] [**-u**] [*protocolfamily*]

ifconfig *interface* [**tcp_low_rto** *rto* | **-tcp_low_rto**]

Description

You can use the **ifconfig** command to assign an address to a network interface and to configure or display the current network interface configuration information. The **ifconfig** command must be used at system startup to define the network address of each interface present on a system. After system startup, it can also be used to redefine an interfaces address and its other operating parameters. The network interface configuration is held on the running system and must be reset at each system restart. The **ifconfig** command interprets the **IFF_MULTICAST** flag and prints its value if it is set.

An interface can receive transmissions in differing protocols, each of which might require separate naming schemes. It is necessary to specify the *addressfamily* parameter, which might change the interpretation of the remaining parameters. The address families that are currently supported are **inet** and **inet6**.

For the DARPA-Internet family, **inet**, the address is either a host name present in the host name database, that is, the **/etc/hosts** file, or a DARPA-Internet address that is expressed in the Internet standard dotted decimal notation.

While any user can query the status of a network interface, only a user who has administrative authority can modify the configuration of those interfaces.

The **ifconfig** function displays the current configuration for a network interface when no optional parameters are supplied.

If a protocol family is specified, **ifconfig** reports only the details specific to that protocol family.

Only a super user can modify the configuration of a network interface.

Gratuitous ARP is supported for Ethernet, token-ring, and FDDI interfaces. This support means when an IP address is assigned, the host sends an ARP request for its own address (the new address) to inform other systems of its address so that they can update their ARP entry immediately. It also lets hosts detect duplicate IP address. If you get a response to the ARP request, an error is logged in **/var/adm/ras/errlog**, which can be viewed by using **errpt** command (or by using SMIT interface) for the error ID **AIXIF_ARP_DUP_ADDR**.

The **ifconfig** command calls the **ifconfig.ib** command. The **ifconfig.ib** command displays the interface information as shown:

```
ib0:flags=e3a0063<UP,BROADCAST,NOTRAILERS,RUNNING,ALLCAST,MULTICAST,GROUPRT>
pmtu 2048 qkey
0x1e qpn 0x48 lid 0x5c hca iba0 port 1 inet 1.2.3.120 netmask 0xffffff00 broadcast
1.2.3.255 inet6
fe80::2:c903:1:1e8d/64 tcp_sendspace 262144 tcp_recvspace 262144 rfc1323 1
```

The interface now displays the path mtu (pmtu), Queue Key (qkey), Queue Pair Number (qpn), Local ID (lid), Host Channel Adapter (hca), and the port number (port). This information is useful during debugging or performing diagnostics.

Note: Any changes that are made to the attributes of an interface by using the **ifconfig** command are lost when the system is rebooted.

When the **detach** command is specified with the **ifconfig** command, all other options are ignored. Specifying the **detach** command prevents the garbage character in the **ipconfig** command from causing any error. For example, the `ifconfig en3 garbage detach` command runs properly even when with the bad option.

Flags

Item	Description
-a	Optionally, the **-a** flag can be used instead of an interface name. This flag instructs **ifconfig** to display information about all interfaces in the system.
-d	The **-d** flag displays interfaces that are down. You can use the flag only with the **-a** or **-l** flag.
-l	This flag can be used to list all available interfaces on the system, with no other additional information. Use of this flag is mutually exclusive with all other flags and commands, except for **-d** and **-u**.
-u	The **-u** flag displays interfaces that are up. You can use the flag only with the **-a** or **-l** flag.
protocolfamily	This flag specifies protocols such as **tcp**, **udp**, **tcp6**, **udp6**, **icmp**, and **icmp6**.

Parameters

Item	Description
address	Specifies the network address for the network interface. For the **inet** family, the *address* parameter is either a host name or an IP address in the standard dotted decimal notation.
addressfamily	Specifies which network address family to change. The **inet** and **inet6** address families are currently supported. This parameter defaults to the **inet** address family.
destinationaddress	Specifies the address of the correspondent on the remote end of a point-to-point link.
interface	Specifies the network interface configuration values to show or change. You must specify an interface with the *interface* parameter when you use the **ifconfig** command. Abbreviations for the interfaces include: • at for asynchronous transfer mode (ATM) • en for Standard Ethernet (inet) • et for IEEE 802.3 Ethernet (inet) • gre for generic routing encapsulation tunnel pseudointerface (inet) • gif for IPv4-over-IPv6 tunnel pseudointerface (inet) • tr for token-ring (inet) • xt for X.25 (inet) • sl for serial line IP (inet) • lo for loopback (inet) • op for serial (inet) • vi for virtual IP address (inet) • ib for IP over InfiniBand (inet) • tap for TAP pseudo-Ethernet Include a numeral after the abbreviation to identify the specific interface (for example, `tr0`). If *interface* is not yet loaded, **ifconfig** *interface* loads that interface and **netstat -in** lists it. In processing a status query for *interface*, that interface is loaded (if not already loaded) to complete the query processing.
parameter	Allows the following parameter values: **alias** Establishes an additional network address for the interface. When changing network numbers, this parameter is useful for accepting packets that are addressed to the old interface. **allcast** Sets the token-ring interface to broadcast to all rings on the network. **-allcast** Confines the token-ring interface to broadcast only to the local ring. **anycast** (inet6 only) Adds the specified anycast address.

Item	Description
	-anycast (inet6 only) Deletes the specified anycast address.
	arp Enables the **ifconfig** command to use the Address Resolution Protocol in mapping between network-level addresses and link-level addresses. The **arp** value is the default.
	-arp Disables the use of the address resolution protocol.
	authority Reserved for future use.
	bridge Reserved for future use.
	-bridge Reserved for future use.
	broadcast *Address* (**inet** only) Specifies the address to use to broadcast to the network. The default broadcast address has a host part of all 1's.
	checksum_offload Enables the flag to indicate that the transmit TCP checksum will be offloaded to the adapter. The command also resets the per-interface counter that determines whether TCP must dynamically enable or disable offloading of checksum computation.
	-checksum_offload Disables transmit TCP checksum offloading.
	create (TAP only) Creates a network interface. You can either create a specific interface, such as **tap0** or specify the **tap** option to create the next available TAP interface, such as **ifconfig tap create**.
	-dad (inet6 only) Does not perform duplicate IPv6 address detection.
	-debug Disables driver-dependent debug code.
	delete Removes the specified network address. This command is used when an alias is incorrectly specified or when it is no longer needed. Incorrectly setting an **ns** address has the side effect of specifying the host portion of the network address. Removing all **ns** addresses lets you specify the host portion again.
	destroy (TAP only) Destroys a network interface. The **ifconfig** command removes the specified network interface from the list of interfaces. It also removes the interface and any associated TAP network device from the Object Data Manager (ODM).

Item	Description
device *dev_name*	(ATM network interface only). Specifies the device name that this interface is associated with. Unlike token ring or Ethernet, in case of ATM, there is not a one-to-one correspondence between an interface and a device. In the case of ATM, there can be more than one interface for every device.
detach	Removes an interface from the network interface list. If the last interface is detached, the network interface driver code is unloaded. For the interface route of an attached interface to be changed, that interface must be detached and added again with **ifconfig**.
down	Marks an interface as inactive (**down**), which keeps the system from trying to transmit messages through that interface. If possible, the **ifconfig** command also resets the interface to disable the reception of messages. Routes that use the interface, however, are not automatically disabled.
eui64	(inet6 only) Computes real IPv6 address by replacing the last 64 bytes of the specified address with the Interface Identifier.
first	Puts an IPv6 address at the first place on an interface to select it as the source for unbound sockets. The syntax for using this parameter is as follows: `ifconfig interface inet6 first address`
firstalias	Same as alias, but sets the address in front of the interface address list to select it as the source for unbound sockets.
group *ID*	Adds a group ID to the group ID list for the interface. This list is used in determining the route to use when forwarding packets that arrived on the interface.
-group *ID*	Removes a group ID from the group ID list for the interface. This list is used in determining the route to use when forwarding packets that arrived on the interface.
hwloop	Enables hardware loopback. The hardware loopback specifies that locally addressed packets that are handled by an interface are sent out by using the associated adapter.
-hwloop	Disables hardware loop-back. The hardware loop-back specifies that locally addressed packets that are handled by an interface must be sent out using the associated adapter.

Item	Description
	ipdst Specifies an Internet host that can receive IP packets encapsulating **ns** packets that are bound for a remote network. An apparent point-to-point link is constructed, and the specified address is taken as the **ns** address and network of the destination.
	ipv6dst Used to specify an IPv6 node that can receive IPv6 packets encapsulating IPv6 or IPv4 packets through a tunnel. The apparent destination of the point-to-point tunnel interface might not be the real destination of the packets. At the tunnel endpoint, the decapsulated packets might then be forwarded to their final destination.
	largesend Enables one LPAR to send large data in a single packet to another LPAR. It works similarly to `largesend` over real adapters except in this case no TCP segmentation is done. If the SEA on VIOS supports `largesend`, the LPAR can transmit large data, which gets segmented by the real adapter on SEA. Use the `chdev` command to enable the `largesend` attribute on SEA.
	-largesend Disables `largesend` over virtual Ethernet. This value is the default.
	link [0-2] Enables special processing of the link level of the interface. These 3 options are interface-specific. In actual effect, however, they are used to select special modes of operation. An example of the usage is to enable Serial Line Internet Protocol (SLIP) compression, or to select the connector type for some Ethernet cards. For more information, see the manual page of the specific driver.
	-link [0-2] Disables special processing at the link level with the specified interface.
	metric *Number* Sets the routing metric of the interface to the value specified by the *Number* variable. The default is 0. The routing metric is used by the routing protocol (the **routed** daemon). Higher metrics have the effect of making a route less favorable. Metrics are counted as additional hops to the destination network or host.
	monitor Enables the underlying adapter to notify the interface layer of link status changes. The adapter must support link status callback notification. If multipath routing is used, alternative routes are selected when a link goes down.
	-monitor Disables monitoring of the adapter link status.

Item	Description
mtu ***Value***	Sets the maximum IP packet size for this system. The *Value* variable can be a number in the range 60 - 65535, but is media-dependent. See Automatic configuration of network interfaces in *Networks and communication management* for the maximum transmission unit (MTU) values by interface.
netmask ***Mask***	Specifies how much of the address must be reserved for subdividing networks into subnetworks. This parameter can be used only with the **inet** address family. The *Mask* variable includes both the network part and the subnet part of the local address, which is taken from the host field of the address. The mask can be specified as a single hexadecimal number that begins with 0x, in standard Internet dotted decimal notation, or begins with a name or alias that is listed in the **/etc/networks** file. In the 32-bit address, the *Mask* variable contains 1s for the bit positions that are reserved for the network and subnet parts, and *Mask* variable contains 0s for the bit positions that specify the host. The *Mask* variable contains at least the standard network portion. The subnet segment is contiguous with the network segment. If the **netmask** parameter is used without specifying the IP address, the netmask value of the first IP address of the specified interface is updated.
pvc	(ATM network interface only). Specifies that this interface supports Permanent Virtual Circuit (PVC) types of virtual connections only.
pktchain	Enables the flag to indicate that this interface can handle multiple packets chained together on the output path.
-pktchain	Disables the flag that indicates that this interface can handle multiple packets chained together on the output path.
svc_c ***server_addr***	(ATM Network interface only). Specifies that this interface supports both switched virtual circuit (SVC) and PVC types of virtual connections. It further specifies that this interface is an ARP client. The *server_addr* is the list of 20-byte ATM addresses of the ARP servers that this client uses. The addresses are specified in the form of xx.xx....xx. The first entry is considered the primary ARP server and the rest are considered secondary ARP servers. The list of 20-byte ARP server addresses must be separated by a comma.
site6	Sets the IPv6 site number (default is 0). This command must be used only with site-local addresses on a multiple-site node.

Item	Description
	svc_s (ATM network interface only). Specifies that this interface supports both SVC and PVC types of virtual connections. It further specifies that this interface is the ARP server for this logical IP subnetwork (LIS). **security** Reserved for future use. **snap** Reserved for future use. **-snap** Reserved for future use. **tcp_low_rto** Enables the use of lower retransmission timeouts (RTO) for TCP connections on a low latency, fast network, such as gigabit ethernet and 10-gigabit ethernet). If the networks experience packet drops, the respective TCP connections use the *rto* value for RTO. The *rto* values are in the range of 0 - 3000 ms. This runtime option must be set in the **if_isno** flags field. The **use_isno** option must also be set for this flag to be effective. **tcp_nocksum** Disables verification of the checksum of TCP data for local traffic to the subnet attached to the interface. Checksum verification of TCP, UDP, and IP headers continues. Checksum verification of TCP data that is read or written from this interface, from or to remote networks also continues. **-tcp_nocksum** Enables verification of the checksum of TCP data for local traffic to the subnet attached to the interface. This value is the default. **thread** (**inet** only) Configures dedicated kernel threads for an interface. This parameter can be used only SMP systems that have multiple CPU. This parameter causes input packets to be queued to a kernel thread after processing by the device driver and input demuxer. The input packet is processed in IP and TCP, or UDP by the thread instead of directly on the interrupt level. Setting this parameter can improve throughput when high-speed adapters bottleneck on a single CPU during interrupt processing by allowing the input packets to be processed on other CPUs running the kernel threads (improved pipelining). For some workloads, this parameter increases the per packet load, due to the thread scheduling load, resulting in higher CPU utilization, and possibly lower throughput. **-thread** (**inet** only) Disables kernel thread support that has been configured with the *thread* parameter.

Item	Description
	tunnel Configures a dedicated tunnel for the trusted communication. A tunnel establishes a virtual link between two trusted nodes for transmitting data packets as payloads of other packet headers. A tunnel can be one of the following types: **Generic routing encapsulation (GRE) tunnel** Expects the source and destination IPv4 addresses of the tunnel endpoint as arguments that are followed by the **tunnel** parameter value. A tunnel is created between the 2 endpoints. **IPv4 over IPv6 tunnel (GIF tunnel)** Expects the source IPv6 address of the tunnel. The address is followed by a destination IPv4 address and a destination IPv6 address that are separated by a comma. For one-to-many tunnels, each target is separated by a comma. **transfer *tointerface*** Transfers an address and its related static routes from *interface* to *tointerface*. For IPv6, this command works only for addresses added by using the **ifconfig** command. `ifconfig interface addressfamily address transfer tointerface` **Note:** If you want to transfer an IP address from one interface to another, and if the destination interface is not part of the virtual LAN (VLAN) to which the IP address belongs, you must add the VLAN to the adapter on which the destination interface is configured. **up** Marks an interface as active (**up**). This parameter is used automatically when you set the first address for an interface. It can also be used to enable an interface after you issue an **ifconfig down** command. **vipa_iflist** Adds the interfaces to the list of interfaces that must use this **vipa** parameter as the source address in the outgoing packets. **-vipa_iflist** Removes the interfaces from the list of interfaces that are configured to use this **vipa** as the source address in the outgoing packets. **scope addrscope zone zoneid** Moves the interface into the topological zone that is specified by *zoneid* at the address scope that is specified by the *addrscope*. IPv6 zones are defined in RFC 4007. The parameter is applicable to only inet6.
rto	Specifies the retransmission timeout in milliseconds. The range for this value is 0 - 3000. **Requirement:** You must set the **timer_wheel_tick** value of the **no** command before you set the *rto* value by using the **ifconfig** command. The *rto* value that you specify must be equal to or a multiple of 10 times the **timer_wheel_tick** value that is set.

The following network options, commonly known as Interface Specific Network Options (ISNO), can be configured on a per interface basis:

rfc1323 [0 | 1]
Enables or disables TCP enhancements as specified by RFC 1323, *TCP Extensions for High Performance*. A value of 1 specifies that all TCP connections by using this interface will attempt to negotiate the RFC enhancements. A value of 0 disables **rfc1323** for all connections by using this interface. The SOCKETS application can override this ISNO and global behavior on individual TCP connections with the **setsockopt** subroutine.

-rfc1323
Removes the use of ISNO for **rfc1323** for this network. A SOCKETS application can override the global behavior on individual TCP connections by using the **setsockopt** subroutine.

tcp_mssdflt *Number*
Sets the default maximum segment size that is used in communicating with remote networks. If you communicate over this interface, a socket uses *Number* as the value of the default maximum segment size.

-tcp_mssdflt
Removes the use of ISNO for the **tcp_mssdflt** option. The global value, which is manipulated through **/usr/sbin/no**, is used instead.

tcp_recvspace *Size*
Specifies the default socket buffer size for interface sockets that are receiving data. The buffer size affects the window size that is used by TCP. (For more information, see the **no** command.)

-tcp_recvspace
Removes the use of ISNO for the **tcp_recvspace** option. The global value is used instead.

tcp_sendspace *Size*
Specifies the default socket buffer size for interface sockets that are sending data. The buffer size affects the window size that is used by TCP. (For more information, see the **no** command.)

-tcp_sendspace
Removes the use of ISNO for the **tcp_sendspace** option. The global value is used instead.

tcp_nodelay [0 | 1]
Specifies that sockets by using TCP over this interface follow the Nagle algorithm when you send data. By default, TCP follows the Nagle algorithm.

-tcp_nodelay
Removes the use of ISNO for the **tcp_nodelay** option.

Tip: Parameters that you set by using the **ifconfig** command are lost the next time that you restart your system. Use the **chdev** command to change the Object Data Manager (ODM) database for each interface to make parameter changes permanent. Use the **lsattr -E -l** [*interface*] command to view the interface attributes and use the **chdev -l** [*interface*] **-a** [*attribute=value*] command to change the attribute. For example:

```
lsattr -E -l en0
chdev -l en0 -a tcp_sendspace=65536
```

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To query the status of a serial line IP interface, enter the following command:

```
ifconfig  sl1
```

In this example, the interface to be queried is `sl1`. The result of the command looks similar to the following result:

```
sl1:  flags=51<UP,POINTOPOINT,RUNNING>
            inet  192.9.201.3  -->  192.9.354.7  netmask  ffffff00
```

2. To configure the local loop-back interface, enter the following command:

```
ifconfig  lo0  inet  127.0.0.1  up
```

3. To mark the local token-ring interface as down, enter the following command:

```
ifconfig  tr0  inet  down
```

In this example, the interface to be marked is `token0`.

> **Note:** Only a user with root user authority can modify the configuration of a network interface.

4. To turn **rfc1323** off for all connections over `en5` (assuming that the global value is 1), enter the following command:

```
ifconfig en0 rfc1323 0
```

5. To configure a list of interfaces to use a vipa, enter the following command:

```
ifconfig vi0 vipa_iflist en0,en1,tr0
```

6. To remove interfaces that are configured to use vipa, enter the following command:

```
ifconfig vi0 -vipa_iflist en1,tr0
```

7. To find out which interfaces are configured to use a vipa, say vi0, enter the following command:

```
ifconfig vi0
```

8. To enable link status monitoring, enter the following command:

```
ifconfig en0 monitor
```

If the link status on adapter `ent0` changes to down, the adapter notifies the interface layer, which causes the interface to also be marked as down.

9. To configure a Generic Routing Encapsulation (GRE) tunnel between the interfaces of two nodes, enter the following command:

```
ifconfig gre0 tunnel 9.3.149.70 9.3.149.121
```

This creates a GRE tunnel between the local interface `9.3.149.70` and the remote interface `9.3.149.121`. The local end of the tunnel is identified by `gre0`.

10. To associate an IP address with the newly created interface, enter the following command:

```
ifconfig gre0 10.10.10.1
```

11. To configure NAT on a GRE tunnel, enter the following command:

```
ifconfig gre0 nat toaddr 127.0.0.1 fromport 80 toport 8080
```

In this example, the original destination port of the GRE packet is 80 and the command changes the destination port to 8080 and the destination address to `127.0.0.1`.

12. To configure a GIF tunnel between one-to-many endpoints (one-to-many tunnel), enter the following command:

```
ifconfig gif0 10.10.10.1 netmask 255.255.255.0 tunnel  2000::4612:6995:6c4a:fa6e
10.10.10.10,2000::4612:6995:6c4a:fa6a 10.10.10.11,2000::4612:6995:6c4a:
fa6b 15.15.15.1,2000::4612:6995:6c4a:f777
```

The command creates a one-to-many tunnel between the source (2000::4612:6995:6c4a:fa6e) and the following 3 targets:

- 2000::4612:6995:6c4a:fa6a
- 2000::4612:6995:6c4a:fa6b
- 2000::4612:6995:6c4a:f777

The command also configures the 10.10.10.1 IP address to the GIF interface.

13. To configure a one-to-one GIF tunnel between two endpoints, enter the following command:

```
ifconfig gif0 10.10.10.1 netmask 255.255.255.0 tunnel  2000::4612:6995:6c4a:fa6e
10.10.10.10,2000::4612:6995:6c4a:fa66
```

The command creates a GIF tunnel between the source (2000::4612:6995:6c4a:fa6e) and the target (2000::4612:6995:6c4a:fa66).

14. The command also configures the 10.10.10.1 IP address to a GIF interface.

Files

Item	Description
/etc/host	Contains the host name database.
/etc/networks	Contains network names.

ike Command

Purpose

Starts, stops, and monitors IP Security dynamic tunnels which use the Internet Key Exchange Protocol (ISAKMP/Oakley).

Syntax

ike cmd=Subcommand [*parameter* ...]

Description

The **ike** is used to start, stop, and monitor IP Security dynamic tunnels using the Internet Key Exchange (IKE) protocol. IP Security tunnels protect IP traffic by authenticating and/or encrypting IP data. The **ike** command performs several functions. It can activate, remove, or list IKE and IP Security tunnels.

Note: You must have root access to use the **ike** command.

The IKE negotiation occurs in two phases. The first phase authenticates the two parties and sets up a **Key Management** (also known as phase 1) **Security Association** for protecting the data that is passed during the negotiation. In this phase the key management policy is used to secure the negotiation messages. The second phase negotiates **Data Management** (also known as the phase 2) **Security Association**, which uses the data management policy to set up IP Security tunnels in the kernel for encapsulating and decapsulating data packets. The secure channel established in phase 1 can be used to protect multiple data management negotiations between 2 hosts.

The **ike** command is used to activate tunnels with identification and policy information which has already been entered using the **ikedb** command. The parameters to be used during the negotiation are entered by the user and stored in a database. The **ike** command allows the activation, removal and listing of tunnels that have been started using the security parameters stored in the database.

In most uses of the **ike** command, activation and deletion occurs for both phases, however the command allows these operations to be done separately.

Subcommands

activate

activate command

Item	Description
Purpose	Start the negotiation of an IKE tunnel. If phase is not specified, both a phase 1 and phase 2 tunnel are started. If IP addresses are supplied, the tunnel is setup using those IP addresses. If the IDs used during the negotiation are not IP addresses, the local and remote host IDs must be entered using the **ikedb** command. A unique tunnel number is created. The tunnel can then be referenced by the tunnel number in the **ike** command to indicate the particular tunnel to be started.
Syntax	**ike cmd=activate** [**phase=1**|**2**] [**numlist=***tunnel_num_list*] [**namelist=tunnel_name_list**] [**remid=***remote_id*] [**ipaddr=***src_addr*,*dst_addr*] [**autostart**]
Description	The **activate** subcommand works using a two phase paradigm. A phase 1 tunnel must be established before a phase 2 tunnel can be started. If a phase 1 tunnel is specified, then only the phase 1 tunnel negotiation takes place. If a phase 2 tunnel is specified, the system checks for the existence of the corresponding phase 1 tunnel before creating the phase 2 tunnel. If the phase 1 negotiation has not been started, it is started automatically. Upon successful completion of a phase 2 tunnel, the tunnel definition and corresponding filter rules are inserted into the IP Security kernel, and the new tunnel is activated. Traffic described by the tunnel definition passing between the designated endpoints is protected by the encryption and authentication algorithms indicated by the associated IKE security policy. Multiple phase 2 tunnels can be started under the same phase 1 tunnel. A situation where this may be desired is if different types of traffic between two endpoints need different levels of security protection. The Security Association used for the phase 1 tunnel can be shared by multiple phase 2 tunnels. The phase 2 tunnels would specify the type of traffic (by protocol and port, or subnet mask, for instance) and could have different security policies protecting them. The **ike** command returns if either a negotiation has been initiated, an error returns, or the tunnel already exists. Since the remote host must be contacted during the negotiation and the amount of time needed to complete the negotiation is uncertain, the **list** subcommand should be used to determine if the negotiation was successful. Errors that are detected during the negotiation process can be captured by using **syslog**.

Item	Description
Flags	**phase** Specifies the type of negotiation desired. If omitted, the **activate** subcommand activates both a phase 1 and phase 2 tunnel. The **phase** flag is an optional flag. **numlist** Initiates the **ike** tunnel number which corresponds to the desired phase 1 or phase 2 tunnel(s) to be started. The **,** (comma) and **-** (dash) characters can be used to delimit values and indicate ranges. The **list** subcommand with the database option **db** can be used to determine the tunnel number for a particular tunnel. An example using tunnel numbers is shown below: `ike cmd=activate numlist=1,3,5-7` This would start tunnels 1, 3, 5, 6 and 7. **remid** Starts phase 1 or phase 2 tunnel(s) from the local ID to the specified remote ID. **remid** could be a phase 1 ID (such as IP address, FQDN, user FQDN and X500DN), a phase 2 ID (such as IP address, subnet and IP address range) or a group ID. The **,** (comma) is used to delimit the subnet id and subnet mask, and the starting and ending IP address. If **remid** is a group name, a tunnel is initiated for each group member. **remid** is an optional flag and can only be used with the **activate** subcommand. It cannot be used in conjunction with the **ipaddr**, **numlist** or **namelist** flags. 1. To activate a phase 1 tunnel to remote IP address 9.3.97.100, type: `ike cmd=activate phase=1 remid=9.3.97.100` 2. To activate a phase 2 tunnel to remote subnet ID 9.3.97.100,255.255.255.0, type: `ike cmd=activate phase=2 remid=9.3.97.100,255.255.255.0` **ipaddr** Starts a phase 1 or phase 2 tunnel between the specified IP Addresses. **autostart** Causes the activation of all phase 1 and phase 2 tunnel database entries which were created with the **autostart** parameter set. The **autostart** flag does not work in conjunction with any other flags pertaining to the **activate** subcommand. **namelist** Specifies a tunnel name or comma-separated list of tunnel names to be activated. This flag requires the use of the **phase** flag.

activate command *(continued)*

Item	Description
Examples	1. To activate a phase 2 tunnel between source IP address x.x.x.x and destination IP address y.y.y.y, enter: `ike cmd=activate phase=2 ipaddr=x.x.x.x,y.y.y.y` The security policy indicated in the database for the IP addresses x.x.x.x and y.y.y.y is used for activating the tunnel. 2. To activate phase 1 tunnels for tunnels 1 and 2, enter: `ike cmd=activate phase=1 numlist=1,2` 3. To activate phase 2 tunnels for inactive tunnels named AIXFW1_DM and remote_office in the database enter: `ike cmd=activate phase=2 namelist=AIXFW1_DM,remote_office` **Note:** Because each phase 2 tunnel must have an associated phase 1 tunnel, a phase 1 tunnel is automatically activated before the phase 2 tunnel is activated.

list

list command

Item	Description
Purpose	Monitors the status of IP Security tunnels by phase. It is also used to view tunnel entries defined in the IKE database.
Syntax	**ike cmd=list** [**phase=1**|**1+**|**2**] [**numlist=** *tunnel_num_list*] [**db** | **role=i**|**r**] [**verbose**]
Description	The **list** subcommand queries the Tunnel Manager and lists phase 1 and phase 2 tunnel status and information according to the result of the query. This command can also be used to view information in the Tunnel Definition database. The default behavior is to list the tunnels currently active. To list the tunnels in the database, the **db** option must be used.

Item	Description
Flags	**phase** Indicates the type and order of the tunnel(s) to be listed. A phase value of 1 results in only the requested phase 1 tunnel information being displayed. A phase value of 2 results in the information for the requested phase 2 tunnel(s) and their associated phase 1 tunnel(s) should be displayed. A phase value of 1+ means that the requested phase 1 tunnel and all associated phase 2 tunnels should be displayed. The default phase value is 1+. **numlist** Lists of the tunnel numbers which you would like to view. If omitted, the information from all tunnels is displayed. The **,** (comma) and **-** (dash) characters can be used to delimit values and indicate ranges. For example: `ike cmd=list numlist=1,3,5-7` When used in conjunction with **db**, tunnels from the IKE Security Policy database are shown. **Note:** Active tunnel numbers and tunnel numbers from the IKE Tunnel Definitions database do not necessarily match up. This is because a single tunnel entry in the database can correspond to multiple active tunnels. **db** Shows the entries in the database. If this flag is omitted, only active tunnels are displayed. This cannot be used in conjunction with **role**. Supply the list of tunnel numbers which you would like to view. **role** Allows the display of tunnels by the point of initiation. If **i** is specified, then the tunnels that were initiated by the local host are displayed. If **r** is specified, then the tunnels where the local host acted as a responder are displayed. If this flag is omitted, both initiator and responder tunnels are shown. This flag cannot be used in conjunction with **db**. **verbose** Shows extended information about the specified tunnels. If this flag is not specified, then only a concise entry for each tunnel is shown.

list command *(continued)*

Item	Description
Examples	**Note:** Tunnel numbers from the database and tunnel numbers from the tunnel manager do not necessarily reflect the same tunnel. 1. To perform a concise (short form) listing of phase 1 tunnels with entries in the tunnel manger, enter: `ike cmd=list phase=1 numlist=1,2,3` These tunnels are either being negotiated, in the active state , or have expired. Only tunnels 1, 2, and 3 are listed. Tunnels can be either initiator or responder role. 2. To perform a concise (short form) listing of of the specified phase 2 tunnels in the database with each preceded by the associated phase 1 tunnel, enter: `ike cmd=list phase=2 numlist=1-3 db` These are tunnels defined in the database which may or may not be currently active in the tunnel manager. All tunnels in the database are used in the initiator role only. 3. To perform a verbose (long form) listing of a phase 1 tunnel followed by all of its associated phase 2 tunnels from the tunnel manager, enter: `ike cmd=list phase=1+ role=r verbose` Only tunnels which were activated in the responder role are listed. All available tunnel numbers are listed since no **numlist** was specified.

remove

remove command

Item	Description
Purpose	Deactivates specified phase 1 or phase 2 tunnel(s).
Syntax	**ike cmd=remove** [**phase=1**|**2**] [**numlist=** *tunnel_num_list*] [**all**]
Description	The **remove** subcommand requests the deactivation of phase 1 or phase 2 tunnel(s). Because phase 2 tunnels are associated with a phase 1 tunnel, if a phase 1 tunnel is deactivated, all phase 2 tunnels under the phase 1 tunnel are not refreshed when the phase 2 tunnel lifetime expires.
Flags	**phase** Indicates the phase of the tunnel to be deactivated and must be specified. A phase value of 1 refers to a phase 1 tunnel and a phase value of 2 refers to a phase 2 tunnel. **numlist** Lists the tunnel numbers you would like to deactivate. The **,** (comma) and **-** (dash) characters can be used to delimit values and indicate ranges. For example: `ike cmd=remove phase=1 numlist=1,3,5-7` When **numlist** is omitted, all tunnels are deactivated. **all** Deactivates all active tunnels. This parameter does not work in conjunction with **numlist**.

remove command *(continued)*

Item	Description
Examples	1. To deactivate phase 1 tunnels numbered 1, 2, and 3, enter: `ike cmd=remove phase=1 numlist=1-3` 2. To deactivate all phase 1 and phase 2 tunnels, enter: `ike cmd=remove all` 3. To deactivate all phase 2 tunnels but keep all phase 1 tunnels active, enter: `ike cmd=remove phase=2 all` 4. To deactivate all phase 1 tunnels (corresponding phase 2 tunnels will not be refreshed), enter: `ike cmd=remove phase=1 all`

log

Table 10. log

Item	Description
Purpose	Read the ISAKMP daemon log level from **/etc/isakmpd.conf** and start logging at that level.
Syntax	**ike cmd=log**
Description	The **log** subcommand causes the ISAKMP daemon to read the log level from **/etc/isakmpd.conf**, and a filename from **/etc/syslog.conf**. The logging level specified is set and the log output, along with other **syslog** output, is placed in the file specified.

Notes:

- If the log level or the output file name in the `/etc/syslog.conf` file is changed, you can refresh the `syslogd` subsystem by running the **`refresh -s syslogd`** command or by refreshing the IKE daemons. You can refresh the IKE daemons by running the **`refresh -s ike`** command.
- There are four valid logging levels for the ISAKMP daemon. They are **none**, **errors**, **events**, and **information**. **none** means no logging, **errors** means logging of only ISAKMP daemon errors will occur, **events** means errors and other ISAKMP daemon events will be logged, and **information** is the highest level of logging which is all inclusive.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Files

Files

Item	Description
/usr/sbin/ike	Location of the **ike** admin commands.
/etc/isakmpd.conf	Configuration file for the **iksakmpd** daemon.
/etc/syslog.conf	Provides configuration information for the **syslogd** daemon.

ikedb Command

Purpose

Retrieves, updates, deletes, imports, and exports information in the IKE database.

Syntax

ikedb **-p**[**F** **s**] [**-e** *entity-file*] [*XML-file*]

ikedb **-g**[**r**] [**-t** *type* [**-n** *name* | **-i** *ID* **-y** *ID-type*]]

ikedb **-d** **-t** *type* [**-n** *name* | **-i** *ID* **-y** *ID-type*]

ikedb **-c**[**F**] [**-l** *linux-file*] [**-k** *secrets-file*] [**-f** *XML-file*]

ikedb **-x**

ikedb **-o**

LDAP supported operations

ikedb –R LDAP **-p** [**-F**]

ikedb –R LDAP **-g** [*policy-name*]

ikedb –R LDAP **-o**

ikedb –R LDAP **-A** <*policy-name*> [-f <xml file name>] [-h ip/host] -C <Dn Name>

ikedb –R LDAP -D <*policy-name*> [-h ip/host] [-F]

Description

The **ikedb** command allows the user to write to (**put**) or read from (**get**) the IKE database. The input and output format is an Extensible Markup Language (XML) file. The format of an XML file is specified by its Document Type Definition (DTD). The **ikedb** command allows the user to see the DTD that is used to validate the XML file when doing a put. While entity declarations can be added to the DTD using the **-e** flag, this is the only modification to the DTD that can be made.

Any external DOCTYPE declaration in the input XML file will be ignored and any internal DOCTYPE declaration might result in an error. The rules followed to parse the XML file using the DTD are specified in the XML standard. **/usr/samples/ipsec** has a sample of what a typical XML file that defines common tunnel scenarios looks like.

Flags

To use LDAP supported operations, configure the host as an LDAP client.

Item	Description
-p	Performs a put, which writes to the database, based on the given *XML-file*. **-F** Forces a put, even if a specified tunnel, protection, proposal, group, or preshared key would overwrite one that exists in the database. The default is for such put attempts to fail. When an **-R** switch is present, the local entities are overwritten in case the name is a duplicate of a name specified as part of applicable policy on the host in the configuration. **-s** Swaps the local and remote IDs of all tunnels. This flag facilitates importing a tunnel that is generated by a peer system. This flag affects only tunnels. This option is illegal if the remote ID of any tunnel is a group. **-e *entity-file*** Specifies the name of the file that contains the <!ENTITY ...> lines as defined by *entity-file*. These lines are added to the internal DTD and allow the user to include XML files in other XML files. ***XML-file*** Specifies the **XML-file** to be used and must be the last argument to be displayed in the command line. The *XML-file* determines whether the write is to a tunnel, protection, proposal, group, pre-shared key, or all of these. If no *XML-file* is specified, input is read from **stdin**. A - (hyphen) can also be used to specify **stdin**. **-R LDAP** The valid value is **LDAP**. When **-p** is used in conjunction with the **-R** switch, the put operation is done by importing the XML configuration file that is associated with the applicable IPSec configuration policy from the LDAP server.
-h	Specifies host name or IP address along with the **-A** flag or the **-D** flag. The IP address can be IPv4 or IPv6.

Item	Description
-g	Performs a **get**, which displays what is stored in the IKE database. Output is sent to **stdout** and is in XML format, which is suitable for processing with **ikedb -p**. **-r** Recursive. If this flag is specified for a phase 1 tunnel, information is also returned for all associated phase 2 tunnels and all protections and proposals associated with both sets of tunnels. **-t *type*** Specifies the *type* of output requested. *Type* can have the value of any of the XML elements under AIX_VPN, such as **IKETunnel**, **IPSecProtection**, and so on. If omitted, the entire database is output. **-n *name*** Specifies the *name* of the requested object. *Name* can be the name of a proposal, protection, tunnel, or group, depending on the value of the **-t** flag. The **-n** flag is valid with all values specified by the **-t** flag, except **IKEPresharedKey**. If omitted, all objects of the specified *type* will be output. **-i *ID*** Specifies the *ID* associated with a pre-shared key. The **-i** flag is only valid with the **IKEPresharedKey** value of the **-t** flag. If omitted, all objects of the specified *type* will be output. The **-i** flag must be used in conjunction with the **-y** flag. **-y *ID-type*** Specifies the *ID-type* defined by the **-i** flag. ID-type can be any of the legal types allowed in the XML file, such as **User_FQDN**, **IPV4_Address**, and so on. The **-y** flag must be used in conjunction with the **-i** flag. **-R LDAP** The valid value is **LDAP**. When the **-g** flag is used in conjunction with the**-R** switch, the get operation is done by displaying XML configuration file stored on the LDAP server for the policy that is associated with the local host. If a policy name is also provided, the xml file stored as part of the policy is displayed on stdout.
-d	Performs a **delete** on the specified item from the database. The flags are the same as for the **-g** flag, except that **-r** is not supported.
-C	Used to provide the IPSec certificate used in the associated clients.
-c	Performs a conversion from a Linux IPSec configuration file to an AIX IPSec configuration file in XML format. It requires as input one or two files from the Linux environment, a configuration file, and possibly a secrets file with pre-shared keys. **-F** Forces a **put**, even if a specified tunnel, protection, proposal, group, or pre-shared key would overwrite one that already exists in the database. The default is for such **put** attempts to fail. The **-F** flag has no effect if the **-f** flag is also used. **-l *linux-file*** Specifies the Linux configuration file as define by *linux-file*. If no file is specified, the system looks for the **ipsec.conf** file in the current directory. **-k *secrets-file*** Specifies the Linux pre-shared keys file as defined by the *secrets-file* parameter. If no file is specified, the system looks for the **ipsec.secrets** file in the current directory. **-f *XML-file*** Specifies the XML configuration file to which the Linux configuration files are converted. The default behavior is to do a put operation directly to the IKE database. If the filename has a hyphen (-), the results are sent to stdout. This flag is invalid if the -R switch is also present on the command line.

Item	Description
-x	Performs an **expunge** operation on the database. This flag empties the database. This flag is invalid if the **-R** flag is also present on the command line.
-o	Performs an **output** of the DTD that specifies all elements and attributes for an XML file that is used by the **ikedb** command. The DTD is sent to **stdout**. When **-R** switch is present, DTD that specifies all the elements and attribute for the XML file allowed to be stored as part of configuration policy on LDAP is sent to **stdout**.
-A	Associates the IP addresses provided with the policy name. If no IP addresses are provided, the first local IPV6 address for the local host is selected and associated with the policy. Policy configuration is enforced by downloading the XML file from LDAP and putting it into the database. The tunnels thus defined are activated. **-f < -path to XML file >** If an XML file is provided, it is stored on the LDAP server as the new XML applicable for the defined policy. If the policy does not exist, this flag is required. **-R LDAP** The valid value is LDAP. This switch must be provided on the command line.
-D	Performs disassociation of configuration policy and IP on LDAP server. This flag is invalid without the **-R** switch. The only valid value for the **R** switch is LDAP. **- F** If the last IP address associated with the specified policy is removed, this switch causes the corresponding policy data (XML configuration file) to be deleted from LDAP server. If this flag is not used, the policy is not deleted from the LDAP server.

Files

Item	Description
/usr/samples/ipsec	Examples of an XML file that sets up various tunnel configurations.

Examples

1. To **put** definitions to the IKE database from an XML file that has been generated on a peer machine and overwrite any existing objects in the database with the same name, type:

   ```
   ikedb -pFs peer_tunnel_conf.xml
   ```

   `peer_tunnel_conf.xml` is the XML file generated on a peer machine.
2. To **get** the definition of the phase 1 tunnel named `tunnel_sys1_and_sys2` and all dependent phase 2 tunnels with respective proposals and protections, type:

   ```
   ikedb -gr -t IKETunnel -n tunnel_sys1_and_sys2
   ```
3. To **delete** all preshared keys from the database, type:

   ```
   ikedb -d -t IKEPresharedKey
   ```
4. To **associate** the host that has the IP address 10.10.10.1 with the configuration policy named Poll with certificate */C=US/O=IBM/CN=test01.austin.ibm.com* with xml file ldap.xml, type:

   ```
   ikedb -R LDAP -A Pol1  -f  ldap.xml  -h 10.10.10.1 -C /C=US/O=IBM/CN=test01.austin.ibm.com
   ```

imake Command

Purpose

C preprocessor interface to the **make** command.

Syntax

imake [**-D***Define*] [**-I***Directory*] [**-T***Template*] [**-f** *FileName*] [**-C** *FileName*] [**-s** *FileName*] [**-e**] [**-v**]

Description

The **imake** command generates **Makefiles** from a template, a set of cpp macro functions, and a per-directory input file called **Imakefile**. This command keeps machine dependencies (such as compiler options, alternate command names, and special **make** command rules) separate from the descriptions of the items to build.

imake invokes cpp with any **-I** or **-D** flags passed on the command line and passes to it the following three lines:

```
#define IMAKE_TEMPLATE "Imake.tmpl"
#define INCLUDE_MAKEFILE "Imakefile"
#include IMAKE_TEMPLATE
```

Override **Imake.tmpl** and **Imakefile** by using the **-T** and **-f** flags, respectively.

The IMAKE_TEMPLATE typically reads the following files:

- A machine-dependent parameters file in which the parameters are specified as cpp symbols
- A site-specific parameters file
- A file that defines variables
- A file containing cpp macro functions for generating **make** command rules
- The **Imakefile** (specified by INCLUDE_IMAKEFILE) in the current directory.

The **Imakefile** file uses the macro functions to indicate what targets to build and the **imake** command generates the appropriate rules.

Imake configuration files contain two types of variables, imake variables and make variables. The imake variables are interpreted by cpp when the **imake** command is run. By convention, they are not case-sensitive. The make variables are written into the **Makefile** for later interpretation by the **make** command. By convention, make variables are uppercase.

The rules file (usually named **Imake.rules** in the configuration directory) contains a variety of cpp macro functions that are configured according to the current platform. The **imake** command replaces any occurrences of the string ``@@'' with a newline character (carriage return) to support macros that generate more than one line of make rules. For example, the macro:

```
#define   program_target(program, objlist)        @@\
program:  objlist                                 @@\
          $(CC)  -o  $@  objlist  $(LDFLAGS)
```

when called with `program_target(foo,foo1.o foo2.o)` will expand to:

```
foo:      foo1.o  foo2.o
          $(CC)  -o  $@  foo1.o  foo2.o  $(LDFLAGS)
```

On systems whose cpp reduces multiple tabs and spaces to a single space, the **imake** command attempts to put back any necessary tabs (the **make** command distinguishes between tabs and spaces). For this reason, precede all colons (:) in command lines by a backslash (\).

Use with

AIXwindows uses the **imake** command extensively for both full builds within the source tree and builds of external software. Two special variables, TOPDIR and CURDIR, are set to make referencing files using relative path names easier. For example, the following command is generated automatically to build the **Makefile** in the **lib/X** directory (relative to the top of the sources):

```
%  ../.././config/imake  -I../.././config  \
            -DTOPDIR=../../.  -DCURDIR=./lib/X
```

To build AIXwindows programs outside the source tree, a special symbol, UseInstalled, is defined and the TOPDIR and CURDIR variables are omitted. If the configuration files are properly installed, you can use the **xmkmf** command.

The **imake** command reads the following files as used by AIXwindows.

Note: The indented format indicates files that include other files.

```
Imake.tmpl                  generic variables
   site.def                 site-specific, BeforeVendorCF defined
         *.cf               machine-specific
             *Lib.rules     shared library
   site.def                 site-specific, AfterVendorCF defined
   Imake.rules              rules
   Project.tmpl             X-specific variables
         *Lib.tmpl          shared library variables
Imakefile
   Library.tmpl             library rules
   Server.tmpl              server rules
   Threads.tmpl              multi-thread rules
```

Note: The **site.def** file is included twice, both before and after the ***.cf** file. Although most site customizations are specified after the ***.cf** file, some, such as the choice of compiler, need to be specified before, because other variable settings may depend on them.

The first time the **site.def** file is included, the **BeforeVendorCF** variable is defined, and the second time, the **AfterVendorCF** variable is defined. All code in the **site.def** file should be placed inside a **#ifdef** macro for one of these symbols.

Flags

Item	Description
-D*Define*	Passed directly to cpp to set directory-specific variables. For example, X-windows uses this flag to set the TOPDIR variable to the name of the directory containing the top of the core distribution, and the CURDIR variable to the name of the current directory, relative to the top.
-e	Indicates that the **imake** command should execute the generated **Makefile**. The default is to leave this to the user.
-f *FileName*	Specifies the name of the per-directory input file. The default is the **Imakefile** file.
-I*Directory*	(Uppercase i) Passed directly to cpp to indicate the directory in which the imake template and configuration files are located.
-C *FileName*	Specifies the name of the **.c** file that is constructed in the current directory. The default is **Imakefile.c**.
-s *FileName*	Specifies the name of the **make** description file to be generated, without invoking the **make** command. If the *FileName* variable is a **-** (dash), the output is written to **stdout**. The default is to generate, but not execute, a **Makefile**.
-T*Template*	Specifies the name of the master template file (which is usually located in the directory specified with **-I**) used by the **cpp** command. The default is the **Imake.tmpl**.
-v	Indicates that **imake** should print the **cpp** command line that it is using to generate the **Makefile**.

Environment Variables

Note: The following environment variables may be set, but their use is not recommended because they introduce dependencies that are not readily apparent when the **imake** command is run.

Item	Description
IMAKEINCLUDE	If defined, specifies an include argument for the C preprocessor. For example: `-I/usr/include/local`
IMAKECPP	If defined, specifies a valid path to a preprocessor program. For example: `/usr/local/cpp` The default is the **/lib/cpp** program.
IMAKEMAKE	Specifies a valid path to a **make** program such as **/usr/local/make**. By default, **imake** uses whatever **make** program is found using the **execvp** subroutine. This variable is only used if the **-e** flag is specified.

Example

```
imake -I/usr/lib/X11/config -DTOPDIR=/usr/lpp/X11/Xamples
```

Files

Item	Description
/usr/tmp/tmp-imake.*nnnnnn*	Specifies the temporary input file for the cpp preprocessor.
/usr/tmp/tmp-make.*nnnnnn*	Specifies the temporary input file for make.
/lib/cpp	The default C preprocessor.

imapd Daemon

Purpose

Starts the Internet Message Access Protocol (IMAP) server process.

Syntax

imapd [**-c**]

Description

The **imapd** command is an IMAP4 server. It supports the IMAP4 remote mail access protocol. Also, it accepts commands on its standard input and responds on its standard output. You normally invoke the **imapd** command with the **inetd** daemon with those descriptors attached to a remote client connection.

The **imapd** command works with the existing mail infrastructure consisting of **sendmail** and **bellmail**.

Flags

Item	Description
-c	Suppresses the reverse host name lookup.

Exit Status

All error and status information is written to a logfile if **syslogd** is configured for logging.

Security

The **imapd** daemon is a PAM-enabled application with a service name of *imap*. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the **usw** stanza of **/etc/security/login.cfg**, to PAM_AUTH as the root user.

The authentication mechanisms used when PAM is enabled depend on the configuration for the **imap** service in **/etc/pam.conf**. The **imapd** daemon requires **/etc/pam.conf** entries for the **auth** and **session** module types. Listed below is a recommended configuration in **/etc/pam.conf** for the **imap** service:

```
#
# AIX imap configuration
#
imap auth      required     /usr/lib/security/pam_aix

imap session   required     /usr/lib/security/pam_aix
```

Files

Item	Description
/usr/sbin/imapd	Contains the **imapd** command.
/etc/services	Specifies the file with port assignments for required services. The following entry must be in this file: `imap2  143/tcp  # Internet Mail Access Protocol`

imapds Daemon

Purpose

Starts the Internet Message Access Protocol (IMAP) server process over TSL/SSL.

Syntax

imapds [**-c**]

Description

The **imapds** command is an IMAP4 server. It supports the IMAP4 remote mail access protocol. Also, it accepts commands on its standard input and responds on its standard output. You normally invoke the **imapds** command with the **inetd** daemon with those descriptors attached to a remote client connection.

The **imapds** command works with the existing mail infrastructure consisting of **sendmail** and **bellmail**.

Flags

Item	Description
-c	Suppresses the reverse host name lookup.

Exit Status

All error and status information is written to a logfile if **syslogd** is configured for logging.

Security

The **imapds** daemon is a PAM-enabled application with a service name of *imap*. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the **usw** stanza of **/etc/security/login.cfg**, to PAM_AUTH as the root user.

The authentication mechanisms used when PAM is enabled depend on the configuration for the **imap** service in **/etc/pam.conf**. The **imapds** daemon requires **/etc/pam.conf** entries for the **auth** and **session** module types. Listed below is a recommended configuration in **/etc/pam.conf** for the **imap** service:

```
#
# AIX imap configuration
#
imap auth      required     /usr/lib/security/pam_aix

imap session   required     /usr/lib/security/pam_aix
```

Files

Item	Description
/usr/sbin/imapds	Contains the **imapds** command.
/etc/services	Specifies the file with port assignments for required services. The following entry must be in this file: `imaps  993/tcp  # imap4 protocol over TLS/SSL`

impfilt Command

Purpose

Imports filter rules from an export file.

Syntax

impfilt [**-v** **4**|**6**] **-f** *directory* [**-l** *filt_id_list*]

Description

Use the **impfilt** command to import filter rules from text export file(s) that are generated by the **expfilt** command. IPsec filter rules for this command can be configured using the `genfilt` command or IPsec smit (IP version 4 or IP version 6).

Flags

Item	Description
-v	IP version of the rules to be imported. The value of **4** specifies IP version 4 and the value of **6** specifies IP version 6. When this flag is not used, both IP version 4 and IP version 6 are imported.
-f	Specifies the directory where the imported text files are to be read.
-l	Lists the IDs of the filter rules to be imported. The filter rule IDs can be separated by "**,**". If this flag is not used, all filter rules for the applicable IP version(s) in the text export files will be imported.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

importvg Command

Purpose

Imports a new volume group definition from a set of physical volumes.

Syntax

importvg [**-V** *MajorNumber*] [**-y** *VolumeGroup*] [**-f**] [**-c**] [**-x**] **|** [**-L** *VolumeGroup*] [**-n**] [**-F**] [**-R**] [**-I**] [**-O**] [**-r** *file_name*] *PhysicalVolume*

Description

The **importvg** command makes the previously exported volume group known to the system. The *PhysicalVolume* parameter specifies only one physical volume to identify the volume group; any remaining physical volumes (those belonging to the same volume group) are found by the **importvg** command and included in the import. An imported volume group is automatically varied unless the volume group is Concurrent Capable. You must use the **varyonvg** command to activate Concurrent Capable volume groups before you access them.

When a volume group with file systems is imported, the **/etc/filesystems** file is updated with values for the new logical volumes and mount points. After importing the volume group and activating it with the **varyonvg** command, you must run the **fsck** command before the file systems can be mounted. However, the mount point information would be missing from the LVCB (logical volume control block) if it is longer than 128 characters. In this case, the **importvg** command will not be able to update the **/etc/filesystems** file with the stanza for the newly imported logical volume. You should manually edit the **/etc/filesystems** file to add a new stanza for this logical volume.

The **importvg** command changes the name of a logical volume if the name already exists in the system. It prints a message and the new name to standard error, and updates the **/etc/filesystems** file to include the new logical volume name. If the **importvg** command renames any filesystem log logical volumes, you must manually update any file systems using that log device to know about the renamed device.

Notes:

1. To use this command, you must either have root user authority or be a member of the **system** group.
2. As part of the **importvg** process, the volume group is automatically varied on by the system after it is imported. However, if the volume group is Concurrent Capable then the **importvg** command prompts you to **varyonvg** the imported volume group manually.
3. A volume group with a mirrored striped logical volume cannot be back ported into a version older than AIX 4.3.3.

You can use the System Management Interface Tool (SMIT) **smit importvg** fast path to run this command.

Flags

Item	Description
-c	This flag is ignored. Only Enhanced Concurrent Capable volume groups will be created.
-f	Forces the volume group to be varied online.

Item	Description
-L*VolumeGroup*	Takes a volume group and learns about possible changes performed to that volume group. Any new logical volumes created as a result of this command emulate the ownership, group identification, and permissions of the **/dev** special file for the volume group listed in the **-y** flag. The **-L** flag performs the functional equivalent of the **-F** and **-n** flags during execution. **Restrictions:** • The volume group must not be in an active state on the system executing the **-L** flag. • The volume group's disks must be unlocked on all systems that have the volume group varied on and operational. Volume groups and their disks may be unlocked, remain active and used via the **varyonvg -b -u** command. • The physical volume name provided must be of a good and known state, the disk named may not be in the missing or removed state. • If a logical volume name clash is detected, the command will fail. Unlike the basic **importvg** actions, clashing logical volume names will not be renamed.
-F	Provides a fast version of **importvg** that checks the Volume Group Descriptor Areas of only the disks that are members of the same volume group. As a result, if a user exercises this flag, they must ensure that all physical volumes in the volume group are in a good and known state. If this flag is used on a volume group where a disk may be in missing or removed state, the command may fail or the results may be inconsistent.
-I	Causes the **importvg** command to fail if imfs fails.
-n	Causes the volume not to be varied at the completion of the volume group import into the system.
-O	Forces varyon the volume group even if it is varied on in some other node. **Note:** In AIX 61 TL8 and later releases, **varyonvg** command updates the LVM metadata and ODM with varyon state of the volume group. During varyon time, **varyonvg** command reads this data and fails if the volume group is already varied in another node. **Varyoffvg** command resets the varyon state of the volume group during varyoff time. If system crashes before varying off the volume group or the volume group is forced off, then **varyonvg** command will fail after reboot. In this scenario, use **–O** flag to force varyon the volume group.
-R	Restores the ownership, group ID, and permissions of the logical volume special device files. These values will be restored only if they were set using **U**, **G** and **P** flags of **mklv** and **chlv** commands. This flag is applicable only for volume groups of the types big and scalable.
-r *file_name*	Restores the performance tunable parameters of the volume group. To use this flag, you must specify the file name that was backed up when you ran the **exportvg -b** command.
-V *MajorNumber*	Specifies the major number of the imported volume group.
-x	This flag is ignored. Only Enhanced Concurrent Capable volume groups will be created.

Attention: This entry must be added after the entry used to initiate **srcmstr**.

Item	Description
-y *VolumeGroup*	Specifies the name to use for the new volume group. If this flag is not used, the system automatically generates a new name. The volume group name can only contain the following characters: "A" through "Z," "a" through "z," "0" through "9," or "_" (the underscore), "-" (the minus sign), or "." (the period). All other characters are considered invalid.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To import the volume group bkvg from physical volume hdisk7, enter:

   ```
   importvg  -y bkvg hdisk7
   ```

   The volume group bkvg is made known to the system.
2. To use the -L on a multi-tailed system:

   ```
   Node A has the volume group datavg varied on.
   Node B is aware of datavg, but it is not varied on.
   Node A: varyonvg -b -u datavg
   Node B: importvg -L datavg hdisk7
   Node A: varyonvg datavg
   ```

Files

Item	Description
/usr/sbin	Directory where the **importvg** command resides.
/tmp	Directory where the temporary files are stored while the command is running.

imptun Command

Purpose

Adds the exported tunnel definitions and optional user-defined filter rules associated with the tunnels to the local host.

Syntax

imptun -f *directory* [**-t** *tunnel_id_list*] [**-v 4** | **6**] [**-n**] [**-r**] [**-g**] [**-l manual**]

Description

Use the **imptun** command to add exported tunnel definitions and optional user-defined filter rules associated with the exported tunnels (files generated by the tunnel owner by using the **exptun** command) to the local host. This command can also import tunnel definitions from the exported files generated by the IBM firewall (SNG) product export command.

A new tunnel ID is generated by the local host when a tunnel is imported to the local tunnel table. The auto-generated filter rules associated with the tunnel also is generated automatically. Importing the exported user-defined filter rules is optional.

If the exported files are transmitted by diskette, it is assumed they will be loaded to a local file directory using a command such as **tar**, depending on the tunnel owner's instructions.

Flags

Item	Description
-f	Specifies the directory from where the exported files will be read.
-g	The suppress system auto-generated filter rule flag. If the **-g** flag is not used, the **imptun** command generates two filter rules for each imported tunnel automatically. The auto-generated filter rules allow all traffic between the two end points of the tunnel to go through the tunnel. If the **-g** flag is specified, the command only imports the tunnel IBM definitions, and the user must add user-defined filter rules to use the tunnel.
-l	Specifies the type of the tunnel(s) you want to import. If **manual** is specified, only manual tunnel(s) are imported. **-n** and **-l** flags are mutually exclusive.
-n	Specifies that the export files were generated by the IBM firewall (version 2.2) tunnel export command. This flag cannot be specified with the **-v** flag. The **-n** flag is also mutually exclusive with the **-r** flag.
-r	Imports the user-defined filter rules associated with the tunnels that are being imported. To use the **-r** flag, it must have been specified with the **exptun** command when the exported files were generated. The **-r** flag is mutually exclusive with the **-n** flag.
-t	Lists the set of tunnel IDs to be imported from the export files. The tunnel definitions identified by these tunnel IDs are added to the local host. If this flag is not used, all the tunnel definitions in the export files are added to the local host.
-v	Specifies the IP version of the tunnel definitions from the exported files that you wish to import. If the **-v** flag is not given, then all IP version 4 and IP version 6 tunnel definitions that exist in the export files are imported.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

inc Command

Purpose

Files new mail in a folder.

Syntax

inc [**+** *Folder*] [**-noaudit** | **-audit** *File*] [**-changecur** | **-nochangecur**] [**-form** *FormFile* | **-format** *String*] [**-help**] [**-file** *File*] [**-truncate** | **-notruncate**] [**-nosilent** | **-silent**] [**-width** *Number*]

Description

The **inc** command files incoming mail in a specified folder and outputs a list of the messages filed. A folder is a system directory. By default, the **inc** command removes the new messages from your mail drop and places them in the specified folder. To file new mail without deleting the mail drop, use the **-notruncate** flag.

If the specified folder does not exist, the **inc** command prompts you for permission to create it. The system creates the folder as a subdirectory of the user's Message Handler (MH) directory. The default folder is **inbox**.

> **Note:** If you do not have a `Path:` entry specified in your **.mh_profile** file, the **inc** command creates the folder as a subdirectory of the current directory.

Filed messages are assigned consecutive message numbers starting with the next highest number in the folder. Each new message receives the protection code specified in the `Msg-Protect:` entry in your **.mh_profile** file. If the `Msg-Protect:` entry does not exist, a protection code of 644 is assigned. If the `Unseen-Sequence:` entry exists, new messages are added to each sequence specified by the entry.

Flags

Item	Description
-audit *File*	Copies the current date to the specified file and appends the output of the **inc** command to the file.
-changecur	Sets the first new message as the current message for the specified folder. This flag is the default.
-file *File*	Files messages from the specified file instead of the user's maildrop.
+*Folder*	Specifies the folder in which to place new messages. By default, the system creates a subdirectory called **inbox** in the user's MH directory.
-form *FormFile*	Identifies a file that contains an alternate output format for the **inc** command.
-format *String*	Specifies a string that defines an alternate output format for the **inc** command.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-noaudit	Suppresses recording of information about any new messages filed. This is the default.
-nochangecur	Prevents alteration of the current message for the specified folder.
-nosilent	Prompts the user for any necessary information. This flag is the default.
-notruncate	Prevents clearing of the mailbox or file from which the **inc** command is taking new messages. If the **-file** flag is specified, the **-notruncate** flag is the default.
-silent	Prevents prompting by the **inc** command for information. This flag is useful when running the **inc** command in the background.
-truncate	Clears the mailbox or file from which the **inc** command is taking new messages. If the **-file** flag is not specified, the **-truncate** flag is the default.
-width *Number*	Sets the number of columns in the command output. The default is the width of the display.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Alternate-Mailboxes:`	Specifies alternate mailboxes.
`Folder-Protect:`	Sets the protection level for new folder directories.
`Msg-Protect:`	Sets the protection level for new message files.

Item	Description
`Path:`	Specifies the user's MH directory.
`Unseen-Sequence:`	Specifies the sequences used to keep track of unseen messages.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To incorporate new mail into the default mail folder, **inbox**, enter:

   ```
   inc
   ```

   If the **inbox** folder exists, the system displays a message similar to the following:

   ```
   Incorporating new mail into inbox...
   65+        04/08    jim@athena.a         Meeting                                 <<The
   meeting will
   66         04/08    jim@athena.a         Schedule                                <<Schedule
   change
   ```

   In this example, two messages are filed in the `inbox` folder. The subject of the first message is `Meeting`, and the first line starts with the words `The meeting will`. The subject of the second message is `Schedule`, and the first line starts with the words `Schedule change`.

2. To incorporate new mail into a new folder called **test cases**, enter:

   ```
   inc     +test cases
   ```

   The system prompts you as follows:

   ```
   Create folder "/home/mary/test cases"?
   ```

   If you wish to create the folder, enter:

   ```
   yes
   ```

   A message similar to the following is displayed:

   ```
   Incorporating new mail into test cases...
   67+        04/08    jim@athena.a            Meeting                                 <<We will
   begin
   68         04/08    jim@athena.a            Schedule
   <<Schedule change
   ```

Files

Item	Description
$HOME/.mh_profile	Customizes the MH user profile.
/etc/mh/mtstailor	Tailors the MH environment to the local environment.
/var/spool/mail/$USER	Specifies the location of the mail drop.
/usr/bin/inc	Contains the **inc** command.

indent Command

Purpose

Reformats a C language program.

Syntax

indent *InputFile* [*OutputFile*] [**-nbad** | **-bad**] [**-nbap** | **-bap**] [**-nbbb** | **-bbb**] [**-nbc** | **-bc**] [**-br** | **-bl**] [**-c***n*] [**-cd***n*] [**-ncdb** | **-cdb**] [**-nce** | **-ce**] [**-ci***n*] [**-cli***n*] [**-d***n*] [**-di***n*] [**-ndj** | **-dj**] [**-nei** | **-ei**] [**-fa**] [**-nfa**] [**-nfc1** | **-fc1**] [**-i***n*] [**-nip** | **-ip**] [**-l***n*] [**-lc***n*] [**-nlp** | **-lp**] [**-npro**] [**-npcs** | **-pcs**] [**-nps** | **-ps**] [**-npsl** | **-psl**] [**-nsc** | **-sc**] [**-nsob** | **-sob**] [**-nslb** | **-slb**] [**-st**] [**-troff**] [**-nv** | **-v**] [**-T***Type*] ...

Description

The **indent** command reformats a C program as specified by flags entered with the command.

If you only specify the *InputFile* parameter, the reformatted file is written back into the *InputFile* parameter and a backup copy of the *InputFile* parameter is written in the current directory with a **.BAK** filename suffix.

If you specify the *OutputFile* parameter, the **indent** command checks to make sure its name is different from the *InputFile* parameter.

To set up your own profile of defaults for the **indent** command, create a file called **.indent.pro** in your login directory or the current directory. In this file, include as many flags as desired, separated by spaces, tabs, or new lines.

Flags in the **.indent.pro** file in the current directory override those in your login directory (with the exception of **-T***Type* flags, which accumulate). If the **indent** command is run and a profile file exists, the profile file is read to set up the defaults of the program. Flags on the command line, however, override profile flags.

Comment Handling

The **indent** command assumes that any comment with a **-** (dash) or ***** (asterisk) immediately after the start of a comment marker (**/*-** or **/****) is a comment surrounded by asterisks. Each line of the comment is left unchanged, except for its indentation. This indentation can be adjusted to account for the change in indentation of the first line of the comment.

All other comments are treated as text. The **indent** command fits as many words (separated by blanks, tabs, or new-lines) on a line as possible. Blank lines break paragraphs.

A block comment is a comment that is not to the right of the code, and extends for more than one line.

If a comment is on a line with code, it is started in the comment column set by the **-c***n* flag. Otherwise, the comment is started at *n* indentation levels less than where code is currently being placed, where *n* is specified by the **-d***n* flag. If the code on a line extends past the comment column, the comment starts further to the right. The right margin can be extended automatically in extreme cases.

Preprocessor Lines Handling

In general, the **indent** command leaves preprocessor lines alone. The only reformatting it does is to straighten up trailing comments. It leaves embedded comments alone. Conditional compilation (code between **#ifdef** and **#endif** lines) is recognized and the **indent** command attempts to compensate correctly for the syntactic peculiarities introduced.

C Syntax Handling

The parser built into the **indent** command attempts to cope with incomplete and misformed syntax. In particular, the use of macros like:

```
#define forever for(;;)
```

is handled properly. For best results, use the **indent** command on source that is syntactically correct.

Flags

Note: Flags can appear before or after file names.

Item	Description
-bad	Forces a blank line after every block of declarations.
-nbad	Suppresses a blank line after every block of declarations; active unless turned off with the **-bad** flag.
-bap	Forces a blank line after every procedure body.
-nbap	Suppresses a blank line after every procedure body; active unless turned off with the **-bap** flag.
-bbb	Forces a blank line before every block comment.
-nbbb	Suppresses a blank line before every block comment; active unless turned off with the **-bbb** flag.
-bc	Forces a new line after each comma in a declaration.
-nbc	Suppresses a new line after each comma in a declaration; active unless turned off with the **-bc** flag.
-bl	Formats compound statements, structure initializations, and enum initializations, as follows: `if (...)` `{` `  code` `}`
-br	Formats compound statements, structure initializations, and enum initializations, as follows: `if (...) {` `  code` `}` This flag is active unless turned off with the **-bl** flag.
-c*n*	Sets the initial tab position for comments on code to the *n* variable. The default value is 33.
-cd*n*	Sets the initial tab position for comments on declarations to the *n* variable. By default, this flag uses the value defined with the **-c** flag.
-cdb	Enables placing comment delimiters on blank lines; active unless turned off with the **-ncdb** flag. The **-cdb** flag affects only block comments, not comments to the right of code. Resulting comments look like the following: `/*` ` * this is a comment` ` */`
-ncdb	Disables placing comment delimiters on blank lines. The **-ncdb** flag affects only block comments, not comments to the right of code. Resulting comments look like the following: `/* this is a comment */`
-ce	Enables forcing **else** statements to follow the immediately preceding **}** (left bracket); active unless turned off with the **-nce** flag.
-nce	Disables forcing **else** statements to follow the immediately preceding **}** (left bracket).

Item	Description
-ci*n*	Indents the continuation lines *n* positions from the beginning of the first line of the statement. Expressions in parentheses have extra indentation added to indicate the nesting, unless the **-lp** flag is in effect. By default, this flag uses the value defined by the **-i** flag.
-cli*n*	Indents the case labels *n* positions to the right of the containing flag statement. Entering **-cli0.5** causes case labels to be indented half a tab stop. This option is the only one that takes a fractional argument. By default, the value is **-cli0**.
-d*n*	Controls the placement of comments that are not to the right of code with the *n* variable. Specifying the **-d1** flag causes such comments to appear one indention level to the left of code. By default, this flag uses **-d0** and comments are aligned with code. The location of comment lines relative to program code affects the comment indention.
-di*n*	Specifies the number of positions to indent an identifier from a preceding declaration keyword with the *n* variable. By default, this flag uses **-di16**.
-dj	Left-justifies declarations.
-ndj	Indents declarations; active unless turned off with the **-dj** flag.
-ei	Enables special **else**-**if** processing; active unless turned off with the **-nei** flag. The **-ei** flag causes **if** statements following **else** statements to have the same indentation as the preceding **if** statement.
-nei	Disables special **else**-**if** processing.
-fa	Flips assign operators from old style C code to the ANSI format. This flag remains active unless turned off with the **-nfa** flag. **Attention:** The possibility of changing the meaning of the code exists if the code was meant for the ANSI compiler. For example, A=-B becomes A-=B. **Note:** Use no spaces between operators. If the user means subtraction, then the flipping is necessary; on the other hand, if the user means A equals the negative of B, the flipping alters the meaning.
-nfa	Suppresses flipping the operators. Use this flag if the code is written for an ANSI compiler.
-fc1	Enables formatting comments that start in column 1; active unless turned off with the **-nfc1** flag.
-nfc1	Disables formatting comments that start in column 1.
-i*n*	Sets the indentation level size. By default, the level size is 8 positions.
-ip	Enables indenting parameter declarations; active unless turned off with the **-nip** flag.
-nip	Disables indenting parameter declarations.

Item	Description
-l*n*	Sets the maximum column position of comments that are to the right of the code. If the comment does not fit on a line, a maximum of 25 characters are printed.
-lc*n*	Sets the maximum line length for block comments to the *n* variable. By default, this flag uses the length specified with the **-l** flag.

Item	Description
-lp	Aligns code surrounded by parentheses in continuation lines; active unless turned off with the **-nlp** flag. If a line has a left parenthesis with no matching right parenthesis on that line, continuation lines start at the position following the left parenthesis. With the **-lp** flag in effect, such lines appear as follows: `p1 = first_procedure(second_procedure(p2,p3),` `third_procedure(p4,p5));` Inserting two more new lines yields the following: `p1 = first_procedure(second_procedure(p2,` `p3),` `third_procedure(p4,` `p5));`
-nlp	Leaves code surrounded by parentheses in continuation lines unaligned. With the **-nlp** flag in effect, such lines appear as follows: `p1 = first_procedure(second_procedure(p2,p3),` `third_procedure(p4, p5));`
-npro	Causes the profile files **./.indent.pro** and **$HOME/.indent.pro** to be ignored.
-pcs	Inserts a space between each procedure call name and the following **(** (left parenthesis).
-npcs	Suppresses a space between each procedure call name and the following **(** (left parenthesis); active unless turned off with the **-pcs** flag.
-ps	Inserts spaces on both sides of the pointer following the **->** operator.
-nps	Suppresses spaces on both sides of the pointer following the **->** operator; active unless turned off with the **-ps** flag.
-psl	Left-justifies the names of procedures being defined; active unless turned off with the **-npsl** flag. The procedure types, if any, remain on the previous lines.
-npsl	Disables left-justification of names of defined procedures.
-sc	Enables the placement of ***** (asterisks) to the left of comments; active unless turned off with the **-nsc** flag.
-nsc	Disables the placement of ***** (asterisks) to the left of comments.
-slb	Treats any single-line comment that is not to the right of the code as a block comment.
-nslb	Disables treating any single-line comment that is not to the right of the code as a block comment; active unless turned off with the **-slb** flag.
-sob	Removes optional blank lines. Works in combination with any of the following flags: **-nbad**, **-nbap**, or **-nbbb**. Removes only blank lines that were inserted by the **-bad**, **-bap**, or **-bbb** flags.
-nsob	Retains optional blank lines; active unless turned off with the **-sob** flag.
-st	Causes the **indent** command to take its input from stdin and output to stdout.
-T*Type*	Adds the *Type* variable to the list of type keywords. Names accumulate so **-T** can be specified more than once. You should specify all the types appearing in your program defined by **typedef** statements to produce the best output from the **indent** command.
-troff	Formats the C program for processing by **troff**. Produces a listing similar to listings produced by the **vgrind** command. If no output file is specified, the default is standard output, rather than formatting in place.

Item	Description
-v	Turns on verbose mode, which reports when one line of input is split into two or more lines of output and gives size statistics at completion.
-nv	Turns off verbose mode; active unless turned off with the **-v** flag.

Examples

1. To format the `test.c` file using the default settings of the **indent** command and place the output into the `newtest.c` file, enter:

```
indent test.c newtest.c
```

2. To format the `test.c` file so that a blank line is forced after every block of declarations and procedure body, use all other default settings, and store the output in the `newtest.c` file, enter:

```
indent test.c newtest.c -bad -bap
```

3. To format the `test.c` file using the default settings of the **indent** command and to define `uint` as a type keyword recognizable to the **indent** command, enter:

```
indent test.c newtest.c -Tuint
```

Files

Item	Description
./.indent.pro	Contains the profile file.
$HOME/.indent.pro	Contains the profile file.
/usr/ccs/bin/indent	Contains the **indent** command.

indxbib Command

Purpose

Builds an inverted index for a bibliography.

Syntax

indxbib *Database* ...

Description

The **indxbib** command makes an inverted index to the named database (or files) for use by the **lookbib** and **refer** commands. These files contain bibliographic references (or other kinds of information) separated by blank lines.

Note: The **indxbib** command expects the database to exist in the current working directory.

A bibliographic reference is a set of lines, constituting fields of bibliographic information. Each field starts on a line beginning with a **%** (percent sign), followed by a key letter, then a space character, and finally the contents of the field, which can continue until the next line starting with a **%** (percent sign). All key letters are ASCII characters.

The **indxbib** command is a shell script that calls the **/usr/lib/refer/mkey** and **/usr/lib/refer/inv** files. The first program, **mkey**, performs the following operations:

1. Truncates words (delimited by blanks or tabs) to six characters.
2. Maps uppercase to lowercase characters.
3. Discards words shorter than three characters.
4. Discards the most commonly used words according to an existing **ign** file. An English language file, **/usr/lib/eign**, has been provided with a list of common English words. It is suggested, but not necessary, that users create their own files, named **ign**, consisting of language-specific common words. This file, if created, should exist in the **/usr/lib/nls/msg/$LANG** directory.
5. Discards numbers (dates) less than 1900 or greater than 2099.

   Note: All dates should be indexed because many disciplines refer to literature written in the 1800s or earlier.

The second program, **inv**, creates in the working directory an entry file (**.ia**), a posting file (**.ib**), and a tag file (**.ic**).

Files

Item	Description
/usr/lib/eign	Contains the default list of common words the **indxbib** command discards while processing.
Database.**ia**	Contains the entry file.
Database.**ib**	Contains the posting file.
Database.**ic**	Contains the tag file.

Environment Variables

Item	Description
NLSPATH	Refers to a list of directory names where the message catalog files can be found.

inetd Daemon

Purpose

Provides Internet service management for a network.

Syntax

Note: Use SRC commands to control the **inetd** daemon from the command line. Use the **rc.tcpip** file to start the daemon with each system restart.

/usr/sbin/inetd [**-d**] [**-t** *SecondsToWait*] [*ConfigurationFile*]

Description

The **/usr/sbin/inetd** daemon provides Internet service management for a network. This daemon reduces system load by invoking other daemons only when they are needed and by providing several simple Internet services internally without invoking other daemons.

The **inetd** daemon starts by default each time you start your system. When the daemon starts, it reads its configuration information from the file specified in the *ConfigurationFile* parameter. If the parameter is not specified, the **inetd** daemon reads its configuration information from the **/etc/inetd.conf** file.

Once started, the **inetd** daemon listens for connections on certain Internet sockets in the **/etc/inetd.conf**. The **/etc/inetd.conf** file describes to the **inetd** daemon how Internet service requests on Internet sockets should be handled. When the **inetd** daemon receives a request on one of these sockets, it determines

which service corresponds to that socket and then either handles the service request itself or invokes the appropriate server.

Subservers of the inetd Daemon

The **inetd** daemon (a subsystem) controls the following daemons (subservers):

- **comsat** daemon
- **ftpd** daemon
- **fingerd** daemon
- **rlogind** daemon
- **rexecd** daemon
- **rshd** daemon
- **talkd** daemon
- **telnetd** daemon
- **tftpd** daemon
- **uucpd** daemon.

The **ftpd**, **rlogind**, **rexecd**, **rshd**, **talkd**, **telnetd**, and **uucpd** daemons are started by default. The **tftpd**, **fingerd**, and **comsat** daemons are not started by default unless they are uncommented in the **/etc/inetd.conf** file.

Inetd Configuration File

The **/etc/inetd.conf** file can be updated by using the System Management Interface Tool (SMIT), the System Resource Controller (SRC), or by editing the **/etc/inetd.conf**.

If you change the `/etc/inetd.conf` file, using SMIT, then the **inetd** daemon will be refreshed automatically and will read the new `/etc/inetd.conf` file. If you change the `/etc/inetd.conf` file using any other editor, run the **refresh -s inetd** or **kill -1** *InetdPID* command to inform the **inetd** daemon of the changes to its configuration file. If you run the **refresh -s inetd** command, the running services continue to run with old configuration until the services terminate and the new services are created.

The entries in the `/etc/inetd.conf` file include the following information:

Item	Description
Service Name	Specifies the name of a valid Internet service.
Socket Type	Specifies the type of Internet socket used for the Internet service. (Only stream and datagram sockets are implemented.) Valid values are: stream dgram sunrpc_udp sunrpc_tcp
Protocol	Specifies the Internet Protocol used for the Internet service. Valid values are: tcp tcp6 udp udp6
Wait/Nowait	Specifies whether the **inetd** daemon should wait for the service to complete before continuing to listen for this type of service request.

Item	Description
Wait/Nowait	Specifies whether the **inetd** daemon should wait for the service to complete before continuing to listen for this type of service request. SRC works like wait, but instead of forking and waiting for the child to die, it does a **startsrc** on the subsystem and store information about the starting of the service. When the service is removed from the **inetd.conf** file and **inetd** is restarted, the service has a **stopsrc** issued to the service to stop it.
User	Specifies the user name that **inetd** should use to start the subserver.
Path	Specifies the fully qualified path name that inetd should execute to provide the service. For services that **inetd** provides internally, this entry should be internal.
Command	Specifies the name of the service to start and its parameters. This field is empty for internal services.

The **inetd** daemon can be run with or without the SRC. In addition, the inetd daemon can be controlled by issuing signals using the kill command.

Flags

Item	Description
-d	Sends debugging messages to the syslogd daemon.
-t *SecondsToWait*	Specifies the number of seconds to wait in the select() system call before looping. The *SecondsToWait* can be a number from 1 to 999999. Without this flag the **inetd** daemon will block until one of the active services is requested by a network connection. This flag should only be used when a machine is servicing many wait services like **tftp** and is not being used for other services. Since timing out the select() system call will cause the **inetd** daemon to use more CPU cycles, this flag is not recomended for most situations.

Service Requests

The Internet service requests that are supported internally by the **inetd** daemon are generally used for debugging. They include the following internal services:

Item	Description
ECHO	Returns data packets to a client host.
DISCARD	Discards received data packets.
CHARGEN	Discards received data packets and sends predefined or random data.
DAYTIME	Sends the current date and time in user-readable form.
TIME	Sends the current date and time in machine-readable form.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

infocmp Command

Purpose

Manages **terminfo** descriptions.

Syntax

infocmp [**-d**] [**-c**] [**-n**] [**-I**] [**-L**] [**-C**] [**-r**] [**-u**] [**-s** { **d**| **i**| **l**| **c**}] [**-v**] [**-V**] [**-1**] [**-w** *Width*] [**-A** *Directory*] [**-B** *Directory*] [*TermName...*]

Description

The **infocmp** command manages **terminfo** descriptions. You can use this command to:

- Compare a binary **terminfo** entry with other **terminfo** entries.
- Print a **terminfo** description from the binary file.
- Rewrite a **terminfo** description to take advantage of the **use** attribute.

The **infocmp** command prints the Boolean attributes first, the numeric attributes second, and the string attributes last.

Comparing Entries

Use the **-d**, **-c**, and **-n** flags to compare entries. The **-d** flag returns the differences between entries. The **-c** flag produces a list of the capabilities that are set and in common between two entries. The **-n** flag returns a list of the capabilities that neither entry has.

To compare **terminfo** entries, you specify two or more *TermName* parameters. The **infocmp** command compares the **terminfo** description of the first *TermName* parameter with each of the descriptions for the subsequent *TermName*s specified. If a capability is defined for only one of the terminal descriptions, the value returned will depend on the type of capability. For Boolean capabilities the **infocmp** command returns an F, the command returns a -1 for integer capabilities, and null for string capabilities.

Producing a Source Listing

Use the **-I** (uppercase i), **-L**, **-C**, and **-r** flags to produce a source listing for one or more terminals. If you do not specify a *TermName* parameter, the system uses the **TERM** environment variable. You can use these source options to produce a source file for a terminfo binary when one is not available.

The **I** (uppercase i) flag produces a listing with the terminfo names. The **-L** flag produces a listing using the long **C** variable names listed in **/usr/include/term.h**.

The **-C** flag uses **termcap** names instead of terminfo capability names when producing the source listing. The **infocmp** commands translates and outputs only those **terminfo** capabilities that have a corresponding **termcap** code name. To remove this restriction, specifying the **-r** flag. This flag causes the command to output **terminfo** capabilities that cannot be translated into **termcap** format.

When using the **-C** and **-r** flags, the **infocmp** command notes any string parameters it was unable to convert to the **termcap** format. You must edit these parameters manually. The command collects all padding information for strings together and places it at the beginning of the string where **termcap** expects it. Mandatory padding is optional after translation. Mandatory padding is padding information with a trailing **/** (slash).

> **Note:** The **-C** and **-r** flags cannot always convert a **terminfo** string into its equivalent **termcap** form. Similarly, a conversion from the **termcap** file format back into the **terminfo** file format does not necessarily reproduce the original source.

Definitions with the use Attribute

Given a list of terminal menus and the **-u** flag, the **infocmp** command compares the first terminal's description against the other terminal descriptions. The **infocmp** command then creates a new description for the first terminal using as much of the subsequent terminal descriptions as possible.

When you specify the **-u** flag and a list of terminal names, the **infocmp** command does the following:

- Compares subsequent terminal descriptions against the first.
- Creates a description of the first terminal you specified relative to the description of the other terminals.

The new description for the first terminal will have the following:

- Capabilities that exist in the subsequent terminals but do not exist for the first terminal will appear with an @ in the resulting description.

  Note: The @ implies that the capability does not exist.
- Capabilities defined in a subsequent terminal with the same value are replaced with *use=<subsequent terminal>*.
- Any capabilities in the first terminal not found in any of the other terminals are printed along with the corresponding values.
- If the first terminal has a capability whose value differs from the value found in at least one of the other terminals, the capability is printed.

You can change a description and specify a capability after the **use** attribute. If this capability is also found in the terminal referenced by the **use** attribute, the second capability takes precedence over the one referenced by the **use** attribute.

Changing Databases

By default, terminal descriptions appear in the system **terminfo** database directory, **/usr/share/lib/terminfo**. You can specify a different database location with the **TERMINFO** environment variable. The **infocmp** command first checks to see if this variable exists. If the variable does not exist, the command uses the system **terminfo** database.

You can use the **-A** and **-B** flag with the **infocmp** command to override the system database. The **-A** flag identifies the **terminfo** database for the first *TermName* parameter. The **-B** flag identifies the database to use for any subsequent terminals you name. Together, these flags make it possible to compare descriptions for two terminals with the same name located in two different databases.

Flags

Item	Description
-A *Directory*	Identifies the **terminfo** database for the first *TermName* parameter.
-B *Directory*	Identifies the **terminfo** database for every *TermName* parameter except the first.
-C	Uses the **termcap** code names to produce the source listing. Will not list **terminfo** capabilities that cannot be translated to **termcap** format.
-c	Lists the capabilities that are common between the two entries. Capabilities that are not set are ignored. This flag can be used as a quick check to see if it is desirable to use the **-u** flag.
-d	Lists the capabilities that are different between terminals. You can use this flag to pinpoint the differences between similar terminal entries.
-I (uppercase i)	Uses the **terminfo** capability names when producing the source listing.
-1 (numeral)	Prints the capabilities one to a line. by default, the fields are printed several to a line to a maximum width of 60 characters.
-L	Uses the long C variable name listed in **/usr/include/term.h** file to produce the source listing.

Item	Description
-n	Compares two entries and lists the capabilities that do not exist in either. If you do not specify a *TermName* parameter, the system uses the **TERM** environment variable for both *TermName* parameters. You can use this as a quick check to see if anything was left out of the description.
-r	Instructs the **infocmp** command to output **terminfo** capabilities that cannot be translated to **termcap** format. This flag is valid only with the **-C** flag.
-s	Sorts the output from the **infocmp** command within each capability type (Boolean, numeric, and string) and according to the argument below: **d** Sort in the order specified in the **terminfo** database. **i** Sort by **terminfo** name. **l** Sort by the long C variable name. **c** Sort by the **termcap** name. If you do not specify an option with the **-s** flag, the command sorts each capability alphabetically by the **terminfo** name within each type. If you specify the **-C** or the **-L** flags with the **-s** flag, the capabilities are sorted by the **termcap** name or the long C variable name, respectively.
-u	Compares two or more terminal descriptions and produces new descriptions using the **use** attribute.
-v	Prints out tracing information on standard error.
-V	Prints out the version of the program in use on standard error and exits.
-w *Width*	Changes the output to the specified number of characters per line. The output includes as many fields as possible that can fit within the specified number of characters. **Note:** Fields are not truncated.

Examples

1. To list the common capabilities between the aixterm and lft terminals, enter:

```
infocmp -c aixterm lft
```

2. To list all of the capabilities that are possible but do not currently exist for the current terminal, enter:

```
infocmp -n
```

3. To produce a source listing for the lft terminal in **terminfo** format, enter:

```
infocmp -I lft
```

4. To produce a source listing for the terminal description my_term that is located in **/tmp** using as much of the lft description as possible, enter:

```
infocmp -A /tmp -u my_term lft
```

File

Item	Description
/usr/share/lib/terminfo	Contains the compiled terminal description database.

telinit or init Command

Purpose

Initializes and controls processes.

Syntax

{ **telinit** | **init** } { **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **a** | **b** | **c** | **h** | **Q** | **q** | **S** | **s** | **M** | **m** | **N** }

Description

The **init** command initializes and controls processes. Its primary role is to start processes based on records read from the **/etc/inittab** file. The **/etc/inittab** file usually requests that the **init** command run the **getty** command for each line on which a user can log in. The **init** command controls autonomous processes required by the system.

The process that constitutes the majority of the **init** command's process dispatching activities is **/usr/sbin/getty**. The **/usr/sbin/getty** process initiates individual terminal lines. Other processes typically dispatched by the **init** command are daemons and the shell.

The **telinit** command, which is linked to the **init** command, directs the actions of the **init** command. The **telinit** command takes a one-character argument and signals the **init** command by way of the **kill** subroutine to perform the appropriate action.

The **telinit** command sets the system at a specific run level. A run level is a software configuration that allows only a selected group of processes to exist. The system can be at one of the following run levels:

Item	Description
0-9	Tells the **init** command to place the system in one of the run levels **0-9**. When the **init** command requests a change to run levels **0-9**, it kills all processes at the current run levels and then restarts any processes associated with the new run levels.
0-1	Reserved for the future use of the operating system.
2	Contains all of the terminal processes and daemons that are run in the multiuser environment. In the multiuser environment, the **/etc/inittab** file is set up so that the **init** command creates a process for each terminal on the system. The console device driver is also set to run at all run levels so the system can be operated with only the console active.
3-9	Can be defined according to the user's preferences.
S,s,M,m	Tells the **init** command to enter the maintenance mode. When the system enters maintenance mode from another run level, only the system console is used as the terminal.

The following arguments also serve as directives to the **init** command:

Item	Description
a,b,c,h	Tells the **init** command to process only those records in the **/etc/inittab** file with **a**, **b**, **c**, or **h** in the run level field. These four arguements, **a**, **b**, **c**, and **h**, are not true run levels. They differ from run levels in that the **init** command cannot request the entire system to enter run levels **a, b, c**, or **h**. When the **init** command finds a record in the **/etc/inittab** file with a value of **a**, **b**, **c**, or **h** in the run level field, it starts the process. However, it does not kill any processes at the current run level; processes with a value of **a**, **b**, **c**, or **h** in the run level field are started in addition to the processes already running at the current system run level. Another difference between true run levels and **a**, **b**, **c**, or **h** is that processes started with **a**, **b**, **c**, or **h** are not stopped when the **init** command changes run levels. Three ways stop **a**, **b**, **c**, or **h** processes: • Type **off** in the *Action* field. • Delete the objects entirely. • Use the **init** command to enter maintenance state.
Q,q	Tells the **init** command to re-examine the **/etc/inittab** file.
N	Sends a signal that stops processes from being respawned.

During system startup, after the root file system has been mounted in the pre-initialization process, the following sequence of events occurs:

1. The **init** command is run as the last step of the startup process.
2. The **init** command attempts to read the **/etc/inittab** file.
3. If the **/etc/inittab** file exists, the **init** command attempts to locate an `initdefault` entry in the **/etc/inittab** file.
   a. If the `initdefault` entry exists, the **init** command uses the specified run level as the initial system run level.
   b. If the `initdefault` entry does not exist, the **init** command requests that the user enter a run level from the system console (**/dev/console**).
   c. If the user enters an **S**, **s**, **M** or **m** run level, the **init** command enters maintenance run level. These are the only run levels that do not require a properly formatted **/etc/inittab** file.
4. If the **/etc/inittab** file does not exist, the **init** command places the system in the maintenance run level by default.
5. The **init** command rereads the **/etc/inittab** file every 60 seconds. If the **/etc/inittab** file has changed since the last time the **init** command read it, the new commands in the **/etc/inittab** file are executed during system startup.

When you request the **init** command to change the run level, the **init** command reads the **/etc/inittab** file to identify what processes should exist at the new run level. Then, the **init** command cancels all processes that should not be running at the new level and starts any processes that should be running at the new level.

The processes run by the **init** command for each of these run levels are defined in the **/etc/inittab** file. The run level is changed by having a root user run the **telinit** command, which is linked to the **init** command. This user-run **init** command sends appropriate signals to the original **init** command initiated by the system during startup. The default run level can be changed by modifying the run level for the `initdefault` entry in the **/etc/inittab** file.

In the maintenance run level, the **/dev/console** console terminal is opened for reading and writing. The password for root is prompted. When the root password is entered successfully, the **su** command is invoked. Two ways exist to exit from the maintenance run level:

- If the shell is terminated, the **init** command requests a new run level.

  OR

- The **init** (or **telinit**) command can signal the **init** command and force it to change the run level of the system.

During a system startup attempt, apparent failure of the **init** command to prompt for a new run level (when **initdefault** is maintenance) may be due to the fact that the terminal console device (**/dev/console**) has been switched to a device other than the physical console. If this occurs and you wish to work at the physical console rather than the **/dev/console**, you can force the **init** command to switch to the physical console by pressing the DEL (delete) key at the physical console device.

When the **init** command prompts for a new run level, enter one of the digits **0** through **9** or any of the letters **S**, **s**, **M**, or **m**. If you enter **S**, **s**, **M**, or **m**, the **init** command operates in maintenance mode with the additional result that if control had previously been forced to switch to the physical console, the **/dev/console** file is switched to this device as well. The **init** command generates a message to this effect on the device to which the **/dev/console** file was previously connected.

If you enter a **0** through **9** run level, the **init** command enters the corresponding run level. The **init** command rejects any other input and re-prompts you for the correct input. If this is the first time the **init** command enters any run level other than maintenance, it searches the **/etc/inittab** file for entries with the **boot** or **bootwait** keywords. If the **init** command finds these keywords, it performs the corresponding task, provided the run level entered matches that of the entry. For example, if the **init** command finds the **boot** keyword, it boots the machine. Any special initialization of the system, such as checking and mounting file systems, takes place before any users are allowed on the system. The **init** command then scans the **/etc/inittab** file to find all entries that are processes for that level. It then resumes normal processing of the **/etc/inittab** file.

Run level **2** is defined by default to contain all of the terminal processes and daemons that are run in the multiuser environment. In the multiuser environment, the **/etc/inittab** file is set up so that the **init** command creates a process for each terminal on the system.

For terminal processes, the shell terminates either as a result of an end of file character (EOF) typed explicitly or as the result of disconnection. When the **init** command receives a signal telling it that a process has terminated, it records the fact and the reason it stopped in **/etc/utmp** file and /**var**/**adm**/**wtmp** file. The /**var**/**adm**/**wtmp** file keeps a history of the processes started.

To start each process in the **/etc/inittab** file, the **init** command waits for one of its descendant processes to stop, for a power fail signal **SIGPWR**, or until the **init** command is signaled by the **init** or **telinit** commands to change the system's run level. When one of the above three conditions occurs, the **init** command re-examines the **/etc/inittab** file. Even if new entries have been added to the **/etc/inittab** file, the **init** command still waits for one of the three conditions to occur. To provide for instantaneous response, re-examine the **/etc/inittab** file by running the **telinit -q** command.

If the **init** command finds that it is continuously running an entry in the **/etc/inittab** file (more than five times in 225 seconds), it assumes that an error in the entry command string exists. It then prints an error message to the console and logs an error in the system error log. After the message is sent, the entry does not run for 60 seconds. If the error continues to occur, the command will respawn the entry only five times every 240 seconds. The **init** command continues to assume an error occurred until the command does not respond five times in the interval, or until it receives a signal from a user. The **init** command logs an error for only the first occurrence of the error.

When the **init** command is requested to change run levels by the **telinit** command, the **init** command sends a **SIGTERM** signal to all processes that are undefined in the current run level. The **init** command waits 20 seconds before stopping these processes with the **SIGKILL** signal.

If the **init** command receives a **SIGPWR** signal and is not in maintenance mode, it scans the **/etc/inittab** file for special power fail entries. The **init** command invokes the tasks associated with these entries (if the run levels permit) before any further processing takes place. In this way, the **init** command can perform cleanup and recording functions whenever the system experiences a power failure. It is important to note that these power fail entries should not use devices that need to be initialized first.

Environments

Because the **init** command is the ultimate ancestor of every process on the system, every other process on the system inherits the **init** command's environment variables. As part of its initialization sequence,

the **init** command reads the **/etc/environment** file and copies any assignments found in that file into the environment passed to all of its subprocesses. Because **init** subprocesses do not run from within a login session, they do not inherit a umask setting from **init**. These processes may set the umask to whatever value they require. A command that is executed by **init** from the **/etc/inittab** file uses **init**'s ulimit values and not the default values as given in **/etc/security/limits**. The result is that a command that is successfully executed from the command line may not execute correctly when invoked by **init**. Any command that has specific **ulimit** requirements should include specific actions to set the **ulimit** values as required.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To request the **init** command to reexamine the **/etc/inittab** file, enter:

   ```
   telinit  q
   ```

2. To request the **init** command to enter maintenance mode, enter:

   ```
   telinit  s
   ```

Files

Item	Description
/etc/inittab	Specifies the **init** command control file.
/etc/utmp	Specifies the record of logged-in users.
/var/adm/wtmp	Specifies the permanent login accounting file.
/sbin/rc.boot	Specifies the pre-initialization command file.
/etc/rc	Specifies the initialization command file.
/etc/environment	Specifies system environment variables.
/dev/console	Specifies the console device driver.

install Command

Purpose

Installs a command.

Syntax

/usr/bin/install [**- c** *DirectoryA*] [**- f** *DirectoryB*] [**- i**] [**- m**] [**- M** *Mode*] [**- O** *Owner*] [**- G** *Group*] [**- S**] [**- n** *DirectoryC*] [**- o**] [**- s**] *File* [*Directory ...*]

Description

The **install** command installs a specified file in a specific place within a file system. It is most often used in makefiles. When replacing files, the **install** command copies (or moves) each file into the appropriate directory, thereby retaining the original owner and permissions based on the behavior of the **cp** and **mv** commands. An attempt is made to change the destination to owner **bin** and group **bin**. The **-O** *Owner* and

-G *Group* flags can be used to specify a different owner or group. The **install** command writes a message telling you exactly which files it is replacing or creating and where they are going.

You must be a super-user if you want to specify the ownership of the installed file with the **-O** or **-G** flags.

If you do not specify the *Directory* parameter, the **install** command searches a set of default directories (**/usr/bin**, **/etc**, and **/usr/lib**, in that order) for a file with the same name as the *File* parameter. The first time it finds one, it overwrites it with *File* and issues a message indicating that it has done so. If a match is not found, the **install** command issues a message telling you there was no match and exits with no further action. If the *File* parameter does not exist in the current directory, the **install** command displays an error message and exits with a nonzero value.

If any directories are specified on the command line, the **install** command searches them before it searches the default directories.

Flags

Item	Description
-c *DirectoryA*	Installs a new command file in the *DirectoryA* variable only if that file does not already exist there. If it finds a copy of *File* there, it issues a message and exits without overwriting the file. This flag can be used alone or with the **-s**, **-M**, **-O**, **-G**, or **-S** flag.
-f *DirectoryB*	Forces installation of *File* in *DirectoryB* whether or not *File* already exists. If the file being installed does not already exist, the command sets the permission code and owner of the new file to **755** and **bin**, respectively. This flag can be used alone or with the **-o**,**-s, -M, -O, -G,** or **-S** flag.
-G *Group*	Specifies a different group for the destination file. The default group is **bin**.
-i	Ignores the default directory list and searches only those directories specified on the command line. This flag cannot be used with the **-c, -f,** or **-m** flags.
-m	Moves the *File* parameter to the directory instead of being copied. Cannot be used with the **-c**, **-f**, **-i**, or **-n** flag.
-M *Mode*	Specifies the mode of the destination file.
-n *DirectoryC*	Installs the *File* parameter in the *DirectoryC* variable if it is not in any of the searched directories, and sets the permissions and owner of the file to **755** and **bin**, respectively. This flag cannot be used with the **-c**, **-f**, or **-m** flag.
-o	Saves the old copy of the *File* parameter by copying it into a file called **OLD***File* in the same directory. This flag cannot be used with the **-c** flag.
-O *Owner*	Specifies a different owner of the destination file. The default owner is **bin**.
-s	Suppresses the display of all but error messages.
-S	Causes the binary to be stripped after installation.

Examples

1. To replace a command that already exists in one of the default directories, enter:

   ```
   install fixit
   ```

   This replaces the **fixit** file if it is found in the **/usr/bin**, **/etc**, or **/usr/lib** directory. Otherwise, the **fixit** file is not installed. For example, if **/usr/bin/fixit** exists, then this file is replaced by a copy of the file **fixit** in the current directory.
2. To replace a command that already exists in a specified or default directory and to preserve the old version, enter:

```
install  -o fixit /etc /usr/games
```

This replaces the **fixit** file if it is found in the **/etc** or **/usr/games** directory or in one of the default directories. Otherwise the **fixit** file is not installed. If the file is replaced, the old version is preserved by renaming it **OLDfixit** in the directory in which it was found.

3. To replace a command that already exists in a specified directory, enter:

```
install  -i fixit /home/jim/bin /home/joan/bin /usr/games
```

This replaces the **fixit** file if it is found in the **/home/jim/bin**, **/home/joan/bin**, or **/usr/games** directory. Otherwise, the file is not installed.

4. To replace a command found in a default directory or install it in a specified directory if it is not found, enter:

```
install  -n /usr/bin fixit
```

This replaces the **fixit** file if it is found in one of the default directories. If the file is not found, it is installed as **/usr/bin/fixit**.

5. To install a new command, enter:

```
install  -c /usr/bin fixit
```

This creates a new command by installing a copy of the **fixit** file as **/usr/bin/fixit**, but only if this file does not already exist.

6. To install a command in a specified directory whether or not it already exists, enter:

```
install  -f /usr/bin  -o  -s fixit
```

This forces the **fixit** file to be installed as **/usr/bin/fixit** whether or not it already exists. The old version, if any, is preserved by moving it to **/usr/bin/OLDfixit** (a result of the **-o** flag). The messages that tell where the new command is installed are suppressed (a result of the **-s** flag).

Compatibility

For compatibility with Berkeley Software Distribution (BSD), two **install** commands exist. See the **installbsd** command.

Files

Item	Description
/usr/bin/install	Contains the **install** command.

install_all_updates Command

Purpose

Updates installed software to the latest level on media and verifies the current recommended maintenance or technology level.

Syntax

install_all_updates **-d** *Device* [**-p**] [**-i**] [**-c**] [**-r**] [**-n**] [**-s**] [**-x**] [**-v**] [**-N**] [**-S**] [**-Y**] [**-V**] [**-D**]

Description

install_all_updates examines currently installed software and attempts to update it to the latest level that is available on the media. **install_all_updates** will not install any filesets that are present on the media, but not installed on the system except in the following situations:

- the new filesets are installed as requisites of other filesets.
- the **/var/adm/ras/bosinst.data** filesets ALL_DEVICES_KERNELS to yes.

For installp images, all installp requisites are enforced.

Note:

1. Currently, **install_all_updates** processes installp images and rpm images. Because the **rpm** utility does not support automatic installation of requisites, some rpm software may not be installable with **install_all_updates**.
2. **install_all_updates** verifies the current recommended maintenance or technology level by using the "**oslevel**" utility and checking with the latest recommended maintenance or technology level known to this version of **install_all_updates**.
3. If **install_all_updates** locates an update to the install utilities (the **bos.rte.install** fileset), it first installs the update and then reinvokes itself to process the remaining updates. The "**-i**" flag can be used to update the install utilities only, this is useful when attempting to view an accurate preview.
4. **install_all_updates applies** all installp updates unless the **COMMIT** flag (**-c**) is specified. For more information of **APPLY** vs. **COMMIT** please see the **installp** man page.
5. **install_all_updates** will by default instruct **installp** to automatically install requisites and to do any necessary file system expansions. The "**-n**" will override the install requisite default, and "**-x**" will override the file system expansion default.
6. The following flags apply to installp updates *only*: **-c**, **-n**, **-x**, **-v**, **-S**, and **-V**.
7. Any library or executable program updated by an interim fix or service update which is in use by an active process will not be reflected in that process unless it is restarted. For example, an update that changes the ksh will not have the changes reflected in any ksh processes that are already running. Likewise, an update to the **libc.a** library will not be reflected in any process that is already running. In addition, any process that is using a library and does a **dlopen** operation of the same library after the library has been updated could experience inconsistencies if it is not restarted.
8. If an attempt is made to update a fileset that is locked by the interim fix manager (the **emgr** command), a notice will be displayed indicating which filesets are locked. The **lslpp** command shows that any locked filesets are in the EFIXLOCKED state.
9. If an attempt is made to update a file set that has an installed build date more recent than the build date of the selected fileset, a message will be displayed to indicate this.

Some installed software must ship new installation images instead of service updates in new technology levels or service packs of the AIX operating system. For instance, a new installation image is required if the requisites of the installation image changes. When filesets are updated by using the **smitty update_all** or **install_all_updates** command, the most current version of the fileset is installed irrespective of whether filesets are updated by using the installation image or service update in the software source.

When a new installation image is installed, the history of the fileset in the system, which is the output of the **lslpp -ah <fileset>** command, is reset. The output of the **lslpp -ah <fileset>** command lists the new level of the fileset instead of original installation that was installed and all changes after that installation. . The following examples show the history of the **bos.ecc_client.rte** file before and after the installation image is installed.

- Before a new installation image for the **bos.ecc_client.rte** file is shipped, the following output is displayed:

```
# lslpp -ah bos.ecc_client.rte
       Fileset         Level       Action       Status       Date       Time
       ----------------------------------------------------------------------------
       Path: /usr/lib/objrepos  bos.ecc_client.rte
                       6.1.9.0     COMMIT       COMPLETE     04/26/17     16:49:31
                       6.1.9.0     APPLY        COMPLETE     04/26/17     16:49:31
                       6.1.9.15    APPLY        COMPLETE     04/26/17     21:02:55
                       6.1.9.45    APPLY        COMPLETE     04/27/17     08:11:05
       Path: /etc/objrepos  bos.ecc_client.rte
                       6.1.9.0     COMMIT       COMPLETE     04/26/17     16:49:42
                       6.1.9.0     APPLY        COMPLETE     04/26/17     16:49:42
```

```
                        6.1.9.15    APPLY        COMPLETE     04/26/17     21:03:07
                        6.1.9.45    APPLY        COMPLETE     04/27/17     08:11:19
```

- After a new installation image for the **bos.ecc_client.rte** file is shipped and installed on the system, the following output is displayed:

```
# lslpp -ah bos.ecc_client.rte
       Fileset         Level        Action       Status       Date         Time
       ----------------------------------------------------------------------------
Path: /usr/lib/objreposbos.ecc_client.rte
                        6.1.9.100    COMMIT       COMPLETE     04/27/17     09:19:12
                        6.1.9.100    APPLY        COMPLETE     04/27/17     09:19:12

Path: /etc/objreposbos.ecc_client.rte
                        6.1.9.100    COMMIT       COMPLETE     04/27/17     09:19:22
                        6.1.9.100    APPLY        COMPLETE     04/27/17     09:19:22
```

Flags

Item	Description
-c	Instructs **installp** to commit all newly installed updates. Updates are applied by default (Please see the **installp** man page for more explanation on applying vs. committing updates).
-d *Device*	Specifies where the installation media can be found. This can be a hardware device such as tape or cdrom or it can be a directory that contains installation images. When installation media is a tape device it should be specified as no-rewind-on-close and no-retension-on-open.
-D	Turns on **install_all_updates** debug output. This flag is for debugging the **install_all_updates** utility and should not be used for normal operations.
-i	Update install utilities only.
-n	Instructs **installp** to not automatically install requisites. Automatic installation of requisites is the default behavior.
-N	Skip updating install utilities first. **Note:** This flag is not recommended unless you are debugging a related problem.
-p	Performs a preview of an action by running all preinstallation checks for the specified action. No software changes are made.
-r	Update rpm images (if possible). This flag is not set by default.
-s	Skip recommended maintenance or technology level verification. The verification is performed by default.
-S	Instructs **installp** to suppress multi-volume processing of cdrom media.
-v	Instructs **installp** to verify that all installed files in the fileset have the correct checksum value after the installation. This operation may require more time to complete the installation.
-V	Instructs **installp** to run in verbose output mode.
-x	Instructs **installp** to not automatically expand file systems. Automatic expansion of file systems is the default.
-Y	Agrees to all software license agreements which are required for software installation.

Exit Status

0

All **lppmgr** related operations completed successfully.

>0
An error occurred.

Security

Only the root user can execute **install_all_updates**.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To install all installp updates on device **/dev/cd0** and to verify the current recommended maintenance or technology level, enter:

```
install_all_updates -d /dev/cd0
```

2. To update any rpm images on your system, with newer technology levels from the **/images** directory, enter:

```
install_all_updates -d /images -r
```

3. To install the latest level of install utilities on device **/dev/cd0** (**bos.rte.install** update), enter:

```
install_all_updates -d /dev/cd0 -i
```

Files

Item	Description
/usr/sbin/install_all_updates	Contains the **install_all_updates** command.

install_assist Command

Purpose

Starts the Installation Assistant application.

Syntax

install_assist

Description

The **install_assist** command starts Installation Assistant, an application designed to simplify the customization of your system after a Base Operating System (BOS) installation. The Installation Assistant guides you through post-installation tasks and, in some cases, automatically installs software packages for you. The Installation Assistant has two interfaces, ASCII and graphical. The interface that displays is based on your terminal type (defined in the **TERM** environment variable).

If your terminal type is not set, the first menu displayed by the ASCII Installation Assistant requires you to enter your terminal type (tty). If you enter a terminal type that is not valid, this menu redisplays until a valid type is entered. If you enter a valid terminal type that does not match your terminal, the next screen displayed could be unreadable. In this case, press the break key sequence to return to the Set Terminal Type screen. For most terminal types, the break key sequence is Ctrl-C.

On a system with an ASCII interface, the newly installed BOS reboots and starts the Installation Assistant to guide you through completing configuration tasks. You must have root user authority to use the

Installation Assistant. To access the Installation Assistant later, type **install_assist** on the command line. You can also access it from a graphics system through the SMIT **smit assist** fast path. If there are outstanding software license agreements that must be accepted before you can continue to use the machine, the Installation Assistant prompts you to view and accept these agreements.

On a system with a graphical interface, the newly installed BOS reboots and the Configuration Assistant starts to guide you through the configuration tasks. If there are outstanding software license agreements that must be accepted before you can continue to use the machine, the Configuration Assistant prompts you to view and accept these agreements. To access the Configuration Assistant later, type **configassist** on the command line.

Most Installation Assistant tasks create or add to the **smit.log** and **smit.script** files in your home directory. (These are the same files appended when you run a SMIT session.) The commands built and run by the Installation Assistant tasks are added to the end of the **smit.log** file along with the command output. The time, name of the task, and the command (flags and parameters included) are added to the end of the **smit.script** file in a format that can easily be used to create executable shell scripts.

Example

1. To start the Installation Assistant, type:

   ```
   install_assist
   ```

2. To access the Configuration Assistant, type:

   ```
   configassist
   ```

3. Access the Installation Assistant from a graphical interface, use the SMIT `smit assist` fast path.

Files

Item	Description
smit.log	Specifies detailed information on your session, with time stamps.
smit.script	Specifies the task commands run during your session, with time stamps.

install_mh Command

Purpose

Sets up mailbox directories.

Syntax

install_mh [**-auto**] [**-help**]

Description

The **install_mh** command sets up mailbox directories. The **install_mh** command is not started by the user. The **install_mh** command is called by other programs only.

The **install_mh** command starts automatically the first time you run any Message Handler (MH) command. The **install_mh** command prompts you for the name of your mail directory. If the directory does not exist, the **install_mh** command queries you if it should be created. Upon receiving a positive response, the **install_mh** command creates the **$HOME/.mh_profile** file and places the `Path:` profile entry in it. This entry identifies the location of your mailbox by specifying the directory path for your MH directory, *UserMHDirectory*.

Flags

Item	Description
-auto	Creates the standard MH path without prompting.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.

Files

Item	Description
$HOME/.mh_profile	Contains the MH user profile.

installbsd Command

Purpose

Installs a command (BSD version of the **install** command).

Syntax

/usr/bin/installbsd [**-c**] [**-g** *Group*] [**-m** *Mode*] [**-o** *Owner*] [**-s**] *BinaryFileDestination*

Description

The **installbsd** command installs the file specified by the *BinaryFile* parameter by moving it to a file or directory specified by the *Destination* parameter. Use of the **-c** flag copies the *BinaryFile* rather than moving it. If the specified *Destination* parameter is a directory, the *BinaryFile* is moved into the directory. If the specified *Destination* parameter already exists as a file, the **installbsd** command removes that file before the *BinaryFile* is moved. The **installbsd** command does not move a file onto itself.

Installing the file /**dev**/**null** creates an empty file.

Flags

Item	Description
-c	Copies the file specified by the *BinaryFile* parameter to the file or directory specified by the *Destination* parameter.
-g *Group*	Specifies a group for the file specified by the *Destination* parameter. The default group is staff.
-m *Mode*	Specifies a mode for the file specified by the *Destination* parameter. The default mode is 755. The specified mode can be an octal number or a symbolic value.
-o *Owner*	Specifies the owner for the file specified by the *Destination* parameter. The default owner is the root user.
-s	Causes the file specified by the *BinaryFile* parameter to be stripped after installation.

Examples

To install a new command called **fixit**, enter:

```
installbsd  -c o mike fixit /usr/bin
```

This command sequence installs a new command by copying the program `fixit` to `/usr/bin/fixit`, with user `mike` as the owner.

Files

Item	Description
/usr/ucb/install	Hard-link to the **/usr/bin/installbsd** file.
/usr/bin/installbsd	Contains the **installbsd** command.

installios Command

Purpose

Sets up the environment and creates NIM resources from the Virtual I/O Server DVD to install the Virtual I/O logical partition and the Integrated Virtualization Manager.

Syntax

To set up the environment and create NIM resources for installing a Virtual I/O logical partition or Integrated Virtualization Manager:

`installios` [`-p` *partition_name* `-i` *ipaddrorhostname* `-S` *subnet_mask* `-g` *gateway* `-d` *path* `-s` *system_name* `-r` *profile* [`-n`] [`-P` *speed*] [`-D` *duplex*] [`-l` *language*] [`-L` *location*] [`-V` *vlan_tag*] [**-Y** *vlan_priority*]]

To clean up tasks from the setup process:

`installios -u` [`-f` | `-U`]

Description

The `installios` command creates NIM resources from the Virtual I/O Server DVD to install a Virtual I/O logical partition and Integrated Virtualization Manager. When invoked on a NIM client, the `-L` flag must be specified with the location of the bos.sysmgt.nim.master fileset. The `installios` command configures the client as a NIM master and creates the resources from the Virtual I/O Server DVD to install the ioserver logical partition or the Integrated Virtualization Manager. After the logical partition or Integrated Virtualization Manager have been installed, the `installios` command can return the NIM master back to its original state by removing the created resources from the DVD or by unconfiguring the NIM master. All of the flags are optional. If no flags are specified, the `installios` wizard runs and the user is prompted to interactively enter the flag information.

Flags

Item	Description
`-d` *path*	Specifies the path to the installation images (`/dev/cd0` or the path to a system backup of the Virtual I/O Server created by the `backupios` command. The path may also specify a remote NFS-mountable location such as *hostname*`:`*/path_to_backup*.

Item	Description
-D *duplex*	Specifies duplex (optional). This is the duplex setting with which to configure the network interface of a client. The network interface of the client must support the value of the *duplex* parameter. This value can be `full`, `half`, or `auto`. The default value is `full` if you do not specify this flag.
-f	Forces a cleanup to deallocate and remove resources which are not yet installed on a Virtual I/O logical partition or Integrated Virtualization Manager.
-g *gateway*	Specifies the client gateway (the default gateway that the client will use during network installation of the Virtual I/O Server operating system).
-i *ipaddrorhostname*	Specifies the client IP address or hostname (the IP address or hostname with which the network interface of a client will be configured for network installation of the Virtual I/O Server operating system).
-l *language*	Specifies the language (optional). This is the language in which the license agreement will be displayed before the installation. When the license is viewed, a prompt displays asking if the license is to be accepted. If the prompt is answered with y, then the installation will proceed and the Virtual I/O Server license is automatically accepted after installation. If the prompt is answered with n, the `installios` command exits and the installation does not proceed. If this flag is not specified, the installation proceeds, but the Virtual I/O Server will not be usable until the license is manually accepted after installation.
-L *location*	Specifies the location of the bos.sysmgt.nim.master fileset to configure a client to become a NIM master.
-n	Specifies that the network interface of a client should not be configured. If this flag is specified, the network interface of a client will not be configured with the IP settings that were specified in the flags given to the `installios` command after the installation has completed.
-p *partition_name*	Specifies the partition name. This is the name of the LPAR that will be installed with Virtual I/O Server operating system. This partition must be of type Virtual I/O Server and the partition name must match the name shown on the HMC; the name is not a host name.

Item	Description
-P *speed*	Specifies speed (optional). This is the communication speed to use when configuring the network interface of a client. The network interface of the client must support the value of the *speed* parameter. This value can be `10`, `100`, `1000`, or `auto`. The default value is `100` if you do not specify this flag.
-r *profile*	Specifies the profile name. This is the name of the profile that will contain the hardware resources that will be installed.
-s *system_name*	Specifies the managed system (the name of the managed system maintained by the HMC). This name must match the name shown on the HMC; the name is not a host name.
-S *subnet_mask*	Specifies the client subnet mask (the subnet mask with which the network interface of a client will be configured for network installation of the Virtual I/O Server operating system).
-u	Cleans up the environment to return the NIM master back to its original state.
-U	Unconfigures the NIM master.
-V *vlan_tag*	Specifies the virtual local area network (VLAN) tag identifier (0 to 4094) that is used for tagging Ethernet frames during the network installation for virtual network communication.
-Y *vlan_priority*	Specifies the virtual local area network (VLAN) tag priority (0 to 7) that is used for tagging Ethernet frames during the network installation for virtual network communication.

Exit Status

Item	Description
0	The **installios** command was successful.

Security

You must have root authority to run the **installios** command

Examples

1. To create Virtual I/O resources on a NIM master for installing client 9.3.6.234, type:

```
installios -d /dev/cd0 -i 9.3.6.234 -g 9.3.6.1 -S 255.255.255.0
```

2. To create Virtual I/O resources on a NIM client for installing client 9.3.6.234 where /tmp contains the bos.sysmgt.nim.master fileset, type:

```
installios -d /dev/cd0 -i 9.3.6.234 -g 9.3.6.1 -S 255.255.255.0 -L /tmp
```

3. To clean up tasks performed while creating Virtual I/O resources, type:

```
installios -u
```

4. To clean up tasks performed during the creation of Virtual I/O resources on a logical partition which has not yet been installed, type:

```
installios -u -f
```

5. To clean up tasks and unconfigure NIM after creating Virtual I/O resources , type:

```
installios -u -U
```

Location

/usr/sbin/installios

Files

Item	Description
/usr/sbin/installios	Contains the **installios** command
/etc/niminfo	Contains variables used by NIM

installp Command

Purpose

Installs available software products in a compatible installation package.

Syntax

To Install with Apply Only or with Apply and Commit

installp [**-R** *path*] [**-a** | **-a** **-c** [**-N**]] [**-e** *LogFile*] [**-V** *Number*] [**-d** *Device*] [**-E**] [**-Y**] [**-b**] [**-S**] [**-B**] [**-D**] [**-I**] [**-p**] [**-Q**] [**-q**] [**-v**] [**-X**] [**-F** | **-g**] [**-O** { [**r**] [**s**] [**u**] }] [**-t** *SaveDirectory*] [**-w**] [**-z** *BlockSize*] { *FilesetName* [*Level*]... | **-f** *ListFile* | **all** }

To Commit Applied Updates

installp [**-R** *path*] **-c** [**-e** *LogFile*] [**-V** *Number*] [**-b**] [**-g**] [**-p**] [**-v**] [**-X**] [**-O** { [**r**] [**s**] [**u**] }] [**-w**] { *FilesetName* [*Level*]... | **-f** *ListFile* | **all** }

To Reject Applied Updates

installp [**-R** *path*] **-r** [**-e** *LogFile*] [**-V** *Number*] [**-b**] [**-g**] [**-p**] [**-v**] [**-X**] [**-O** { [**r**] [**s**] [**u**] }] [**-w**] { *FilesetName* [*Level*]... | **-f** *ListFile* }

To Deinstall (Remove) Installed Software

installp [**-R** *path*] **-u** [**-e** *LogFile*] [**-V** *Number*] [**-b**] [**-g**] [**-p**] [**-v**] [**-X**] [**-O** { [**r**] [**s**] [**u**] }] [**-w**] { *FilesetName* [*Level*]... | **-f** *ListFile* }

To Clean Up a Failed Installation:

installp [**-R** *path*] **-C** [**-b**] [**-e** *LogFile*]

To List All Installable Software on Media

installp { **-l** | **-L** } [**-e** *LogFile*] [**-d** *Device*] [**-B**] [**-I**] [**-q**] [**-E**] [**-z** *BlockSize*] [**-O** { [**s**] [**u**] }]

To List All Customer-Reported Problems Fixed with Software or Display All Supplemental Information

installp { **-A**| **-i** } [**-e** *LogFile*] [**-d** *Device*] [**-B**] [**-I**] [**-q**] [**-z** *BlockSize*] [**-O** { [**s**] [**u**] }] { *FilesetName* [*Level*]... | **-f** *ListFile* | **all** }

To List Installed Updates That Are Applied But Not Committed

installp -s [**-e** *LogFile*] [**-O** { [**r**] [**s**] [**u**] }] [**-w**] { *FilesetName* [*Level*]... | **-f** *ListFile* | **all** }

To List Platform Specific Installable Software on Media

installp { **-l** | **-L** } { **-M** *Platform* } [**-e** *LogFile*] [**-d** *Device*] [**-B**] [**-I**] [**-q**] [**-z** *BlockSize*] [**-O** { [**s**] [**u**] }]

Description

Notes:

1. The **noclobber** option of the Korn or C shell must be unset in the environment from which an installation is performed.
2. Update all can be accomplished with smitty or with `install_all_updates`.

The **installp** command installs and updates software.

A fileset is the lowest installable base unit. For example, **bos.net.tcp.client 4.1.0.0** is a fileset. A fileset update is an image with a different modification level or a different fix level. For example, **bos.net.tcp.client 4.1.0.2** and **bos.net.tcp.client 4.1.1.0** are both fileset updates for **bos.net.tcp.client 4.1.0.0**.

When a base level (fileset) is installed on the system, it is automatically committed. You can remove a fileset regardless of the state (such as committed, broken, committed with applied updates, and committed with committed updates).

When a fileset update is applied to the system, the update is installed. The current version of that software, during installation, is saved in a special save directory on the disk so that later you can return to that version if desired. After a new version of a software product has been applied to the system, that version becomes the currently active version of the software.

Updates that have been applied to the system can be either committed or *rejected* at a later time. The **installp -s** command can be used to get a list of applied updates that can be committed or rejected.

When updates are committed with the **-c** flag, the user is making a commitment to that version of the software product. The saved files from all previous versions of the software product are removed from the system, making it impossible to return to a previous version of the software product. Software can be committed at the time of installation by using the **-ac** flags.

Notes:

- Committing the already applied updates does not change the currently active version of a software product. It merely removes saved files for previous versions of the software product.
- The signature verification is performed during the package installation. The verification is based on the level of digital signature verification policy.

When a base level is removed with the **-u** flag, the files that are part of the software product and all its updates are removed from the system. Most cleanup of system configuration information pertaining to the product is also done, but this is dependent on the product and may not always be complete.

When a software product update is rejected with the **-r** flag, the current version of the software product is changed to the immediate previous version of the update. Files saved for the rejected update and any updates that were applied after it are removed from the system.

A software product that is to be removed from the system can be in any state. Product updates can be in either the applied or committed state, and they will also be removed.

If a previously interrupted installation leaves any software in a state of either applying or committing, it is necessary to perform cleanup with the **-C** flag before any further installations are allowed. Although the **installp -C** command accepts software product names on the command line without returning an error, an attempt is always made to clean up all products when the **-C** flag is used. An attempt is made to clean

up any incomplete installations by removing those parts that were previously completed. An attempt is also made to return to the previous version of the software product, if one exists, as the currently active version. If this cannot be done, the software product is marked as *broken*, and unpredictable results can occur if the user attempts to use it. Therefore, it is advisable for the user to reinstall any broken software products or updates.

The **-t** flag specifies an alternate location for a save directory that holds files being replaced by an update. This option is primarily useful in the following two circumstances.

- You have enough local disk space for saving replaced files but you do not want to permanently expand the root and **/usr** file systems.

  In this case, you can choose to create a separate file system for the alternate save directory. When you are satisfied with the updated system and have committed all applied updates, disk space can be retrieved by deleting the save file system.
- You do not have enough local disk space for saving replaced files but you have access to ample disk space on a remote system. In this case, you can specify a directory that is mounted from a remote file system.

  If a remote file system is used, commit the updates *as soon as possible*. You may want to initiate the installation action as an `apply` and `commit` operation with the **-ac** flags. If you want to `apply` only to be able to reject unwanted updates, then test the newly installed updates *as soon as possible* and then commit or reject them.

  Take into account the following considerations when using an alternate save directory:
- It is recommended that you use the same alternate save location on each invocation of the **installp** command.
- If an alternate save directory is used for an apply operation, make sure that the file system containing that directory remains mounted. It is highly recommended that any necessary mounts be done automatically on a reboot.
- If an alternate save directory is missing on a commit operation, the commit takes place, a warning is given stating that the save directory could not be deleted. In this case, you must delete the save directories that are no longer used in order to retrieve that disk space.
- If an alternate save directory is missing on reject, the reject operation cannot be done because the saved files are missing. An error is given, and the entire reject operation is canceled. If the missing save directory is not caused by a temporary situation (for example, the inability to contact a remote directory on the network) your only options are to commit the updates or leave them in an applied state permanently.
- When doing a system backup, ensure that you back up any alternate save directories that do not reside in the root volume group.
- The installation process safeguards users with a remote save directory from the possibility of two different systems using the same remote directory. However, use directory path names that easily and uniquely identify each user's system. For example, you can add the system's host name in the path name.
- Do not create a **mksysb** backup of a system with a remote save directory and then try to restore the **mksysb** image onto a system other than the original. In this case, using a **mksysb** image to install several like systems causes multiple ownership of the same remote save directory.

The **installp -A** command can be used to obtain a list of the Authorized Program Analysis Report (APAR) numbers and summaries for all customer-reported problems that are fixed in the specified software package. The **installp -i** command can be used to display supplemental information contained in files that can be a part of the specified software package.

To list all the software products and updates on the specified installation media, use the **installp -l** command. The output of the **installp** command with the **-l** flag resembles the following:

```
# Fileset Name             Level                   I/U Q Content
#=============================================================
  X11.adt.include          4.1.0.0                 I  N usr
#   AIX windows Application Development Toolkit Include F
```

```
  X11.adt.lib              4.1.0.0                  I  N usr
#   AIX windows Application Development Toolkit Libraries
#
  X11.adt.motif            4.1.0.0                  I  N usr
#    AIX windows Application Development Toolkit Motif

#
  X11.adt.bitmaps          4.1.0.0                  I  N usr
#    AIX windows Application Development Toolkit Bitmap Fi

#
  X11.adt.ext              4.1.0.0                  I  N usr
#    AIX windows Application Development Toolkit for X Ext
#
  X11.adt.imake            4.1.0.0                  I  N usr
#    AIX windows Application Development Toolkit imake
#
  X11.apps.rte             4.1.0.0                  I  N usr
#    AIX windows Runtime Configuration Applications
#
  X11.apps.msmit           4.1.0.0                  I  N usr
#    AIX windows msmit Application
```

The field descriptions are as follows:

Item	Description
`Fileset Name`	Name of the fileset to be installed.
`Level`	Level of the fileset to be installed.
`I/U`	The type of package to which the fileset belongs. The fileset can belong to an installation package or to one of several types of update packages. The package types are as follows: **I** Indicates an installation package. **S** Indicates a single update. **SR** Indicates a required update. Whenever the **installp** command encounters a required update, the update is automatically included in the input list. **SF** Indicates a required update. Whenever the **installp** command encounters a required update, the update is automatically included in the input list. Reserved for updates to the **installp** fileset. **M** Indicates a maintenance or technology package. This is a packaging update that contains only a list of other updates to be applied. This package delivers no files. **ML** Indicates an update package that identifies a new maintenance or technology level for the product. This is a cumulative set of all updates since the previous product level.
`Q`	Quiescent (quiet) column. A **Y** indicates that running processes can be affected by the installation of this fileset. Refer to the documentation supplied with the software product. An **N** indicates that running processes are not affected by the installation of this fileset. A **B** indicates **bosboot** and quiescent. A **b** indicates **bosboot** and not quiescent.

Item	Description
Content	Content column: **usr,root** **/usr** and root file systems (AIX 3.2 and later) **usr** **/usr** file system only (AIX 3.2 and later) **share** **/usr/share** file system only (AIX 3.2 and later)

Output from the **installp -s** command, which is used to get a list of applied software fileset updates and updates that are available to be either committed or rejected, resembles the following:

```
Installp Status
---------------
Name                        Part    Level                     State
-------------------------------------------------------------------
bos.net.tcp.client          USR     4.1.0.2                   APPLIED
bos.net.tcp.client          ROOT    4.1.0.2                   APPLIED
bos.rte.commands            USR     4.1.0.1                   APPLIED
bos.rte.misc_cmds           USR     4.1.0.1                   APPLIED
bos.rte.tty                 USR     4.1.0.1                   APPLIED
```

The field descriptions are as follows:

Item	Description
Name	Name of the installed software product fileset.
Part	The part of the fileset where:
ROOT	root file system
SHARE	**/usr/share** file system
USR	**/usr** file system.
Level	The level of the installed software product option.
State	The state of the installed software product option.

The software products and updates to be installed can be identified in one of following ways:

- by the keyword **all**, which indicates that all software contained on the specified installation media is to be installed
- by a list of software product names (each of which can optionally be followed by a level) that indicates the software to be installed
- by the **-f** flag followed by a file name, where each line in the file is an entry containing a software product name, optionally followed by a level, or is a comment line that begins with a # and is ignored

  Note: The **installp** program uses the **sysck** command to verify files after restoring them. The **sysck** command does not recognize the following special characters in file names: ~, ` , ', \, ", $, ^, &, ( ), |, {}, [], <>, and ?. If a file name contains any of these characters, installation fails.

The *FilesetName* parameter can be used to specify an entire software product or any separately installable filesets within the software package. For example, **bos.net** is the name of a software package, and the separately installable filesets within that software package are **bos.net.ncs.client**, **bos.net.nfs.client**, and **bos.net.tcp.client**. If the user specifies **bos.net** for the *FilesetName* parameter, then all of the separately installable filesets listed are installed. If the user specifies **bos.net.tcp.client** for the *FilesetName* parameter, then only that fileset is installed.

The *Level* parameter indicates the level of the software product or update that is to be installed. The *Level* parameter is of the form *vv.rr.mmmm.ffff* where:

Item	Description
vv	is a numeric field of 1 to 2 digits that represents the version number.
rr	is a numeric field of 1 to 2 digits that represents the release number.
mmmm	is a numeric field of 1 to 4 digits that represents the modification level. Modification level is also called maintenance level or technology level.
ffff	is a numeric field of 1 to 4 digits that represents the fix level.

If a user is installing an installation package from installation media that contains only installation packages it is not necessary to specify the level. More than one software product installation package with different levels does not often exist on the same installation medium. However, when this does occur **installp** installs the specified software product at the latest software product level when *Level* is not specified with *FilesetName*. For installation media that contain either update packages only or contain both installation and update packages, all applicable update packages that are present on the installation media for the specified *FilesetName* are also installed when *Level* is not specified. For installation media that contain both installation and update packages the user can request the installation of only installation packages or only update packages by specifying the **-I** or **-B** flags, respectively. If the user wants to install only some of the updates on the installation medium for a specific software product both *FilesetName* and *Level* for each of the updates to be installed for that software product must be specified.

You can use the following example to install TCP/IP and one of its updates that are both contained in the **/usr/sys/inst.images** directory.

```
installp -a -d/usr/sys/inst.images bos.net.tcp.client 4.1.0.0
bos.net.tcp.client 4.1.0.2
```

Note: If there are duplicate filesets at the same level, **installp** uses the first one that it finds in the install table of contents (**.toc**). This situation can occur when **bffcreate** is used to extract images from different media to the same installation directory. For this reason, make sure that update images are not extracted to the same directory as base level images for the same fileset at the same level.

A summary report is given at the end of the **installp** output that lists the status of each of the software products that were to be installed. An example summary report for the previous **installp** command is as follows:

```
Installp Summary
----------------
Name                    Level        Part        Event        Result
-------------------------------------------------------------------------
bos.net.tcp.client      4.1.0.0      USR        APPLY        SUCCESS
bos.net.tcp.client      4.1.0.0      ROOT       APPLY        SUCCESS
bos.net.tcp.client      4.1.0.2      USR        APPLY        SUCCESS
```

Note:

1. If a previously installed level of a fileset update is in the broken state, the **-acgN** flags must be used when that fileset update is installed again.
2. The **installp** command cannot install a mkinstallp package or bff image that is larger than 2 GB in size. An alternative is to break the bff image into multiple packages that are less than 2 GB in size.
3. If an attempt is made to update a fileset that is locked by the interim fix manager (the **emgr** command), a notice is displayed indicating the filesets that are locked. The **lslpp** command shows that any locked filesets are in the EFIXLOCKED state.
4. If an attempt is made to update a fileset that has an installed build date more recent than the build date of the selected fileset, a message is displayed to indicate this.

Summary Report Values

The summary report identifies the name of the product option and the part of the product. Other information given includes the requested action (event) and the result of that action.

Event Values

The Event column of the summary report identifies the action that has been requested of the **installp** command. The following values are displayed in this column:

Event	Definition
APPLY	An attempt was made to apply the specified fileset.
COMMIT	An attempt was made to commit the specified fileset update.
REJECT	An attempt was made to reject the specified fileset update.
CLEANUP	An attempt was made to perform cleanup for the specified fileset.
DEINSTALL	An attempt was made to remove the specified fileset.

Result Values

The Result column of the summary report gives the result of **installp** performing the requested action. It can have the following values:

Result	Definition
SUCCESS	The specified action succeeded.
FAILED	The specified action failed.
CANCELLED	Although preinstallation checking passed for the specified option, it was necessary to cancel the specified action before it was begun. Interrupting the installation process with Ctrl+c can sometimes cause a canceled action, although, in general, a Ctrl+c interrupt causes unpredictable results.

Flags

Item	Description
-A	Displays the APAR number and summary of all customer-reported problems that are fixed in the specified software package. No installation is attempted.
-a	Applies one or more software products or updates. This is the default action. This flag can be used with the **-c** flag to apply and commit a software product update when installed.
-b	Prevents the system from performing a bosboot in the event that one is needed.
-B	Indicates that the requested action should be limited to software updates.
-C	Cleans up after an interrupted installation and attempts to remove all incomplete pieces of the previous installation. Cleanup must be performed whenever any software product or update is in a state of either *applying* or *committing* and can be run manually as needed. For backward compatibility, other flags and parameters can be accepted with **installp -C**, but are ignored because all necessary cleanup is attempted.

Item	Description
-c	Commits all specified updates that are currently applied but not committed. When an update is committed all other software products it is dependent on must also be committed (unless they are already in the committed state). The specified software product is dependent on any software product that is a prerequisite or corequisite of the specified product. If the requisite software products are not in the committed state, the commit fails and error messages are displayed. The **-g** flag can be used to automatically commit requisite software product updates.
-D	Deletes the installation image file after the software product or update has been successfully installed. When the **-g** flag is specified, the installation image files for any products that are automatically included are also deleted. This flag is valid only with the **-a** or **-ac** flags and is not valid with the **-Or** flag. This flag is also only valid when the device is a directory and an installation image file on the system where the installation is taking place.
-d *Device*	Specifies where the installation media can be found. This can be a hardware device such as tape or diskette, it can be a directory that contains installation images, or it can be the installation image file itself. When the installation media is a product tape or Corrective Service tape, specified the tape device as no-rewind-on-close and no-retension-on-open. Examples of this would be **/dev/rmt0.1** for a high density tape, or **/dev/rmt0.5** for a low density tape. Use the options specified by the tape supplier. The default device is **/dev/rfd0**.
-e *LogFile*	Enables event logging. The **-e** flag enables the user to append certain parts of the **installp** command output to the file specified by the *LogFile* variable. By default the output of the **installp** command goes to **stdout** and **stderr**, unless SMIT or VSM is used, in which case the output goes to the **smit.log**. The *LogFile* variable must specify an existing, writable file, and the file system in which the file resides must have enough space to store the log. The log file does not wrap. Not all output is appended. Copyright information is still displayed to the user. Error messages are displayed on the screen and are sent to the file specified by the *LogFile* variable. A results summary of the **installp** command invocation is also displayed on the screen and sent to the *LogFile*. This flag is primarily used by NIM and BOS install to limit the output shown to the user, but keep useful information for later retrieval.
-E	Displays software license agreements. This flag is only valid with the **-a** or **-l** flags. If the **-E** flag is specified with the **-a** flag, a new section is displayed showing the pending license agreements associated with the selected filesets. If the **-E** flag is specified with the **-l** flag, output is displayed showing the license agreements associated with all filesets on the media.
-F	This option can be used to force the installation of a software product even if there exists a previously installed version of the software product that is the same as or newer than the version currently being installed. The **-F** flag is not valid with update packages or the **-g** flag. When you use the **-F** flag, the **-I** flag is implicit.
-f *ListFile*	Reads the names of the software products from *ListFile*. If *ListFile* is a - (dash), it reads the list of names from the standard input. Software fileset names, optionally followed by a level, should be one per line of text, and any text following the second set of white spaces or tabs on a line is ignored. Output from the **installp -l** command is suitable for input to this flag.

Item	Description
-g	When used to install or commit, this flag automatically installs or commits, respectively, any software products or updates that are requisites of the specified software product. When used to remove or reject software, this flag automatically removes or rejects dependents of the specified software. The **-g** flag is not valid when used with the **-F** flag. **Note:** This flag also automatically pulls in a superseding update present on the media if the specified update is not present. This flag causes the newest update to be installed for a given fileset, when there are multiple superseding updates for the same fileset on the installation media.
-I	(uppercase i) Indicates that the requested action must be limited to base level filesets.
-i	Displays on standard output the **lpp.instr**, **lpp.doc**, **lpp.README**, and **README** files on the installation media for the software product, if they exist. This flag can take a significant amount of time for a large number of filesets.
-J	This flag is used when the **installp** command is executed from the System Management Interface Tool (SMIT) menus.
-l	(lowercase L) Lists all the software products and their separately installable options contained on the installation media to standard output. No installation occurs. The **-l** flag is not valid with the **-Or** flag.
-L	Displays the contents of the media by looking at the table of contents (TOC) and displaying the information in colon-separated output. This flag is used by smit and vsm to list content of the media. The format provided: **package:fileset:v.r.m.f:PTF:type:state:supersede:\ sup_ptf:sup_state:latest_sup:quiesce:Descr:\ netls_vendor_id:netls_prod_id:netls_prod_ver:relocatable:build date**
-M*Platform*	Specifies the *Platform* value. Any of the following values can be used to list the installable software packages: **R** Specifies POWER processor-based platform packages only. **N** Specifies neutral packages, that is, packages that are not restricted to the POWER processor-based platform. **A** Specifies all packages.
-N	Overrides saving of existing files that are replaced when installing or updating. This flag is valid only with the **-ac** flags. If there is a failure in the system during the installation, there is no recovery of replaced files when this flag is used.
-O{[**r**][**s**][**u**]}	Installs the specified part of the software product. The **r** indicates the **/** (root) part is to be installed, the **s** indicates the **/usr/share** part is to be installed, and the **u** indicates the **/usr** part is to be installed. The **-O** flag is not needed with standard systems because without this flag all parts are installed by default. This flag is needed for use with the installation of diskless or dataless workstations and is designed for use by the **nim** command. The **-Or** option is not valid with the **-d** or **-l** flags.

Item	Description
-p	Performs a preview of an action by running all preinstallation checks for the specified action. This flag is only valid with apply, commit, reject, and remove (**-a**, **-c**, **-r**, and **-u**) flags.
-Q	Suppresses errors and warnings concerning products failing to install due to instrequisites.
-q	Specifies quiet mode, which suppresses the prompt for the device, except for media volume change.
-r	Rejects all specified software updates that are currently applied but not committed. When a software update is rejected any other software product that is dependent on it (that is, those software products that have the specified software product as a requisite) must also be rejected. The **-g** flag can be used to reject automatically dependent software updates. The keyword **all** is not valid with the reject flag (**-r**).
-R *path*	Indicates a user-specified installation location.
-s	Lists information about all software products and updates that have been applied but not committed. This list comprises the software that is available to be either committed or rejected.
-S	Suppresses multiple volume processing when the installation device is a CD-ROM. Installation from a CD_ROM is always treated as a single volume, even if the CD-ROM contains information for a multiple volume CD set. This same suppression of multiple volume processing is performed if the **INU_SINGLE_CD** environment is set.
-t *SaveDirectory*	Specifies an alternate save directory location for files being replaced by an update. The **-t** flag is only valid with an apply or an apply/commit operation for updates. This flag is not valid with the **-N** flag. The **-t** flag is useful when there is insufficient space in the default file systems (**/** and **/usr**) or when it is undesirable to permanently expand these file systems. It may be desirable for the specified directory to be a remote file system. A remote file system must have ample space, because the **installp** command cannot expand remote file systems.
-u	Removes the specified software product and any of its installed updates from the system. The product can be in either the committed or broken state. Any software products that are dependent on the specified product must also be explicitly included in the input list unless the **-g** flag is also specified. Removal of any **bos.rte** fileset is never permitted.
-v	Verifies that all installed files in the fileset have the correct checksum value after the installation. Installed files are always verified for correct file size after installation. Use this flag after network or remote device installations. If any errors are reported, it might be necessary to install the software product again. Post-installation requisite consistency checks are also started by this flag.

Item	Description
-V *Number*	Specifies the verbose option that provides four levels of detail for preinstallation output. The valid values for the *Number* parameter are 2, 3, or 4. The default level of verbosity, without the use of the -V flag, prints an alphabetically ordered list of FAILURES, WARNINGS, and SUCCESSES from preinstallation processing. Requisite failures are reported with emphasis on the real cause of the failure. Extraneous requisites for failed filesets are not displayed. The preinstallation output is modified by levels 2 through 4 as described below: **2** Prints alphabetically ordered list of FAILURES and WARNINGS. Requisite failures are displayed with additional information describing requisite relationships between selected filesets and the requisites causing them to fail. Failing requisites suppressed under *Level 1* are displayed. Preinstallation SUCCESSES are displayed in the order in which they are processed. **3** Level 3 is the same as Level 2, with the exception that additional requisite information is displayed for SUCCESSES. **4** Level 4 is the same as Level 3 for SUCCESSES and WARNINGS. Requisite failures are displayed in a format depicting detailed requisite relationships. **Note:** If verbosity level 2 or higher is used, the files that are restored on to the system is shown in the output. Because this will make **installp**'s output much more verbose, make sure that your **/** (root) file system does not become full when the /smit.log becomes large (if using smit to run installp).
-w	Does not wildcard *FilesetName*. Use this flag from smit so it only installs the fileset chosen and not the filesets that match. For example, if you choose `foo.rte`, `foo.rte.bar` is not automatically pulled in, as it would be by default, without the **-w** flag.
-X	Attempts to expand any file systems where there is insufficient space to do the installation. This option expands file systems based on current available space and size estimates that are provided by the software product package. Note that it is possible to exhaust available disk space during an installation even if the **-X** flag is specified, especially if other files are being created or expanded in the same file systems during an installation. Also note that any remote file systems cannot be expanded.
-Y	Agrees to required software license agreements for software to be installed. This flag is only valid with the **-a** flag.
-z *BlockSize*	Indicates in bytes the block size of the installation media. The default value of Size is 512.
FilesetName	This is the name of the software product to be installed and can specify either an entire software product or any separately installable filesets within the software product. This can be used to specify the name of a fileset or fileset update.
Level	This indicates the level of the software product or update that is to be installed and is of the form vv.rr.mmmm.ffff. If a fileset update has an additional fix ID (also know as ptf id), that ID must also be specified in the Level as in vv.rr.mmmm.ffff.ppppppp.

Exit Status

Item	Description
0 (zero)	Indicates that all attempted installations were successful, or that no processing was required for the requested action on the requested filesets (for example, if a requested fileset was already installed).
nonzero	Indicates that some part of the installation was not successful.

A summary report is given at the end of the **installp** output that lists the status of each of the software products that were to be installed. For those software products that could not be installed or whose installation failed, the user can search for the cause in the more detailed information that is continually displayed from the **installp** command during the installation process.

Security

Privilege Control: Only the root user can run this command.

Note:

If the Trusted Execution (TE) policy is turned on along with the TSD_LOCK policy or the TSD_FILE_LOCK policy, the **installp** command fails. To continue with the installation, manually turn off the TSD_LOCK policy or the TSD_FILE_LOCK policy. The **installp** command runs successfully with TE policies if the TSD_LOCK policy or the TSD_FILE_LOCK policy is not turned on.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list all software products and installable options contained on an installation cartridge tape, type:

   ```
   installp -L -d /dev/rmt0.1
   ```

2. To list all customer-reported problems fixed by all software products on an installation tape, type:

   ```
   installp -A -d /dev/rmt0.1 all
   ```

3. To install (automatically committed) all filesets within the **bos.net** software package (located in the **/usr/sys/inst.images** directory) and expand file systems if necessary, type:

   ```
   installp -aX -d/usr/sys/inst.images bos.net
   ```

4. To reinstall and commit the NFS software product option that is already installed on the system at the same level (from tape), type:

   ```
   installp -acF -d/dev/rmt0.1 bos.net.nfs.client 4.1.0.0
   ```

5. To install (apply only) certain updates that are contained on diskette for the TCP/IP software product, type:

   ```
   installp -a bos.net.tcp.client 4.1.0.2 bos.net.tcp.server 4.1.0.1
   ```

6. To remove a fileset named bos.net.tcp.server, type:

   ```
   installp -u bos.net.tcp.server
   ```

7. To specify an alternate storage directory on a remote file system for a BOSNET TCP/IP update with **-t/temp_space**, see the following example: the save directory becomes **/temp_space/***My_Hostname***/usr/lpp/bos.net/bos.net.nfs.client/4.1.1.0.save**.

```
mount Server_Name:/Save_Area /temp_space

installp -a -t /temp_space/My_Hostname  \
bosnet.nfs.client 4.1.1.0
```

8. To capture a log file of all output from the **installp** command, use the **script** command as in the following example. Output is written to the **typescript** file in the current directory.

   ```
   script
   installp ...
   <Ctrl>d
   ```

   or

   ```
   installp ... 2>&1 | tee /tmp/inst.out
   ```

   In the second example, output is written to the screen and a copy is saved.

9. To preview the installation of the **bos.net.tcp.client** fileset from the CD using the **installp** command, type:

   ```
    installp -pacgXd /dev/cd0 bos.net.tcp.client
   ```

10. To install TCP/IP and one of its updates that are both contained in the **/usr/sys/inst.images**, use the **installp** command as in the following example.

    A summary report is given at the end of the **installp** command output that lists the status of each of the software products that were to be installed. An example summary report for the previous **installp** command is as follows:

    ```
    Installp Summary
    ----------------
    Name                 Level        Part       Event        Result
    -------------------------------------------------------------------
    bos.net.tcp.client   4.1.0.0      USR        APPLY        SUCCESS
    bos.net.tcp.client   4.1.0.0      ROOT       APPLY        SUCCESS
    bos.net.tcp.client   4.1.0.2      USR        APPLY        SUCCESS
    ```

    Note: This summary is also saved in **/var/adm/sw/installp.summary** until the next **installp** invocation. The header file **inuerr.h** in the **/usr/include** directory describes the fields making up the records in the **installp.summary** file.

11. To list software products (located in the **/usr/sys/inst.images** directory) that are installable on POWER processor-based machines, type:

    ```
    installp -l -MR -d /usr/sys/inst.images
    ```

12. To update all filesets from a CD that are currently installed on the system, type:

    ```
    lslpp -lc | awk -F ":" '{print $2}' | tail -n +2 > /tmp/lslpp
    installp -agXd /dev/cd0 -e /tmp/install.log -f /tmp/lslpp
    ```

    where the **-e** logs the output to the **/tmp/install.log** file.

Files

Item	Description
/dev/rfd0	Specifies the default restore device.
/dev/rmt*n*	Specifies the raw streaming tape interface.
/usr/sys/inst.images directory	Contains files in backup format for use in installing or updating a complete set or subset of software products.

instfix Command

Purpose

Installs filesets associated with keywords or fixes.

Syntax

instfix [**-T** [**-M** *Platform*]] [**-s** *String*] [**-S**] [**-k** *Keyword* | **-f** *File*] [**-p**] [**-d** *Device*] [**-i** [**-c**] [**-q**] [**-t** *Type*] [**-v**] [**-F**]] [**-a**] [**-R**]

Description

The **instfix** command allows you to install a fix or set of fixes without knowing any information other than the Authorized Program Analysis Report (APAR) number or other unique keywords that identify the fix.

Any fix can have a single fileset or multiple filesets that comprise that fix. Fix information is organized in the Table of Contents (TOC) on the installation media. After a fix is installed, fix information is kept on the system in a fix database.

The **instfix** command can also be used to determine if a fix is installed on your system.

Notes:

- Return codes for the **instfix** command are documented in the `/usr/include/inuerr.h` file, which is shipped with the `bos.adt.include` fileset. There is also a general failure code of 1 and a single reference to EACCES (13) from the `/usr/include/errno.h` file.
- Listing interim fix information is possible only when using the **-f**, **-i**, **-k**, **-q**, **-r**, **-t**, and **-v** flags. You cannot install the interim fixes using the **instfix** command.

Flags

Item	Description
-a	Displays the symptom text associated with a fix. Can be combined with the **-i**, **-k**, or **-f** flag.
-c	Displays colon-separated output for use with **-i** flag. Output includes keyword name, fileset name, required level, installed level, status, and abstract. To display filesets that are not installed, the `-v` flag must also be used. Status values are: **-** Down level **=** Correct level **+** Superseded **!** Not installed
-d *Device*	Specifies the input device. Not valid with the **-i** and **-a** flags.
-F	Returns failure unless all filesets associated with the fix are installed.
-f *File*	Specifies the input file containing keywords or fixes. Use **-** (dash) for standard input. The **-T** flag produces a suitable input file format for **-f**.
-i	Displays whether fixes or keywords are installed. Use this flag with either the **-k** or the **-f** flag. Installation is not attempted when the **-i** flag is used. If you do not specify the **-k** or the **-f** flag, all known fixes are displayed.

Item	Description
-k *Keyword*	Specifies an APAR number or keyword to be installed. Multiple keywords can be entered. A list of keywords entered with the **-k** flag must be contained in quotation marks and separated with spaces.
-M *Platform*	Specifies that any of the *Platform* values might be used to list the fixes for that particular platform. **R** Specifies POWER processor-based platform fixes only. **N** Specifies neutral fixes, that is, fixes that are not restricted to the POWER processor-based platform. **A** Specifies all fixes.
-p	Displays file sets associated with keywords. This flag is used with either the **-k** or the **-f** flag. Installation is not attempted when the **-p** flag is used.
-q	Specifies quiet mode. Use this flag with the **-i** flag. If you use the **-c** flag, no heading is displayed, otherwise there is no output.
-R	Specifies the User Specified Installation Location (USIL).
-s *String*	Searches for and displays fixes on media containing a specified string.
-S	Suppresses multiple volume processing when the installation device is a CD-ROM. Installation from a CD_ROM is always treated as a single volume, even if the CD-ROM contains information for a multiple volume CD set. This same suppression of multiple volume processing is performed if the **INU_SINGLE_CD** environment is set.
-T	Displays the entire list of fixes present on the media.
-t*Type*	Limits the search operation to a given type when used with the **-i** flag. Valid types are: **f** fix **p** preventive maintenance **i** interim fix
-v	Specifies verbose mode when used with the **-i** flag. Displays information about each fileset associated with a fix or keyword. Use this flag with the `-i` flag to display filesets that are not installed. An uninstalled fileset is indicated by an `!` (exclamation point).

Security

Privilege Control: You must be the root user to install using the **instfix** command, but any user can run the **instfix** command to query the fix database.

Examples

1. To install all filesets associated with fix `IX38794` from the tape mounted on `/dev/rmt0.1`, type:

   ```
   instfix  -k IX38794  -d /dev/rmt0.1
   ```

2. To install all fixes on the media in the tape drive, type:

   ```
   instfix  -T  -d /dev/rmt0.1 | instfix  -d /dev/rmt0.1  -f-
   ```

The first part of this command lists the fixes on the media, and the second part of this command uses the list as input.

3. To list all keyword entries on the tape containing the string SCSI, type:

```
instfix  -s SCSI  -d /dev/rmt0.1
```

4. To inform the user on whether fixes IX38794 and IX48523 are installed, type:

```
instfix  -i  -k "IX38794 IX48523"
```

5. To create a list of filesets associated with fix IX12345 for bffs in the /bffs directory, type:

```
instfix  -p  -k IX12345  -d /bffs | installp -acgX -f- -d /bffs
```

This sequence passes the list of fixes to the **installp** command to be applied and committed. The **installp** command extends filesystems as needed with the flags shown. This example shows that you can select other **installp** flags. The **instfix** command calls **installp** if the **-p** flag is not used.

6. To list all of the fixes that are not restricted to the POWER processor-based platform, type:

```
instfix  -T  -MN  -d /dev/cd0
```

Files

Item	Description
/usr/sbin/instfix	Contains the **instfix** command.
/usr/lib/objrepos/fix	Specifies the path to the Object Data Manager database.

inucp Command

Purpose

Performs simple copy operations for the **installp** command. This command is used by the **installp** command and the install scripts.

Syntax

inucp **-s** *StartDirectory* [**-e** *FinalDirectory*] *ListFile* *ProductName*

Description

The **inucp** command copies the files in a file tree with its root at *StartDirectory* to the appropriate place on the *FinalDirectory* root.

Before replacing files that may already exist in the *FinalDirectory* file tree, the **inusave** command should be called to save the files until needed by the **inurecv** command.

The *ListFile* parameter specifies a list, one per line, of all the files for *ProductName*. *ListFile* is the full path name of the file that contains the relative path names of files that the product needs to have copied.

The *ProductName* parameter specifies the name of the software product to be copied.

Flags

Item	Description
-e *FinalDirectory*	Indicates the root of the file tree that the files are to be copied to. The *FinalDirectory* should be the base of the file tree. The default directory is the **/** (root) directory when this flag is not specified.
-s *StartDirectory*	Indicates the root of the file tree that the files are to be copied from.

Environment Variables

Item	Description
INUEXPAND	This flag is set to 1 by the **installp** command if file systems are to be expanded if necessary to do the copy (that is, the **-X** flag was passed). It is set to 0 if file systems are not to be expanded. If this environment variable is not set, the default is not to expand file systems.
INUTEMPDIR	This flag is set by the **installp** command to the path of the current temporary directory. If this flag is not set the default is **/tmp**.

Error Codes

The **inucp** command returns the following error codes, which are defined in **inuerr.h**.

Item	Description
INUACCS	One or both of *StartDirectory* and *FinalDirectory* are not directories.
INUBADAR	Could not archive files in **lpp.acf** file.
INUBADC1	The copy operation failed.
INUBADMN	Unrecognizable flag specified.
INUGOOD	No error conditions occurred.
INUNOAP2	Could not access the *ListFile*.
INUNODIR	No write access to *FinalDirectory*.
INUNOLPP	One or both of *StartDirectory* and *FinalDirectory* do not have the necessary permissions.
INUNOMK	Could not create a needed directory.
INUNOSPC	Insufficient space for the copy and **INUEXPAND** was not set.
INUTOOFW	One or more parameters were missing.
INUTOOMN	Too many parameters were specified.

Security

Privilege Control: You must be the root user to run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To copy all the files listed in the **/usr/lpp/X11/inst_root/al** list from the **/usr/lpp/X11/inst_root** file tree to the root directory, enter:

```
inucp  -s /usr/lpp/X11/inst_root /usr/lpp/X11/inst_root/al X11
```

inudocm Command

Purpose

Displays contents of files containing supplemental information.

Syntax

inudocm [**-d** *Device*] [**-q**] { *ProductName* ... | **all** }

Description

> **Note:** This command is used by the **installp** command and is not recommended as a way to get README information.(See **installp -i**.)

The **inudocm** command is used to display supplemental information. The files from the media that are displayed, if they exist, are the **lpp.doc** file, the **lpp.instr** file, the **lpp.README** file and the **README** file.

The *ProductName* parameter specifies the name of the software product being checked. Specify **all** to display information about all software products that are known to the system.

Flags

Item	Description
-d *Device*	Specifies where the installation media can be found. The *Device* parameter can specify a hardware device, such as a tape or diskette drive, a directory that contains installation images, or an installation image file. The default device is **/dev/rfd0**.
-q	Specifies quiet mode, which suppresses prompts.

Security

Privilege Control: Only a root user can run this command.

Examples

To display the update instructions for the **snaserv** software product on **/dev/rfd0**, enter:

```
inudocm snaserv
```

Files

Item	Description
/usr/sbin/inudocm	Contains the **inudocm** command.
/usr/lpp/*ProductName***/lpp.instr**	Specifies the update instructions for the software product.
/usr/lpp/*ProductName***/lpp.README**	Specifies special instructions for the software product.
/usr/lpp/*ProductName***/README**	Specifies special instructions for the software product.
/usr/lpp/*ProductName***/lpp.doc**	Specifies the updates to the documentation for the software product.

inulag Command

Purpose

Acts as the front end to the subroutines to manage license agreements.

Syntax

inulag **-r** [**-n** *FilesetName* | **-s** *FileName* | **-p** *Product*] [**-d** *Description* [**-m** *MessageSpecification*]] **-f** *File*

inulag **-l** | **-q** [**-c** | **-v**] [**-n** *FilesetName* | **-s** *FileName* | **-p** *Product* | **-a**]

inulag **-u** [**-n** *FilesetName* | **-s** *FileName* | **-p** *Product*]

inulag **-A**

inulag **-D**

Description

The **inulag** command manages software license agreements. The basic forms are license agreement registration, license agreement listing, license agreement deactivation, license agreement validation, and license agreement revalidation.

The **-r** flag manages software license agreement registration of a fileset installed with **installp** or an independently-installed product installed through another installer. The path to a file that is always installed with an independently-installed product must be specified with the **-s** flag when the license agreement is registered.

The **-l** flag lists software license agreement registrations. If the **-c** flag is specified, the path to the software license agreement file is displayed rather than the contents of the file.

The **-q** flag queries for existence of software license agreements. A return code of 0 is returned if a license agreement exists. If the **-a** flag is also specified, then a return code of 0 is returned if there is a pending license agreement.

The **-u** flag removes the listing of software license agreements for a fileset or independently-installed product.

The **-D** flag forces revalidation of software license agreements upon the next system reboot.

Flags

Item	Description
-a	Used with the **-l** flag to show products that have a pending license agreement.
-A	Registers agreements for all pending license agreements.
-c	Used with the **-l** flag for colon-separated listing. Cannot be used with the **-v** flag.
-d *Description*	Specifies the default description for the fileset or product to which license applies.
-D	Forces the revalidation all license agreements on the next reboot.
-f *File*	Specifies the pathname specification for the license agreement. A '%L' in the specification is a substitution pattern for the current locale. **en_US** is the default locale. A "%l" in the specification matches the first two characters of the locale unless the current locale is **zh_CN**, in which all five characters of the locale designation are used.
-l	Lists software license agreements.
-m *MessageSpecification*	Specifies the message catalog for a translated description of the form "catalog,set number,message number".
-n *FilesetName*	Specifies the name of a fileset registered in the software vital product database governed by the license agreement.

Item	Description
-p *Product*	Specifies the product id, a nontranslatable alphanumeric string that uniquely identifies a product.
-q	Queries for license agreements. Does not show output. The value of 0 is returned if a license agreement exists. The **-q** flag can be used with other flags to query for particular license agreements or pending license agreements.
-r	Registers a software license agreement. Requires the **-f** flag for the path to the agreement file and either the **-n** flag or the **-s** flag to indicate the fileset name or signature file containing software subject to the agreement. The **-r** flag cannot be used with the **-l**, **-q**, or the **-u** flag. License agreements are registered as pending (status='P') during system installation, and NIM SPOT installation unless the environment variable **ACCEPT_LICENSES** is set to yes.
-s *FileName*	Specifies a signature file unique to installed software that identifies software not registered in the software vital product database that is governed by the license agreement. This is for use by software products not registering into the software vital product database. This form exists for the purpose of identifying software installed but not registered in the software vital product database. The *FileName* includes the full path to the file.
-u	Removes a license agreement. This does not actually remove the license agreement file, rather it changes the status of a license agreement associated with a fileset to inactive. Inactive license agreements do not need to be reagreed to, but they do not show up when listing installed software licenses.
-v	Used with the **-l** flag for verbose listing. Cannot be used with the **-c** flag.

Security

The agreement database is writable only by root. As a result, all flags other than the **-l** flag can only be used by a user operating with root user authority.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

inurecv Command

Purpose

Recovers files saved by the **inusave** command.

Syntax

inurecv *ProductName* [*OptionList*]

Description

The **inurecv** command recovers files and archive constituent files saved from a previous **inusave** command. It uses the **update.list** and **archive.list** files from the directory specified by the **INUSAVEDIR** environment variable. The **inurecv** command recovers files saved by program-provided installation or update procedures.

The **inurecv** command is primarily called by the **installp -r** command and the **installp -C** command to recover the files for a rejected program or a program that needs to be cleaned up.

The **inurecv** command is used to recover all the files for an installable program by separate calls to **inurecv** for the root, **/usr**, and **/usr/share** file trees. The save directories for the root, **/usr**, and **/usr/share** parts of an installation are:

- **/lpp/***PackageName***/***FilesetName***/V.R.M.F.save**,
- **/usr/lpp/***PackageName***/***FilesetName***/V.R.M.F.save** , and
- **/usr/share/lpp/***PackageName***/***FilesetName***/V.R.M.F.save**

respectively, when set up by the **installp** command. *Level* refers to the level of the software product and has the format of *vv.rr.mmmm.ffff.ppppppppp*, where *vv* = version, *rr* = release, *mmmm* = modification, *ffff* = fix, and *ppppppppp* = fix ID (only for Version 3.2 images).

Parameters

Item	Description
OptionList	Specifies the full path name of a stanza file that contains the names of the separately installable options, such as **bosnet.tcp.obj**, that are to be recovered for the *ProductName* software product. The option names in the *OptionList* file must be specified one per line.
ProductName	Specifies the installable software product, such as **bosnet**, whose files are to be recovered.

Environment Variables

Item	Description
INUEXPAND	This flag is set to 1 by the **installp** command if file systems are to be expanded if necessary to do the recover (that is, the **-X** flag was passed to **installp**). It is set to 0 if file systems are not to be expanded. If this environment variable is not set, the default is not to expand file systems.
INUSAVE	This flag is set to 1 by the **installp** command if files are to be saved (that is, the **-N** flag was not passed), and otherwise set to 0. The **inurecv** command attempts to recover files if **INUSAVE** is set to 1. If **INUSAVE** is set to 0, **inurecv** performs no recovery and exits with a return code of **INUGOOD**. If this environment variable is not set, the default is to attempt to recover files.
INUSAVEDIR	The full path name to the directory where files are saved. If this environment variable is not set, then the directory used is **/usr/lpp/***ProductName***/inst_updt.save**.
ODMDIR	The Object Data Manager object repository where the software vital product data is saved. If this environment variable is not set, the default directory used is **/etc/objrepos**.

Error Codes

Item	Description
INUBADC1	A copy of a file from one directory to another was unsuccessful.
INUGOOD	No error conditions occurred.
INUNORP1	Unsuccessful replacement of a file in an archive file during program recovery.
INUNOSAV	The save directory does not exist.
INUNOSVF	A file that was saved in the save directory was not found.

Security

Privilege Control: Only the root user can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To recover all files previously saved for the **snaserv** program, enter:

```
inurecv snaserv
```

Files

/lpp/*PackageName*/*FilesetName*/V.R.M.F.save
Files saved for the root file tree.

/usr/lpp/*PackageName*/*FilesetName*/V.R.M.F.save
Files saved for the **/usr** file tree.

/usr/share/lpp/*PackageName*/*FilesetName*/V.R.M.F.save
Files saved for the **/usr/share** file tree.

inurest Command

Purpose

Performs simple archive and restore operations for the **installp** command and shell scripts. This command is used by the **installp** command and the install scripts.

Syntax

inurest [**-d** *Device*] [**-q**] *ListFile ProductName*

Description

The **inurest** command restores or archives all files listed in the file specified by the *ListFile* parameter.

If files are to be archived, there must be an archive control file, **/usr/lpp/***ProductName***/lpp.acf**, which contains entries in the following form:

```
ComponentFile LibraryFile.a.
```

If the archive control file exists, the **inurest** command compares each of the file names in the *ListFile* file to the component files listed in **/usr/lpp/***ProductName***/lpp.acf**. Whenever the **inurest** command finds a match, the file name is added to a list of files that are archived. This list is then used to archive the restored files into a copy of the corresponding archive. When the archive is finished, the copy replaces the original file.

The *ListFile* parameter specifies the full path name of a file containing the relative path names, one per line, of files that a product needs to have restored.

The *ProductName* parameter specifies the software product to be restored.

Flags

Item	Description
-d *Device*	Specifies the input device. The default device is the **/dev/rfd0** device.
-q	Specifies quiet mode. Suppresses the prompt from **restore**.

Environment Variables

Item	Description
INUEXPAND	This flag is set to 1 by the **installp** command if file systems are to be expanded if necessary to do the restore (that is, the **-X** flag was passed). It is set to 0 if file systems are not to be expanded. If this environment variable is not set, the default is not to expand file systems.
INULIBDIR	This is the directory where files that are specific to software product installation reside. If **INULIBDIR** is not set the **/usr/lpp/***ProductName* directory is used.
INUTEMPDIR	The directory to use for temporary space that is needed during the execution of this command. If this environment variable is not set, then the directory used is **/tmp**.

Error Codes

Item	Description
INUBADRC	Restoration of an updated version of files was unsuccessful.
INUBADMN	Unusable flag was specified.
INUCHDIR	Cannot change directory.
INUGOOD	No error conditions occurred.
INUNOAP2	Unable to access the apply list.
INUNORP2	Failed replacing a constituent file in the archive file.
INUTOOFW	One or more parameters are missing.
INUTOOMN	Too many parameters are specified.

Security

Privilege Control: Only the root user can run this command.

Examples

To restore all the files listed in the **ac** file for the **snaserv** program, enter:

```
inurest /usr/lpp/snaserv/ac snaserv
```

Files

Item	Description
$INULIBDIR/lpp.acf	Archive control file.

inurid Command

Purpose

Removes information that is used for the installation of diskless or dataless clients and workload partitions from the **inst_root** directories of installed software.

Syntax

inurid [**-q** | **-r**]

Description

The **inurid** command is used to remove files stored in the **inst_root** directories of installed software.

The names of these directories are of the forms: **/usr/lpp/***PackageName***/inst_root** for software products and **/usr/lpp/***PackageName***/***OptionName***/v.r.m.f/inst_root** for AIX Version 4 updates.

When this command is called, the **inst_root** directories are removed for all products and updates in the committed state. Also, an indicator is stored in the Software Vital Product Data indicating that the proper **inst_root** directory information is to be removed after the completion of each future installation action, for example, actions performed by the **installp** command.

> **Attention:** When you remove **inst_root** directories to save disk space, there are implications to doing so. By removing these directories, the system cannot be used to create workload partitions, or be used as a Shared Product Object Tree (SPOT) server of diskless or dataless clients. Also, after **inst_root** directories are removed from a system, there is no way to retrieve the directories. Therefore, the system cannot later be converted to a workload partition or SPOT server without reinstalling the entire operating system.

Flags

Item	Description
-q	Queries whether **inst_root** directories have been removed from the system. A return value of 0 indicates that **inst_root** directories have not been removed and a return value of 1 indicates that the **inst_root** directories have been removed.
-r	Requests **inst_root** directories be removed from the system.

Security

Privilege Control: You must be the root user to run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/usr/lib/instl/inurid	Contains the **inurid** command.

inusave Command

Purpose

Saves files that are installed or updated during an installation procedure. This command is used by the **installp** command and the install scripts.

Syntax

inusave *ListFile ProductName*

Description

The **inusave** command saves the files and archived files that are listed in the file specified by the *ListFile* parameter for the *ProductName* software product. The **inusave** command is designed for use with the **installp** command.

The **inusave** command creates the **/usr/lpp/***PackageName***/***FilesetName***/V.R.M.F.save** directory if it does not already exist, where *Level* has the form *vv.rr.mmmm.ffff* and *vv* = the version, *rr* = the release, *mmmm* = the modification, and *ffff* = fix. This is the directory in which the installation procedures store saved files. The save directory is defined by the **INUSAVEDIR** environment variable.

The save directories for the / (root), **/usr**, and **/usr/share** parts of an installation are:

- **/lpp/***PackageName***/***FilesetName***/V.R.M.F.save**,
- **/usr/lpp/***PackageName***/***FilesetName***/V.R.M.F.save** , and
- **/usr/share/lpp/***PackageName***/***FilesetName***/V.R.M.F.save**

respectively, when set up by the **installp** command. The **installp** command calls **inusave** for each of these three directories. The *ListFile* parameter is the full path name of the file that lists the files that are to be saved if a current copy exists.

If a file named in the *ListFile* file already exists, the **inusave** command copies that file to the **$INUSAVEDIR/update**.*n* file, where *n* is an integer assigned by the **inusave** command. If the file does not exist, the **inusave** command assumes that this entry in the *ListFile* parameter represents either a new file or a file to be archived or processed by the archive procedure described later in this section.

The **inusave** command maintains a list of saved files in the **$INUSAVEDIR/update.list** file. This file is a stanza file with an entry for each saved file. Entries in the **update.list** file resemble the following:

```
/usr/bin/chkey:
         update.n = update.1
         option = bosnet.nfs.obj
         _id = 209
         _reserved = 0
         _scratch = 0
         lpp_id = 72
         private = 0
         file_type = 0
         format = 1
         loc0 = /usr/bin/chkey
         size = 7800
         checksum = 44561

/usr/bin/domainname:
         update.n = update.2
         option = bosnet.nfs.obj
         _id = 210
         _reserved = 0
         _scratch = 0
         lpp_id = 72
         private = 0
         file_type = 0
         format = 1
         loc0 = /usr/bin/domainname
```

```
        size = 2526
        checksum = 12439
```

In the previous example **/usr/bin/chkey** (the name of the stanza) is the name of an original file that was saved and **update.1** is the name of the file in the **$INUSAVEDIR** directory to which it was copied. The file **/usr/bin/chkey** belongs to the **bosnet.nfs.obj** installable option of the software product **bosnet**. The stanza name and the first two items in the stanza (update.n and option) exist for each stanza in the **update.list** file. The remaining items in the stanza, which may vary, are information from the Software Vital Product Data (SWVPD) database.

An archived constituent file is saved if there is a valid archive control file, **lpp.acf**, in the current directory. If the **lpp.acf** file exists, the **inusave** command compares each of the file names in *ListFile* to the constituent file names in **lpp.acf**. When it finds a match, the **inusave** command uses the **ar** command to extract the constituent file from its associated archive file. It then moves the file to the **$INUSAVEDIR/archive**.*n* file, where *n* is an integer selected by the **inusave** command.

The **inusave** command maintains a list of the extracted files that have been saved in the **$INUSAVEDIR/archive.list** file. This file is a stanza file with an entry for each saved constituent file. Entries in the **archive.list** file resemble the following:

```
/prodx.filea:
        archive.n = archive.1
        arc_name = /usr/lib/productx/libprodx.a
        option = productx.option1.obj
        _id = 833
        _reserved = 0
        _scratch = 0
        lpp_id = 7
        private = 0
        file_type = 0
        format = 1
        loc0 = /prodx.filea
        loc1 = "h11,h12"
        loc2 =
"/usr/lpp/productx.filea/s11,/usr/lpp/productx.filea/s12"
        size = 1611
        checksum = 62793
```

In the previous example **/prodx.filea** (the name of the stanza) is the name of the original constituent file that was saved and **archive.1** is the name of the file in the **$INUSAVEDIR** directory to which it was copied. The **/usr/lib/productx/libprodx.a** is the full path name of the archive file defined in the **lpp.acf** archive control file. The constituent file **/prodx.filea** belongs to the **productx.option1.obj** installable option of the software product **productx**. The stanza name and the first three items in the stanza (archive.n, arc_name, and option) will exist for each stanza in the **archive.list** file. The remaining items in the stanza, which may vary, are information from the SWVPD database.

Parameters

Item	Description
ListFile	Specifies the full path name of the file containing a list of relative path names, one per line, of files that are to be saved.
ProductName	Specifies the installable software product whose files are to be saved.

Environment Variables

Item	Description
INUEXPAND	This flag is set to 1 by the **installp** command if file systems are to be expanded if necessary to do the save (that is, the **-X** flag was passed to **installp**). It is set to 0 if file systems are not to be expanded. If this environment variable is not set, the default is not to expand file systems.

Item	Description
INUSAVE	This flag is set to 1 by the **installp** command if files are to be saved (that is, the **-N** flag was not passed to **installp**). It is set to 0 if files are not to be saved. If this environment variable is not set, the default is to save files.
INUSAVEDIR	The full path name to the directory where files are to be saved. If this environment variable is not set, then the directory to be used is **/usr/lpp/***ProductName***/inst_updt.save**.
INUTEMPDIR	The directory to use for temporary space that is needed during the execution of this command. If this environment variable is not set, then the directory used is **/tmp**.

Error Codes

The following error codes are defined in **/usr/include/inuerr.h**:

Item	Description
INUBADSC	A save directory could not be created.
INUBADC2	A file could not be copied from one directory to another.
INUGOOD	No error conditions occurred.
INUNOAP1	Could not access *ListFile*.
INUTOOFW	One or more parameters were missing.
INUTOOMN	Too many parameters were specified.

Security

Privilege Control: Only the root user can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To save all the files listed in the **snaserv.al** file of the **snaserv** program, enter:

```
inusave /usr/lpp/snaserv/snaserv.al snaserv
```

Files

/usr/lpp/*PackageName*/lpp.acf
Specifies the archive control file.

/lpp/*PackageName*/*FilesetName*/V.R.M.F.save
Specifies the save directory for the root.

/usr/lpp/*PackageName*/*FilesetName*/V.R.M.F.save
Specifies the save directory for the **/usr** files.

/usr/share/lpp/*PackageName*/*FilesetName*/V.R.M.F.save
Specifies the save directory for the **/usr/share** files.

inutoc Command

Purpose

Creates a **.toc** file for directories that have backup format file install images. This command is used by the **installp** command and the install scripts.

Syntax

inutoc [*Directory*]

Description

The **inutoc** command creates the **.toc** file in *Directory*. If a **.toc** file already exists, it is recreated with new information. The default installation image *Directory* is **/usr/sys/inst.images**. The **inutoc** command adds table of contents entries in the **.toc** file for every installation image in *Directory*.

The **installp** command and the **bffcreate** command call this command automatically upon the creation or use of an installation image in a directory without a **.toc** file.

Error Codes

Item	Description
INUBADIR	Usage error or *Directory* did not specify a directory.
INUCHDIR	Unable to change directories to *Directory*.
INUCRTOC	Could not create the **.toc** file.
INUGOOD	No errors occurred.
INUSYSFL	A system call failed.

Security

Privilege Control: Only the root user can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To create the **.toc** file for the **/usr/sys/inst.images** directory, enter:

```
inutoc
```

2. To create a **.toc** file for the **/tmp/images** directory, enter:

```
inutoc /tmp/images
```

Files

Item	Description
/usr/sys/inst.images	The default directory to create a **.toc** file.
.toc	The file created by this command in the specified directory.

inuumsg Command

Purpose

Displays specific error or diagnostic messages provided by a software product's installation procedures. This command is used by the **installp** command and the install scripts.

Syntax

inuumsg *Number* [*Argument1*] [**,** *Argument2*] [**,** *Argument3*] [**,** *Argument4*]

Description

The **inuumsg** command displays error or diagnostic messages for a software product's installation procedures. Rather than each procedure having its own text, messages are maintained in a central message catalog, **/usr/lpp/msg/$LANG/inuumsg.cat**. When you run the **inuumsg** command and specify the message *Number*, the error message is displayed. Up to four string arguments, *Argument1* to *Argument4*, can be substituted into the message in the appropriate location.

Exit Status

Item	Description
0	Indicates the message was found and displayed.
1	Indicates the message was not found and not displayed.

Security

Privilege Control: Only the root user can run this command.

Examples

To see error message number 3, enter:

```
inuumsg 3
```

Files

Item	Description
/usr/lpp/msg/$LANG/inuumsg.cat	The message catalog.

inuwpar Command

Purpose

Performs software installation tasks in detached workload partitions.

Syntax

/usr/sbin/inuwpar [**-d** *directory* | **-D**] [**-G**] { **-A** | **-f** *wparnamesfile* | **-w** *wparname,...* } *cmdname* [*option ...*]

Description

The **inuwpar** command performs a software installation or maintenance task on all detached workload partitions (WPARs) or named detached WPARs. A detached workload partition is a system workload partition with a writable **/usr** file system or a writable **/opt** file system that is not shared with the global environment.

The **inuwpar** command operates on workload partitions that can be specified in the following ways:

- A comma-separated list of workload partitions that are specified by the **-w** flag.
- A list of workload partitions (one per line) in the file that is specified by the **-f** flag.
- All detached system workload partitions if you specify the **-A** flag.

If you specify the **-G** flag, **inuwpar** runs the installation command first in the global environment.

Restriction: You cannot run the **inuwpar** command on application workload partitions. You cannot successfully run the **inuwpar** command on shared system workload partitions that have read-only **/usr** and **/opt** file systems unless a relocation path is specified to the command.

If you do not specify the **-d** or **-D** flag and the options of the *cmdname* command contains a **-d** *directory* option, the **inuwpar** command attempts to mount that directory into the workload partition environment as the installation device for the command.

See the "Parameters" on page 1749 section for all installation commands that can be used with the **inuwpar** command:

Flags

Item	Description
-A	Applies the installation command to all detached system workload partitions.
-d *directory*	Specifies the directory in the WPAR where the installation directory is accessible. By default, the directory is mounted from the installation command into a temporary directory within the WPAR file system. If the options of the installation command contain a **-d** *directory* option, the directory is used as the installation directory for the command.
-D	Specifies that the directory that is used in the installation command is accessible within the WPAR file systems.
-f *wparnamesfile*	Specifies a file containing a list of detached workload partitions to which the installation command is applied.
-G	Runs the installation command within the global environment and the detached system workload partitions.
-w *wparname,...*	Specifies one or more detached workload partitions to which the installation command is applied.

Parameters

Item	Description
cmdname	Specifies the installation command to run. You can specify the following installation commands: • **geninstall** • **install_all_updates** • **installp** • **instfix** • **update_all**
option	Specifies the option to be used with the installation command.

Exit Status

Item	Description
0	The command was able to run on all applicable workload partitions. The exit value does not mean that the return code of the command that has been run on all workload partitions was necessarily 0.
>0	An error occurred.

Examples

1. To install the **bos.games** file set and all of its requisite software from the **/mydev** directory in the global environment and all detached workload partitions, enter the following command:

   ```
   inuwpar -G -A installp -qaXd /mydev bos.games
   ```

2. To install the **bos.games** file set and all of its requisite software from the **/mydev** directory in the global environment and the workload partitions that are listed in **/tmp/wparlist** file , enter the following command:

   ```
   inuwpar -G -f /tmp/wparlist installp -qaXd /mydev bos.games
   ```

3. To install all of the file sets that are associated with fix IX38794 from the **/mydev** directory in workload partitions **wpar1** and **wpar5**, enter the following command:

   ```
   inuwpar -w wpar1,wpar5 instfix -k IX38794 -d /mydev
   ```

4. To update installed software to the latest level from **/mydev** directory in all detached workload partitions, enter the following command:

   ```
   inuwpar -A install_all_updates -d /mydev
   ```

invscout Command

Purpose

Surveys the host system for currently installed microcode or Vital Product Data (VPD).

Syntax

invscout [**-c**] **-v** [**-m** *machine_type_and_model*] [**-s** *serial_number*] [**-q**]

invscout [**-u** [*mask*]] [**-e**] [**-r**] [**-m** *machine_type_and_model*] [**-s** *serial_number*] [***-catl microcode_catalog_path***] [**-q**]

invscout [**-U** | **-UF** [*mask*]] [**-e**] [**-m** *machine_type_and_model*] [**-s** *serial_number*] [***-catl microcode_catalog_path***] [***-fl microcode_file_path***] [**-q**]

invscout [**-h** | **-g**]

Description

The **invscout** command executes one instance of the stand-alone version of the Inventory Scout process. The **invscoutd** command starts the server daemon side of a client-server version.

The Inventory Scout process supports two survey types:

- Microcode Survey
- Vital Product Data (VPD) Survey (**-v**)

Microcode Survey

A Microcode Survey gathers data from the host system on currently installed microcode for **invscout**-supported systems, devices and adapters. The following table describes the types of data that the Microcode Survey gathers and the files in which it stores the data.

Table 11. The data captured and the files produced by the Microcode Survey

File	Data stored	Associated flag	Display and print methods
Microcode Survey Upload File	A comparison of the gathered microcode levels and the latest levels available.	None. Use the **invscout** command with no flags to create a Microcode Survey Upload File.	Upload to a Web server over the Internet.

Table 11. The data captured and the files produced by the Microcode Survey (continued)

File	Data stored	Associated flag	Display and print methods
Microcode Update Results Formatted Text Report File	Contains a subset of the information recorded in the Microcode Survey Upload File. The subset includes the following information: • Information about the **invscout** execution itself. • The previous level of microcode that was installed on each device. • The level of microcode that is currently installed on each device. • The latest level of microcode that is available for each device. • For each device, the results of an attempted action to update the microcode to the latest level.	Use the **-U** option to create a Microcode Update Results Formatted Text Report File.	Print or display on a monitor.
Microcode Survey Results Formatted Text Report File	Contains a subset of the information recorded in the Microcode Survey Upload File. The subset includes the following information: • Information about the **invscout** execution itself. • The level of microcode that is currently installed on each device. • The latest level of microcode that is available for each device. • A suggested action for each device that can be applied according to the downloaded catalog.mic file.	Use the **-u** flag to send the file to the screen from where you invoked the **invscout** command.	Print or display on a monitor.

Table 11. The data captured and the files produced by the Microcode Survey (continued)			
File	**Data stored**	**Associated flag**	**Display and print methods**
Microcode Survey Formatted Text Report File **Attention:** The Microcode Survey Formatted Text Report File is deprecated. Use the Microcode Survey Results Formatted Text Report File instead.	Contains a subset of the information recorded in the Microcode Survey Upload File. The subset includes the following information: • Information about the **invscout** execution itself. • The level of microcode that is currently installed on each device.	Use the **-r** flag to send the file to the screen from where you invoked the **invscout** command. **Attention:** The **-r** flag is deprecated. Use the –u option instead.	Print or display on a monitor.

All of the previous reports can contain information on the following:

- system microcode
- service microcode
- device and adapter microcode

VPD Survey (-v)

A VPD Survey stores the system VPD in a **VPD Survey Upload File** that can be uploaded to a Web server via the Internet . Once on a Web server, a CGI forwards the file to a repository and produces a Web page indicating the status of the operation.

No formatted text report is available for VPD Surveys.

Survey Results Concatenation (-c)

This option concatenates two or more **Microcode Survey Upload Files** into a single **Microcode Survey Concatenated Upload File** or two or more VPD Survey Upload Files into a single **VPD Survey Concatenated Upload File**. A Concatenated Upload File can be uploaded to a Web server using the Internet and processed by the server CGI to give the same results as would have been obtained by uploading and processing all the component files individually. The input files can be any valid upload files but, typically, this operation is done to simplify the task of uploading the results from several host systems.

- The version of the command executing the concatenation and the versions of the commands that produced the files to be concatenated must all be the same.
- Microcode Survey Upload Files cannot be concatenated with VPD Survey Upload Files.
- Versions 2.1.0.0 and subsequent versions of this command do not require concatenation of Microcode Survey Upload Files, because the files are processed locally.

To concatenate a set of existing Microcode Survey upload files, do the following:

1. Copy the files into the **Microcode Survey Concatenation Input Directory**.
2. Execute:

```
invscout -c
```

3. Find the output **Microcode Survey Concatenated Upload File** in the same directory as the upload file for a Microcode Survey.

To concatenate a set of existing VPD Survey upload files, do the following:

1. Copy the files into the **VPD Survey Concatenation Input Directory**.
2. Execute:

```
invscout -v -c
```

3. Find the output **VPD Survey Concatenated Upload File** in the same directory as the upload file for a VPD Survey.

Flags

Item	Description
-v	Sets the survey or concatenation type to VPD (the default is Microcode).
-c	Concatenates existing survey upload files (the default is to perform a new survey).
-r	For a Microcode Survey, sends a copy of the formatted text report file to the screen from which the command was invoked. This flag is ignored if either the **-v** or the **-c** flag is used. **Attention:** This flag is deprecated. Use the –u option instead.
-m *machine_type_and_model*	For a VPD survey, allows input of the host platform machine type and model for hosts that use/require this information.
-s *serial_number*	For a VPD survey, allows input of the host serial number for hosts that use or require this information.
-catl microcode_catalog_path	Overrides the default location of the microcode catalog path.
-g	Displays the versions of this command and of the logic database currently in use.
-q	Suppresses most run-time messages.
-h	Generates a help (usage) statement. If this flag is used, all other flags are ignored.
-U *mask*	Updates devices with available microcode updates. This flag requires that the following items exist on the system: • A valid microcode catalog file • Valid microcode images Valid options for *mask* include any combination of the following values: • L, l: Latest • C, c: Current • P, p: Previous • A, a: Available • O, o: Outcome • D, d: Description • E, e: Effect • S, s: Suggested action

Item	Description
-UF *mask*	Updates devices and the system firmware with available microcode updates. This flag requires that the following items exist on the system: • A valid microcode catalog file • Valid microcode images This flag might reboot the system. Valid options for *mask* include any combination of the following values: • L, l: Latest • C, c: Current • P, p: Previous • A, a: Available • O, o: Outcome • D, d: Description • E, e: Effect • S, s: Suggested action
-u *mask*	Generates a formatted text report that identifies hardware for which microcode updates are available. This flag requires that a valid catalog file exists on the system. Valid options for *mask* include any combination of the following values: • L, l: Latest • C, c: Current • P, p: Previous • A, a: Available • O, o: Outcome • D, d: Description • E, e: Effect • S, s: Suggested action
-fl microcode_file_path	Changes the default path name of the microcode files. These files are stored in an .rpm format.
-e	Must be used with the –U, –UF, or -u flags. The -e flag sets *catalog.mic* and retrieves the updates from the fix central database.

Exit Status

This command returns the following exit values:

Item	Description
0	Indicates successful completion.
Non-zero	Indicates an error occurred.

If an error occurs, the command writes an **error log**.

Security

This command is owned by root, and is installed with the **setuid** bit ON so that any user can run it.

Examples

1. To run one Microcode Survey and send the results to a formatted text report file and an upload file, type:

   ```
   invscout
   ```

2. To run one VPD Survey and send the results to an upload file, type:

   ```
   invscout -v
   ```

3. To concatenate previously produced Microcode Survey upload files into a single upload file, type:

   ```
   invscout -c
   ```

   Note: Only applicable to Versions of this command prior to 2.1.0.0.

4. To concatenate previously produced VPD Survey upload files into a single upload file, type:

   ```
   invscout -v -c
   ```

Files

Item	Description
/usr/sbin/invscout	Contains the **invscout** command.
/var/adm/invscout/*host*.mup	Microcode Survey Upload File. The *host* variable is the host name of the system represented in the file.
/var/adm/invscout/*host*.vup	VPD Survey Upload File. The *host* variable is the host name of the system represented in the file.
/var/adm/invscout/invs.con.mup	Microcode Survey Concatenated Upload File.
/var/adm/invscout/invs.con.vup	VPD Survey Concatenated Upload File.
/var/adm/invscout/invs.err	Error log written if the command encounters an error.
/var/adm/invscout/invs.mic.con.inp	Microcode Survey Concatenation Input Directory.
/var/adm/invscout/invs.mrp	Microcode Survey Formatted Text Report File.
/var/adm/invscoup/invs.murp	Microcode Update Results Formatted Text Report File. This file identifies hardware updates that have been successfully applied.
/var/adm/invscout/invs.murrp	Microcode Survey Results Formatted Text Report File. This file identifies hardware for which updates are available.
/var/adm/invscout/invs.vpd.con.inp	VPD Survey Concatenation Input Directory
/var/adm/invscout/invscout.log	Log file.
/var/adm/invscout/microcode	Directory for microcode-related actions. Default location for microcode catalog file.
/var/adm/invscout/microcode/ catalog.mic	Default microcode catalog file.
/var/adm/invscout/tmp	Holds **invscout** temporary files. All files in this directory are deleted at the start of every execution of this command.

invscoutd Command

Purpose

Launches a permanent Inventory Scout server daemon.

Syntax

invscoutd [**-o**] [**-p** *Portno*] [**-b** *Bufsize*] [**-d** *maxcatsize*] [**-t** *Timeout*] [**-v** *Verblev*]

Description

The **invscoutd** command implements a permanent Inventory Scout server daemon on one machine in the local network of the user. The usual client is a Java applet running in the Web browser of the user, which was downloaded from a central Inventory Scout CGI application.

Daemon initialization involves reading command line options and several local Inventory Scout companion files. When in operation, each client-server transaction involves reading from a well-known socket for a text string and returning a text report over the same socket.

The daemon maintains a record of its actions in a log file. Depending on the specified verbosity level, the log lines may contain startup and shutdown banners, traces of each call, detailed internal program traces, and error statements. Depending on the specified verbosity level, startup banners may also be written to **stderr**.

Protocols

Client connections to the daemon's socket use the Internet TCP/IP protocol. In a transaction, the invoking client applet sends an action request, as a URL-encoded text string, to the server daemon. The request is by any ASCII control character (x00 to x1F), which triggers the processing of the request.

Some requests require the client to pass additional data. In these cases, the additional data immediately follow the termination byte for a length specified in the action request.

With one exception (ACTION=PING), the server daemon always returns a pseudo MIME format text report written back over the same socket connection. The pseudo MIME format is used even for error results. The daemon terminates the returned text and the transaction itself by closing the socket, resulting in an end-of-file (EOF) indication to the invoking client. The client should close the socket at its end of the connection as soon as the EOF is received.

URL-encoded message

The action request string is a standard URL-encoded string. For example:

```
"ACTION=actionword&NAME1=value1&NAME2&NAME3=word%xx+word+word\0"
```

Supported Field Names and Values		
Name	**Meaning/Use**	**Supported Values**
ACTION	See the action request table that follows.	The left-hand column of the action request table constitutes a list of supported Values.
MRDM	Allows the client to provide a (cleartext) password for any ACTION that uses/requires this information. The value is case sensitive.	Any ASCII string (case sensitive).

Supported Field Names and Values *(continued)*		
Name	**Meaning/Use**	**Supported Values**
DATALEN	This name must be present if additional binary data immediately follow an ACTION string termination byte, and must be absent if no additional data follow the termination byte. The integer value provided specifies the number of additional data bytes. If the client attempts to write more data than this, if the action does not accept the DATALEN parameter and discards any additional data, or if the action processor detects an early error, the daemon may prematurely close the client-to-server socket pipe. A transaction with n greater than a specific maximum value will immediately return an error code (see the **-d** command line option).	Any integer up to the value implied by the presence or absence of the **-d** command line option
CLIENT	Allows the client to identify itself for any ACTION that uses/requires this information.	The HSC value instructs Inventory Scout to allow certain actions that are only allowed when under the control of an HMC Inventory Scout master.
MODEL	Allows the client to inform the server of the server's model number for VPD surveys that use/require this information.	Any ASCII string of up to 25 characters (restrictions apply with some machines)
SERIAL	Allows the client to inform the server of the server's serial number for VPD surveys that use/require this information.	Any ASCII string of up to 25 characters (restrictions apply with some machines)

Note:

1. Field names and their values are separated by equal signs (=).
2. **Name**=*Value* pairs are separated by an & character.
3. The **Name** field is always case insensitive.
4. The *Value* field is case insensitive, unless documented otherwise.
5. The **ACTION**=*keyword* pair must always be present.
6. A string between ampersands without an equal sign is parsed as a **Name** with an Empty value.
7. Spaces can be represented by + (plus signs).
8. Binary characters may be coded as the escape sequence of a percent sign followed by exactly two hexadecimal chars (%xx). This escape sequence must also be used to code URL metacharacters like the &, = (equal sign), and + (plus sign) within a Value.
9. The control character termination byte must always be sent by the client.

Action Requests		
Action	**MRDM**	**Description**
PING	not required	The daemon *immediately* closes the socket, causing an immediate EOF in the client. This is the only action that does not return a result code or text of any kind. Example: `"action=ping\0"` `<EOF>`
ECHO	not required	The daemon returns a text report consisting of the original unparsed request string followed by a linefeed. A password (MRDM) is not required but is echoed if provided, along with everything else. Additional data (DATALEN) is not required but is echoed if present, as is, after the request string. For the ECHO request, DATALEN is silently truncated to a maximum of 2000. Example: `"action=ECHO&MRDM=xyz&datalen=5\0abcde"` `"RESULT=0\n"` `"\n"` `"action=ECHO&MRDM=xyz&datalen=5\n"` `"abcde"<EOF>`
URLDECODE	not required	The daemon returns a text report of the request string after parsing, and an exact copy of any subsequent data. A password (MRDM) is not required but is parsed and returned if provided. Additional data (DATALEN) is not required but is parsed and returned if provided; however, any actual additional data beyond the request string is discarded. Each numbered line of the report exhibits one parsed Name=Value pair from the original string. Example: `"action=UrlDecode&subaction=xyz\0"` `"RESULT=0\n"` `"\n"` `"  0: ACTION       UrlDecode\n"` `"  1: SUBACTION    xyz\n"` `<EOF>`
TESTPWD	required	The daemon returns RESULT=0 if the MRDM password is valid. Otherwise it returns RESULT=2. Additional data (DATALEN) is not accepted and is discarded if present. Example: `"ACTION=TESTPWD&MRDM=thepassword\0"` `"RESULT=0\n"` `"\n"` `<EOF>`
VERSIONS	not required	The daemon reports the current version numbers of the Inventory Scout itself. Additional data (DATALEN) is not accepted and is discarded if present. Example: `"ACTION=VERSIONS\0"` `"RESULT=0\n"` `"\n"` `"1.2.3.4\n"` `"5.6.7.8\n"` `<EOF>`

Action Requests *(continued)*		
Action	**MRDM**	**Description**
CATALOG	required	The daemon updates the scout's microcode catalog file with the file data passed. Both password and the data length parameter must be included in the request string. The daemon does not necessarily have to execute as root for this action but it must have file write permissions to **/var/adm/invscout/microcode/catalog.mic**. Example: `"ACTION=CATALOG&MRDM=xyz&DATALEN=17042\0"` `"...17042 bytes of ascii data..."` `"RESULT=0\n"` `"\n"` `<EOF>`
MCODES	required	The daemon executes the Microcode Survey Option. Additional data (DATALEN) is not accepted and is discarded if present. Example: `"ACTION=MCODES&MRDM=xyz\0"` `"RESULT=0\n"` `"\n"` `"Report Line 1\n"` `"Report Line 2\n"` `:` `:` `"Report Line N\n"` `<EOF>`
VPDS	required	The daemon executes the VPD Survey Option. Additional data (DATALEN) is not accepted and is discarded if present. Example: `"ACTION=VPDS&MRDM=xyz\0"` `"RESULT=0\n"` `"\n"` `"Report Line 1\n"` `"Report Line 2\n"` `:` `:` `"Report Line N\n"` `<EOF>`

Results

The daemon returns a text result in a pseudo MIME format. It returns a header consisting of one or more **Name**=*Value* pairs, each on a line by itself. The first **Name**=*Value* pair always is the result code in the form **RESULT**=*number*. The result code is always returned for every action, except for the PING action.

Internal scout result codes applicable only to the Java applet client are not documented in the following information.

An optional free-form text report may follow the header lines depending on the result code. If there is a free-form text report, the header is first terminated by an empty line, such as two adjacent linefeeds.

In any event, the result report is terminated by an EOF indicator after reading the last of the report text from the socket. The EOF also signifies the end of the transaction itself.

Result Codes	
Result=	**Description**
0	Complete success.
1	Daemon aborted due to memory allocation error. This can happen in either the parent server daemon or one of the service children.

Result Codes *(continued)*	
Result=	**Description**
2	Service child daemon aborted because the required password (MRDM=password) was missing or not valid.
3	Service child daemon aborted because the action name-value pair (ACTION=keyword) was missing or not valid.
4	Service child daemon aborted because it was unable to reset its user ID to `invscout`.
21	Service child daemon aborted due to overflow of socket input buffer. The text report part of the result is a native language error message. Client must reduce the length of the request string, or kill and restart the daemon with an increased buffer size.
22	Service child daemon aborted due to socket read error. The text report part of the result is a native language error message including the system's I/O errno string. A logfile entry will also contain the system's errno string.
23	Service child daemon aborted due to socket read timeout. The text report part of the result is a native language error message. Client must send a control character termination byte after the end of the request string, and must always send as many data bytes as specified in the DATALEN parameter. The timeout period may be changed with the -t command line argument.
24	Service child daemon aborted due to premature EOF while reading request string. The text report part of the result is a native language error message. Client must send a termination byte after the end of the request string before closing the socket connection.
25	Service child daemon aborted due to missing or invalid DATALEN parameter for an action that requires it. The text report pair of the result is a native language error message. Client must send the length of the data for all actions which pass additional binary data beyond the URL-encoded request string. Most such actions also require that the DATALEN value be limited to a specific maximum size.
26	Service child daemon aborted due to regular file I/O error, such as a permissions error, out of disk space, and so on. The text report part of the result is a native language error message. Usually, the I/O problem must be corrected on the server machine before the client can attempt the action again.
27	Service child daemon aborted because it was unable to retrieve the version number for an activity that required it.

Flags

Specify any arguments, beginning with a hyphen (-). Space is not allowed between a flag and its value.

Item	Description
-o	Overwrites an existing logfile. If the **-o** flag is not specified, new logfile lines are appended to any existing logfile.
-p *Portno*	Changes this server's port number from the default value of 808 to *Port*.
-b *Bufsize*	Inventory Scout commands are specified as URL-encoded strings read from a TCP/IP socket into a 1024 byte fixed length buffer. The **-b** flag can change the buffer size to *Bufsize* bytes if future protocol changes require a larger read buffer.
-d *maxcatsize*	Changes the maximum microcode catalog file size from the default value of 50000 to a value that you specify.

Item	Description
-t *Timeout*	The client applet writes a control character termination byte at the end of the URL-encoded request string to indicate the end of the request. If the **invscoutd** daemon does not receive the termination byte within a timeout period, it aborts the transaction and closes the socket. Similarly the client must send all bytes of the additional data specified in the DATALEN parameter with sufficient speed to prevent timeout between read blocks. The **-t** option changes the default timeout period from 30 seconds to *Timeout* seconds.
-v *Verblev*	The amount of detail written to the logfile and stderr depends on the verbosity level of the daemon. Each level incorporates the messages in the lower levels; increasing the verbosity level increases the number and types of messages that are written. The verbosity level is an integer ranging from 0 to 25. The **-v** flag changes the verbosity level from the default value of 18 to *Verblev*.

Verbosity Levels	
Level	**Description**
0	All error and status messages disabled.
5	Only fatal error messages are written. Fatal errors result in the death of the server. Usually, similar messages are written to both the *Logfile* and stderr.
10	All error messages are written. These include nonfatal errors such as protocol errors, as well as fatal errors. Nonfatal error messages are usually written only to the *Logfile*.
15	This level includes startup and shutdown banner messages. Simple banner messages are usually written to both the *Logfile* and stderr.
18	This level includes call trace status messages. Every client call results in a single trace message. This is the default level for the **invscoutd** daemon. Trace messages are written only to the *Logfile*.
20	This level includes program trace messages. Program traces are fairly detailed program execution status messages typically used for debugging purposes. This level is not suitable for usual production execution because over time, it floods the *Logfile* with large amounts of text. Trace messages are written only to the *Logfile*.
25	This is the maximum level and includes extensive program debug messages. This level is not suitable for usual production execution. Trace messages are written only to the *Logfile*.

Exit Status

This command returns the following exit values:

Item	Description
0	Indicates successful initialization
Non-zero	Indicates unsuccessful initialization

Security

The daemon must execute as effective user ID `0` (root). It is owned by root, and is installed with the "setuid" bit ON so that any user can launch it. At certain execution points, however, service children of the daemon reset their user ID to the authentication user ID `invscout`. The daemon does not execute unless the user `invscout` has been created on the host system.

By default, an accompanying cleartext password is required from the client for most operations. If the client's password does not match the system password for the authentication user ID `invscout`, the action exits with a return code. The authentication user ID cannot be changed.

Files

Item	Description
/usr/sbin/invscoutd	Contains the **invscoutd** command
/etc/security/password	Host system password file
/var/adm/invscout/microcode	Directory for microcode-related actions. Default location for microcode catalog file.
/var/adm/invscout/microcode/catalog.mic	Default microcode catalog file.
/var/adm/invscout/invscout.log	Log file

ioo Command

Purpose

Manages Input/Output (I/O) tunable parameters.

Syntax

ioo [**-p** | **-r**] [**-y**]{ **-o** *Tunable* [**=***NewValue*] }

ioo [**-p** | **-r**] [**-y**] {**-d** *Tunable*}

ioo [**-p** | **-r**] [**-y**] **-D**

ioo [**-p** | **-r**] [**-F**] **-a**

ioo -h [*Tunable*]

ioo [**-F**] **-L** [*Tunable*]

ioo [**-F**] **-x** [*Tunable*]

Note: Multiple **-o**, **-d**, **-x**, and **-L** flags are allowed.

Description

Note: The **ioo** command can be executed only by root.

The **ioo** command configures Input/Output (I/O) tuning parameters. This command sets or displays current or next boot values for all I/O tuning parameters. This command can also make permanent changes or defer changes until the next reboot. Whether the command sets or displays a parameter, is determined by the accompanying flag. The **-o** flag can either display the value of a parameter or set a new value for a parameter.

If a process reads sequentially from a file, the values that are specified by the **minpgahead** parameter determine the number of pages to be read ahead when the condition is first detected. The value that is specified by the **maxpgahead** parameter sets the maximum number of pages that are read ahead, regardless of the number of preceding sequential reads.

The operating system allows tuning of the number of file system **bufstructs** (**numfsbuf**) and the amount of data that is processed by the write behind algorithm (**numclust**).

Note: The tunable variables which apply to the entire system might not be modified from within a workload partition.

Understanding the Effect of Changing Tunable Parameters

Misuse of the **ioo** command can cause performance degradation or operating-system failure. Before you start experimenting with the **ioo** command, you must be familiar with Performance overview of the Virtual Memory Manager.

Before you modify any tunable parameter, you must first read about all its characteristics in the Tunable Parameters section, and follow any Refer To pointer, to fully understand its purpose.

You must then make sure that the Diagnosis and Tuning sections for this parameter truly apply to your situation and that changing the value of this parameter might help improve the performance of your system.

If both the Diagnosis and Tuning sections contain only "N/A", you must probably never change this parameter unless directed by AIX development.

Flags

Item	Description
-h [*Tunable*]	Displays help about the *Tunable* parameter if one is specified. Otherwise, displays the **ioo** command usage statement.
-a	Displays current, reboot (when used with **-r**) or permanent (when used with **-p**) value for all tunable parameters, one per line in pairs *tunable = value*. For the permanent option, a value is only displayed for a parameter if its reboot and current values are equal. Otherwise NONE displays as the value.
-d *Tunable*	Resets *Tunable* to its default value. If a *Tunable* must be changed (that is, it is not set to its default value) and is of type **Bosboot** or **Reboot**, or if it is of type **Incremental** and is changed from its default value, and **-r** is not used in combination, it is not changed but a warning displays.
-D	Resets all tunables to their default value. If tunables that must be changed are of type **Bosboot** or **Reboot**, or are of type **Incremental** and were changed from their default value, and **-r** is not used in combination, they are not changed but a warning displays.
-o *Tunable* [**=***NewValue*]	Displays the value or sets *Tunable* to *NewValue*. If a *Tunable* must be changed (the specified value is different from current value), and is of type **Bosboot** or **Reboot**, or if it is of type **Incremental** and its current value is bigger than the specified value, and **-r** is not used in combination, it is not changed but a warning displays. When **-r** is used without a *NewValue*, the nextboot value for tunable displays. When **-p** is used without a *NewValue*, a value displays only if the current and next boot values for the *Tunable* are the same. Otherwise NONE displays as the value.
-p	Specifies that the changes apply to both the current and reboot values when used in combination with the **-o**, **-d**, or **-D** flags. Turns on the updating of the **/etc/tunables/nextboot** file in addition to the updating of the current value. These combinations cannot be used on **Reboot** and **Bosboot** type parameters, their current value cannot be changed. When used with **-a** or **-o** without specifying a new value, the values display only if the current and next boot values for a parameter are the same. Otherwise NONE displays as the value.
-r	Makes changes that apply to **reboot** values when used with the **-o**, **-d**, or **-D** flags. That is, it turns on the updating of the **/etc/tunables/nextboot** file. If any parameter of type **Bosboot** is changed, the user is prompted to run **bosboot**. When used with **-a** or **-o** without specifying a new value, next boot values for tunables display instead of current values.

Item	Description
-F	Forces restricted tunable parameters to be displayed when you specify the **-a**, **-L** , or **-x** flag on the command line. If you do not specify the **-F** flag, restricted tunables are not included, unless they are named in association with a display flag (the **-o, -a , -x**, or **-L** flag).
-L [*Tunable*]	Lists the characteristics of one or all tunables, one per line, by using the following format:

```
NAME                      CUR    DEF    BOOT   MIN    MAX    UNIT
TYPE
     DEPENDENCIES
------------------------------------------------------------------------------
minpgahead                2      2      2      0      4K     4KB pages
D
     maxpgahead
------------------------------------------------------------------------------

maxpgahead                8      8      8      0      4K     4KB pages
D
     minpgahead
------------------------------------------------------------------------------

pd_npages                 64K    64K    64K    1      512K   4KB pages
D
------------------------------------------------------------------------------

maxrandwrt                0      0      0      0      512K   4KB pages
D
------------------------------------------------------------------------------

numclust                  1      1      1      0             16KB/cluster
D
------------------------------------------------------------------------------

numfsbufs                 196    196    196
M
------------------------------------------------------------------------------

recoveryMode              1      1      1      0      1      N/A
D
------------------------------------------------------------------------------

...
where:
    CUR = current value
    DEF = default value
    BOOT = reboot value
    MIN = minimal value
    MAX = maximum value
    UNIT = tunable unit of measure
    TYPE = parameter type: D (for Dynamic), S (for Static), R (for Reboot),
               B (for Bosboot), M (for Mount), I (for Incremental),
               C (for Connect), and d (for Deprecated)
    DEPENDENCIES = list of dependent tunable parameters, one per line
```

-x [*Tunable*]	Lists the characteristics of one or all tunables, one per line, by using the following (spreadsheet) format:

```
tunable,current,default,reboot,min,max,unit,type,{dtunable }


where:
    current = current value
    default = default value
    reboot = reboot value
    min = minimal value
    max = maximum value
    unit = tunable unit of measure
    type = parameter type: D (for Dynamic), S (for Static), R (for Reboot),
               B (for Bosboot), M (for Mount), I (for Incremental),
               C (for Connect), and d (for Deprecated)
    dtunable = space separated list of dependent tunable parameters
```

-y	Suppresses the confirmation prompt before the **bosboot** command is run.

If you modify (by using the **-o**, **-d** or **-D** flags) a restricted tunable parameter, it results in a warning message to warn the user that a tunable parameter of the restricted-use type is modified. If you also specify the **-r** or **-p** flags, you are prompted for confirmation of the change. In addition, at system reboot, the presence of restricted tunable parameters, which are in the **/etc/tunables/nextboot** file, is modified to a value that is different from their default value (by using a command line that specifies the **-r** or **-p** flags). The modification results in an error log entry that identifies the list of these modified tunable parameters.

When you modify a tunable, you can specify a tunable parameter value by using the abbreviations K, M, G, T, P, and E to indicate their correspondent values:

Abbreviation	Power of two
K	2^{10}
M	2^{20}
G	2^{30}
T	2^{40}
P	2^{50}
E	2^{60}

Thus, a tunable value of 1024 might be specified as 1-K.

Any change (with the **-o**, **-d** or **-D** flags) to a parameter of type Mount results in a message, warning you that the change is only effective for future mountings.

Any change (with the **-o**, **-d** or **-D** flags) to a parameter of type Connect results in **inetd** being restarted, and a message, warning you that the change is only effective for future socket connections.

Any attempt to change (with the **-o**, **-d** or **-D** flags) a parameter of type **Bosboot** or **Reboot** without **-r**, results in an error message.

Any attempt to change (with the **-o**, **-d** or **-D** flags but without the **-r** flag) the current value of a parameter of type **Incremental** with a new value smaller than the current value, results in an error message.

Tunable Parameters Type

All the tunable parameters that are manipulated by the tuning commands (**no**, **nfso**, **vmo**, **ioo**, `raso`, and **schedo**) are classified into these categories:

Item	Description
Dynamic	If the parameter can be changed at any time
Static	If the parameter can never be changed
Reboot	If the parameter can be changed only during reboot
Bosboot	If the parameter can be changed only by running bosboot and rebooting the machine
Mount	If changes to the parameter are only effective for future file systems or directory mounts
Incremental	If the parameter can be incrementally increased, except at boot time
Connect	If changes to the parameter are only effective for future socket connections
Deprecated	If changing this parameter is no longer supported by the current release of AIX.

For parameters of type Bosboot, whenever a change is performed, the tuning commands automatically prompt the user to ask if they want to execute the **bosboot** command. For parameters of type Connect, the tuning commands automatically restart the **inetd** daemon.

Note: The current set of parameters that is managed by the **ioo** command includes only Static, Dynamic, Mount, and Incremental types.

Compatibility Mode

When running in pre-5.2 compatibility mode (controlled by the **pre520tune** attribute of **sys0**, see **Performance tuning enhancements for AIX 5.2** in the*Performance management*), reboot values for parameters, except those parameters that are of type *Bosboot*, are not meaningful because in this mode they are not applied at boot time.

In pre-5.2 compatibility mode, setting reboot values to tuning parameters continues to be achieved by embedding calls to tuning commands in scripts that are called during the boot sequence. Parameters of type Reboot can therefore be set without the **-r** flag so that existing scripts continue to work.

This mode is automatically turned ON when a machine is migrated to AIX 5.2. For complete installations, it is turned OFF and the reboot values for parameters are set by applying the content of the **/etc/tunables/nextboot** file during the reboot sequence. Only in that mode are the **-r** and **-p** flags fully functional. See **Kernel Tuning** in *Performance Tools Guide and Reference* for more information.

Tunable Parameters

For default values and range of values for tunables, refer the **ioo** command help (**-h** <*tunable_parameter_name*>).

Item	Description
aio_active	**Purpose:** Indicates whether the AIO kernel extension is used and pinned. **Tuning:** A value of 1 indicates that the AIO kernel extension is used and pinned.
aio_maxreqs	**Purpose:** Specifies the maximum number of asynchronous I/O requests that can be outstanding at one time. **Tuning:** The specified number includes I/O requests that are in progress, as well as those requests that are waiting in queues to be initiated. The maximum number of asynchronous I/O requests cannot be less than the value of AIO_MAX, as defined in the `/usr/include/sys/limits.h` file, but can be greater. It would be appropriate for a system with a high volume of asynchronous I/O to have a maximum number of asynchronous I/O requests larger than AIO_MAX.
aio_maxservers	**Purpose:** Specifies the maximum number of AIO servers (kernel processes dedicated to asynchronous I/O processing) allowed to service slow path I/O requests. **Tuning:** This value is a per cpu value. The value of **maxservers** cannot be less than **minservers**. There can never be more than this many asynchronous I/O requests in progress at one time, so this number limits the possible I/O concurrency.

Item	Description
aio_minservers	**Purpose:** Specifies the minimum number of AIO servers (kernel processes dedicated to asynchronous I/O processing) that remain active to process slow path I/O requests. **Tuning:** This value is a per cpu value. The value of minservers cannot be greater than maxservers. When the kernel extension is loaded, no AIO servers are created regardless of the current or default settings. This value allows a minimal AIO footprint on systems where AIO is never used. As I/O requests are initiated, AIO servers are created to service them until the maximum value allowed by maxservers is reached. Once the minservers value is exceeded, the number of servers does not fall below minservers.
aio_server_inactivity	**Purpose:** Specifies how long an AIO server sleeps without servicing an I/O request. **Tuning:** When this time limit is exceeded, the server exits, unless it causes the number of available servers to fall below **minservers**. In this case the server goes back to sleep. The time the server sleeps in this rare case is the larger of the times that are specified for the current and default values for **server_inactivity**. It is a rare case and indicates that there might be an imbalance between the number of available servers and the amount of I/O.
dk_closed_path_recovery	**Purpose:** Enables or disables the support for recovering multipath I/O (MPIO) paths that were in the failed state when the MPIO disk was closed. The recovery operation is attempted periodically after the MPIO disk is closed until the MPIO path is recovered. If the MPIO paths are in the failed state and if the MPIO disk is already closed, the failed MPIO paths cannot be recovered when you set this tunable parameter to 1. You can open and close the MPIO disk, and initiate the recovery of MPIO paths for MPIO disks that are already closed by using the command such as `lsmpio -o -l hdisk`*X*. This feature is supported by default on AIX path-control modules (PCMs). **Tuning:** A value of 0 disables the support for recovering MPIO paths that are in the failed state. A value of 1 enables the support for recovering MPIO paths that are in the failed state. The default value is 0.
dk_lbp_enabled	**Purpose:** Allows you to enable or disable the support for the Logical Block Provisioning (thin-provisioning) in the AIX operating system. When disabled, AIX will not attempt to release the blocks that is not used from a thin-provisioned disk. **Tuning:** A value of 0 disables the Logical Block Provisioning (LBP) support. A value of 1 enables the LBP support. The default value is 1.

Item	Description
dk_lbp_num_bufs	**Purpose:** Defines the size for the pool of pre-allocated buffers that are used for the LBP support. **Tuning:** Controls the maximum number of *unmap* requests that can be processed by the disk driver at any given time. The buffer pool is a system-wide resource pool. On any thin-provisioned disk, only one *unmap* request can be active at a time. The default value for this parameter is 64 (buffer). For example, if you have 64 buffers then you have 32 KB (64 buffers x 512 bytes = 32 KB) of total pinned memory. The value of this tunable must be in the range of 1 - 1024.
dk_lbp_buf_size	**Purpose:** Defines the size of each buffer in the LBP buffer pool. The default value is 512 bytes. This value can be changed to 4096 (4K), in which case, blocks can released for those disks that support 4K block size. **Tuning:** The value for this tunable should be same as the largest supported block size by any disk attached with the AIX system.
j2_atimeUpdateSymlink	**Purpose:** If **j2_atimeUpdateSymlink** is set to 1, then the access time of the symbolic link of Enhanced journaled file system (JFS2 or Enhanced JFS) is updated on **readlink**. **Tuning:** A value of 0 indicates that the access time of JFS2 symbolic links is not updated on **readlink**. There is a performance penalty that is associated with turning **j2_atimeUpdateSymlink** on, so this tunable must not be changed unless there is a real need for it. **SUSv3** does not require that access time be updated on **readlink**, however JFS, and many other platforms do update the access time on **readlink**. This tunable is provided for compatibility with JFS and other UNIX conformant systems.
j2_dynamicBufferPreallocation	**Purpose:** Specifies the number of 16-K slabs to preallocate when the file system is running low of **bufstructs**. **Tuning:** A value of 16 represents 256-K. File system does not need remounting. The **bufstructs** for JFS2 are now dynamic; the number of buffers that start on the paging device is controlled by **j2_nBufferPerPagerDevice**, but buffers are allocated and destroyed dynamically after this initial value. If the number of "external pager file system I/Os blocked with no **fsbuf** (from **vmstat -v**) increases, the **j2_dynamicBufferPreallocation** must be increased for that file system, as the I/O load on the file system might be exceeding the speed of preallocation. A value of 0 disables dynamic buffer allocation completely.

Item	Description
j2_inodeCacheSize	**Purpose:** Controls the amount of memory JFS2 uses for the inode cache. **Tuning:** The value does not explicitly indicate the amount that might be used, but is instead a scaling factor; it is used in combination with the size of the main memory to determine the maximum memory usage for the inode cache.
j2_maxPageReadAhead	**Purpose:** Specifies the maximum number of pages to be read ahead when a sequentially accessed file is processed on JFS2. **Tuning:** The difference between **minfree** and **maxfree** must always be equal to or greater than **j2_maxPageReadAhead**. If run time decreases when the value of **j2_maxPageReadAhead** increases, ensure that the other performance of the other applications does not deteriorate.
j2_maxRandomWrite	**Purpose:** Specifies a threshold for random writes to accumulate in RAM before subsequent pages are flushed to disk by the write behind algorithm of JFS2. **Tuning:** The random write behind threshold is on a per-file basis. Useful if too many pages are flushed out by **syncd**.
j2_metadataCacheSize	**Purpose:** Controls the amount of memory Enhanced JFS uses for the metadata cache. **Tuning:** The value does not explicitly indicate the amount that is not used, but is instead a scaling factor; it is used in combination with the size of the main memory to determine the maximum memory usage for the inode cache.
j2_minPageReadAhead	**Purpose:** Specifies the minimum number of pages to be read ahead when processing a sequentially accessed file on Enhanced JFS. **Tuning:** Useful to increase if there are lots of large sequential accesses. Ensure that the performance of the other application does not deteriorate. Value of 0 might be useful if I/O pattern is purely random.
j2_nPagesPerWriteBehindCluster	**Purpose:** Specifies the number of pages, per cluster, that is processed by the write behind algorithm of Enhanced JFS. **Tuning:** Useful to increase if more pages must be kept in RAM before they are scheduled for I/O, when the I/O pattern is sequential. It might be appropriate to increase, if striped logical volumes or disk arrays are being used.

Item	Description
j2_nRandomCluster	**Purpose:** Specifies the distance apart (in clusters) that writes must exceed to be considered as random by the random write behind algorithm of Enhanced JFS. **Tuning:** Useful to increase if more pages must be kept in RAM before they are scheduled for I/O, when the I/O pattern is random and random write behind is enabled (**j2_maxRandomWrite**).
j2_recoveryMode	**Purpose:** Sets the behavior for recovery from JFS2 write errors. **Tuning:** The default value of 1 indicates that automatic recovery from JFS2 write errors is set. The value of 0 indicates that the file systems remain in a degraded mode until unmounted.
j2_syncByVFS	**Purpose:** Changes the delay between each invocation of sync processing for a JFS2 file system. **Tuning:** This tunable allows JFS2 file systems to be synchronized at a rate that is different from the standard sync daemon period. When this tunable is set to a nonzero value, it is the number of seconds to delay between the iterations of sync processing for each JFS2 file system. By using this tunable, the sync operations can be spread out more than the sync daemon can spread because the sync daemon handles all file systems simultaneously. It also allows changing the number of threads that handle file system sync operations.
j2_syncConcurrency	**Purpose:** Changes the number of threads that are run to sync data to JFS2 file systems. Each thread operates on a file system at a time. **Tuning:** When there are many file systems mounted, it might be necessary to increase this value to get all the file systems handled by the sync operation on a timely basis. **Note:** This value is effective only when the **j2_syncByVFS** tunable parameter is nonzero.
j2_syncDelayReport	**Purpose:** Notifies you if the time required to sync the file systems exceeds a specified number of seconds. **Tuning:** This tunable parameter sets the number of seconds that is allowed to complete sync processing for a file system. If that number is exceeded, a message is generated in the `syslog` file. This message is only informative and does not change any other sync behavior.

Item	Description
j2_syncPageCount	**Purpose:** Sets the maximum number of modified pages of a file that is written to disk by the sync system call in a single operation. **Tuning:** When an application that uses file system caching is run and does large numbers of random writes, it might be necessary to adjust this setting to avoid lengthy delays during sync operations.
j2_syncPageLimit	**Purpose:** Sets the maximum number of times that the sync system call uses **j2_syncPageCount** to limit pages written before increasing that count to allow progress on the sync operation. **Tuning:** This tunable must be set when **j2_syncPageCount** is set and must be increased if the effect of the **j2_syncPageCount** change is not sufficient.
lvm_bufcnt	**Purpose:** Specifies the number of LVM buffers for raw physical I/Os. **Tuning:** Applications performing large writes to striped raw logical volumes are not obtaining the wanted throughput rate. LVM splits large raw I/Os into multiple buffers of 128-K a piece. A value of 9 means that about 1 MB I/Os can be processed without waiting for more buffers. If a system is configured to have striped raw logical volumes and is doing writes greater than 1.125 MB, increasing this value might help the throughput of the application. If a system performs larger than 1 MB raw I/Os, it might be useful to increase this value.
maxpgahead	**Purpose:** Specifies the maximum number of pages to be read ahead when a sequentially accessed file is processed. **Tuning:** The value must be a power of two and must be greater than or equal to **minpgahead**. Observe the elapsed execution time of critical sequential-I/O-dependent applications with the time command. Because of limitations in the kernel, do not exceed 512 as the maximum value used. The difference between **minfree**, and **maxfree** must always be equal to or greater than **maxpgahead**. If execution time decreases with higher **maxpgahead**, observe other applications to ensure that their performance does not deteriorate.

Item	Description
maxrandwrt	**Purpose:** Specifies a threshold (in 4 KB pages) for random writes to accumulate in RAM before subsequent pages are flushed to disk by the write behind algorithm. **Tuning:** The random write behind threshold is on a per-file basis. The maximum value indicates the largest file size, in pages. You can change the value if **vmstat n** shows page out and I/O wait time peaks at regular intervals (usually when the sync daemon is writing pages to disk). It is useful to set this value to 1 or higher if numerous I/O occurs when **syncd** runs. A value of 0 disables random write behind and indicates that random writes stay in RAM until a sync operation. Setting **maxrandwrt** ensures that these writes get flushed to disk before the sync operation occurs. However, it might degrade performance, because the file is then being flushed each time. Tune this option to favor interactive response time over throughput. After the threshold is reached, all subsequent pages are then immediately flushed to disk. The pages up to the threshold value stay in RAM until a sync operation.
numclust	**Purpose:** Specifies the number of 16-K clusters that are processed by the sequential write behind algorithm of the VMM. **Tuning:** Useful to increase if more pages must be kept in RAM before they are scheduled for I/O, when the I/O pattern is sequential. It might be appropriate to increase if striped logical volumes or disk arrays are being used.
numfsbufs	**Purpose:** Specifies the number of file system **bufstructs**. **Tuning:** File system must be remounted. If the VMM must wait for a free **bufstruct**, it puts the process on the VMM wait list before the start I/O is issued and wakes it up, once a **bufstruct** is available. might be appropriate to increase if striped logical volumes or disk arrays are being used.
pd_npages	**Purpose:** Specifies the number of pages that must be deleted in one chunk from RAM when a file is deleted. **Tuning:** The maximum value indicates the largest file size, in pages. Real-time applications that experience sluggish response time while files are being deleted. Tuning this option is only useful for real-time applications. If real-time response is critical, adjusting this option might improve response time by spreading the removal of file pages from RAM more evenly over a workload.
posix_aio_active	**Purpose:** Indicates whether the AIO kernel extension is used and pinned. **Tuning:** A value of 1 indicates that the AIO kernel extension is used and pinned.

Item	Description
posix_aio_maxreqs	**Purpose:** Specifies the maximum number of asynchronous I/O requests that can be outstanding at one time. **Tuning:** The specified number includes I/O requests that are in progress, as well as those requests that are waiting in queues to be initiated. The maximum number of asynchronous I/O requests cannot be less than the value of AIO_MAX, as defined in the `/usr/include/sys/limits.h` file, but can be greater. It would be appropriate for a system with a high volume of asynchronous I/O to have a maximum number of asynchronous I/O requests larger than AIO_MAX.
posix_aio_maxservers	**Purpose:** Specifies the maximum number of AIO servers (kernel processes dedicated to asynchronous I/O processing) allowed to service slow path I/O requests. **Tuning:** This value is a per processor value. The value of **maxservers** cannot be less than **minservers**. There can never be more than this many asynchronous I/O requests in progress at one time, so this number limits the possible I/O concurrency.
posix_aio_minservers	**Purpose:** Specifies the minimum number of AIO servers (kernel processes dedicated to asynchronous I/O processing) that remain active to process slow path I/O requests. **Tuning:** This value is a per cpu value. The value of **minservers** cannot be greater than **maxservers**. When the kernel extension is loaded, no AIO servers are created, regardless of the current or default settings. This handling allows a minimal AIO footprint on systems where AIO is never used. As I/O requests are initiated, AIO servers are created to service them until the maximum allowed by **maxservers** is reached. Once the **minservers** values exceed, the number of servers does not fall below **minservers**.
posix_aio_server_inactivity	**Purpose:** Specifies how long an AIO server sleeps without servicing an I/O request. **Tuning:** When the time limit is exceeded, the server exits, unless it causes the number of available servers to fall below **minservers**. In this case the server goes back to sleep. The time the server sleeps in this rare case is the larger of the times that are specified for the current and default values for **server_inactivity**. It is a rare case and indicates that there might be an imbalance between the number of available servers and the amount of I/O.

Memory Usage and Statistics

To display the file system memory usage, enter the following command:

```
cat /proc/sys/fs/jfs2/memory_usage
```

This returns the metadata cache, the **inode** cache, and the total memory usage in bytes.

To display the file system statistics, enter the following command:

```
cat /proc/sys/fs/jfs2/statistics
```

This returns the number of **icache** hits, **icache** misses, and **icache** activates.

To display the system statistics related to the Logical Block Provision support such as the number of times the out-of-buffer value appears and the number of times the **unmap** operation failed, enter the following command:

```
cat/proc/sys/disk/lbp/statistics
```

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list the current and reboot value, range, unit, type, and dependencies of all tunable parameters that are managed by the **ioo** command, enter the following command:

   ```
   ioo -L
   ```

2. To list the current, default, and reboot values, range, unit, and type of the j2_recoveryMode tunable parameter, enter the following command:

   ```
   ioo -L j2_recoveryMode
   ```

   The result might be similar to the following output:

   ```
   NAME              CUR    DEF    BOOT   MIN    MAX    UNIT          TYPE
   ------------------------------------------------------------------------
   recoveryMode       1      1       1     0      1      n/a            D
   ------------------------------------------------------------------------
   ```

3. To display help information for the j2_nPagesPerWriteBehindCluster tunable parameter, enter the following command:

   ```
   ioo -h j2_nPagesPerWriteBehindCluster
   ```

4. To set maxrandwrt to 4 after the next reboot, enter the following command:

   ```
   ioo -r -o maxrandwrt=4
   ```

5. To permanently reset all ioo tunable parameters to default, enter the following command:

   ```
    ioo -p -D
   ```

6. To list the reboot value of all ioo parameters, enter the following command:

   ```
    ioo -r -a
   ```

7. To list (spreadsheet format) the current and reboot value, range, unit, type, and dependencies of all tunables parameters that are managed by the **ioo** command, enter the following command:

   ```
   ioo -x
   ```

iostat Command

Purpose

Reports Central Processing Unit (CPU) statistics, asynchronous input/output (AIO) and input/output statistics for the entire system, adapters, TTY devices, disks CD-ROMs, tapes and file systems.

Syntax

iostat [**-a**] [**-b**] [-l] [**-s**] [**-t**] [**-T**] [**-V**] [**-z**] [{ -A [-P] [-q | -Q] } | { [**-d** | **-p**] [**-D**] [**-R**] }[**-m**]] [{ **-f** | **-F** } [*filesystems*,...]][**-S** *power*] [**-O** *Options*][**-@** *wparname* | **ALL** | **Global**] [*drives* ...] [*interval*] [*count*]

iostat [**-X** [-o *filename*]] [interval[count]]

Restriction: The **-a, -A, -b, -d, -D, -m, -p, -P, -q, -Q, -R, -t,** and **-z** flags, the *drives* parameter, and the *wparname* parameter are restricted inside workload partitions.

Note: You must set an interval when you are using the **-b** flag. The minimum value of the interval that you can specify is 2 seconds for the **-b** flag. The Block IO statistics need to be enabled using the raso tunable **biostat**. Once the raso tunable is enabled to collect Block IO statistics, the operating system takes couple of second to populate the statistics before it can be reported. Hence, you need to wait for few seconds before you issue the **iostat -b** command, after enabling the Block IO statistic collection.

Description

The **iostat** command is used to monitor system input/output (I/O) devices (physical and logical) that are loaded, by observing the time for which these devices are active. The **iostat** command also generates reports that can be used to change system configuration to better balance the I/O load between file systems, physical volumes, and adapters.

The **iostat** command generates an XML file when the **–X** option is specified.

The **iostat** command generates various utilization and throughput reports based on the options that you specify. On multiprocessor systems, CPU statistics are calculated system-wide as averages among all processors.

A report generated by the **iostat** command consists of system configuration information and various utilization and throughput reports. The system configuration row displays at the start of the **iostat** command and whenever there is a change in monitored configuration. In addition to system configuration, WPAR configuration is also displayed for the WPAR that has enforced resource limits when the **-@** flag is used.

The system configuration and WPAR configuration information includes the following values:

lcpu
Indicates the number of logical CPUs.

drives
Indicates the number of disks (including CDs). This information is displayed only when adapters, disks, or CDs are monitored.

tapes
Indicates the number of tapes. This information is displayed only when adapters or tapes are monitored.

ent
Indicates the entitled capacity. This information is displayed only when the partition is running with shared processor.

vdisk
Indicates the number of virtual devices. This information is displayed only when adapters, disks, or CDs are monitored.

wpars
Indicates the number of active system workload partitions. This information is displayed only when you specify the **-@** flag.

maxserver
Indicates the maximum number of AIO servers that can serve slow-path IOs. This is a system-wide value. It is displayed only if asynchronous I/O is monitored.

cpulim
Indicates the processor-resource limit for a WPAR in terms of processor units. This information is displayed only for WPARs with enforced processor-resource limit.

rset
Indicates the resource-set type (regular or exclusive) that is associated with the WPAR. This information is displayed only when there is a resource set that is associated with the WPAR.

The *Interval* parameter specifies the amount of time in seconds between each report. If the *Interval* parameter is not specified, the `iostat` command generates a single report containing statistics for the time since system startup (boot). The *Count* parameter can be specified in conjunction with the *Interval* parameter. If the *Count* parameter is specified, the value of count determines the number of reports generated at *Interval* seconds apart. If the *Interval* parameter is specified without the *Count* parameter, the **iostat** command generates reports continuously.

The **iostat** command is useful in determining whether a physical volume is becoming a performance bottleneck and if there is potential to improve the situation. The % utilization field for the physical volumes indicates how evenly the file activity is spread across the drives. A high % utilization on a physical volume is a good indication that there may be contention for this resource. Since the CPU utilization statistics are also available with the **iostat** report, the percentage of time the CPU is in I/O wait can be determined at the same time. Consider distributing data across drives if the I/O wait time is significant and the disk utilization is not evenly distributed across volumes.

Beginning with AIX 5.3, the `iostat` command reports number of physical processors consumed (physc) and the percentage of entitlement consumed (% entc) in Micro-Partitioning® environments. These metrics will only be displayed on Micro-Partitioning environments.

Note: Some system resource is consumed in maintaining disk I/O history for the **iostat** command. Use the **sysconfig** subroutine, or the SMIT to stop history accounting. While the `iostat` command is running for *Count* of iterations and if there is a change in system configuration that affects the output of `iostat` command, it prints a warning message about the configuration change. It then continues the output after printing the updated system configuration information and the header.

If you specify the **-a** flag, the information is displayed in a report in the following order:

- An adapter-header row.
- A line of statistics for the adapter.
- A disk or tape-header row and the statistics of all the disks, CD-ROMs, or tapes connected to the adapter. Such reports are generated for all the disk or tape adapters that are connected to the system.
- A line of statistics for each disk or tape that is configured.

If the *Drive* parameter is specified, only those names specified are displayed. One or more alphabetic or alphanumeric values can be specified for Drives. If you specify the *Drive* parameter, the TTY and CPU reports are displayed and the disk or tape report contains statistics for the specified drives. If a drive name that you specified is not found, the report lists that name and displays the message `Drive Not` Found and gives the report of all the available drives on the system. If you did not configure drives on the system, no disk or tape report is generated.

Restriction: The first character in the *Drive* parameter cannot be numeric.

Tape utilization report is generated only if you specified the **-p** or **-a** flag.

Note: The **-@** option is not supported when executed within a workload partition.

Reports

The **iostat** command generates four types of reports, the TTY and CPU utilization report, the disk/tape utilization report, the file system utilization report, the system throughput report and the adapter throughput report.

Tips:

- When you invoke the **iostat** command with the **-@** *ALL* option, if there is no information related to a workload partition (WPAR) for a metric, a dash (-) is displayed in the place of a value.
- When you invoke the **iostat** command with the **-@** *WparName* option or inside a WPAR, if there is no information related to a workload partition (WPAR) for a metric, that metric is marked with "@" and the system-wide value is displayed for that metric.
- If a metric is not available for that release, a dash (-) is displayed in the place of a value.

TTY and CPU Utilization Report

The first report generated by the **iostat** command is the TTY and CPU utilization report. For multiprocessor systems, the CPU values are global averages among all processors. Also, the I/O wait state is defined system-wide and not per processor. The TTY and CPU utilization report has the following format:

Column	Description
`tin`	Shows the total number of characters read by the system for all TTYs.
`tout`	Shows the total number of characters written by the system to all TTYs.
`% user`	Shows the percentage of CPU utilization that occurred while executing at the user level (application).
`% sys`	Shows the percentage of CPU utilization that occurred while executing at the system level (kernel).
`% idle`	Shows the percentage of time that the CPU or CPUs were idle and the system did not have an outstanding disk I/O request.
`% iowait`	Shows the percentage of time that the CPU or CPUs were idle during which the system had an outstanding disk I/O request.
`physc`	Shows the number or the fraction of physical processors consumed, displayed only if the partition is running with shared processor.
`% entc`	Shows the percentage of entitled capacity consumed, which is displayed only if the partition is running with shared processor. Because the time base over which this data is computed can vary, the entitled capacity percentage can sometimes exceed 100%. This excess is noticeable only with small sampling intervals.
`% rc`	Shows the percentage of the consumed processor resource. The information is displayed only for WPARs with enforced processor-resource limit.

This information is updated at regular intervals by the kernel (typically sixty times per second). The TTY report provides a collective account of characters per second received from all terminals on the system as well as the collective count of characters output per second to all terminals on the system.

Methods Used to Compute CPU Disk I/O Wait Time

The method used to compute CPU disk I/O wait time is as follows: The AIX operating system only marks an idle CPU as wio if an outstanding I/O was started on that CPU. This method can report much lower wio times when just a few threads are doing I/O and the system is otherwise idle. For example, a system with four CPUs and one thread doing I/O will report a maximum of 25 percent wio time. A system with 12 CPUs and one thread doing I/O will report a maximum of 8 percent wio time. NFS client reads/writes go through the VMM, and the time that biods spend in the VMM waiting for an I/O to complete is now reported as I/O wait time.

Disk/Tape Utilization Report

The second report generated by the **iostat** command is the disk/tape utilization report. By default, the disk utilization report is displayed, and you must specify the **-p** flag to display the tape utilization report.

When you specify the **-m** flag, the path utilization report is displayed.

The disk report provides statistics on a per-physical-disk basis, and tape utilization report provides statistics on a per-tape-basis. The default report has the following format:

Item	Description
`% tm_act`	Indicates the percentage of time the physical disk/tape was active (bandwidth utilization for the drive).
`Kbps`	Indicates the amount of data transferred (read or written) to the drive in KB per second.
`tps`	Indicates the number of transfers per second that were issued to the physical disk/tape. A transfer is an I/O request to the physical disk/tape. Multiple logical requests can be combined into a single I/O request to the disk. A transfer is of indeterminate size.
`Kb_read`	The total number of KB read.
`Kb_wrtn`	The total number of KB written.

If you specify the `-D` flag, the report has the following metrics for disk/tape. Extended metrics for disk are displayed by default and users need to specify the **–p** option for tape utilization report:

Metrics related to transfers (xfer):	
`% tm_act`	Indicates the percentage of time the physical disk/tape was active (bandwidth utilization for the drive).
`bps`	Indicates the amount of data transferred (read or written) per second to the drive. Different suffixes are used to represent the unit of transfer. Default is in bytes per second.
`tps`	Indicates the number of transfers per second that were issued to the physical disk/tape. A transfer is an I/O request to the physical disk/tape. Multiple logical requests can be combined into a single I/O request to the disk. A transfer is of indeterminate size.
`bread`	Indicates the amount of data read per second, from the drive. Different suffixes are used to represent the unit of transfer. Default is in bytes per second.
`bwrtn`	Indicates the amount of data written per second, to the drive. Different suffixes are used to represent the unit of transfer. Default is in bytes per second.

Read Service Metrics (read):	
`rps`	Indicates the number of read transfers per second.
`avgserv`	Indicates the average service time per read transfer. Different suffixes are used to represent the unit of time. Default is in milliseconds.
`minserv`	Indicates the minimum read service time. Different suffixes are used to represent the unit of time. Default is in milliseconds.
`maxserv`	Indicates the maximum read service time. Different suffixes are used to represent the unit of time. Default is in milliseconds.
`timeouts`	Indicates the number of read timeouts per second.
`fails`	Indicates the number of failed read requests per second.

Write Service Metrics (write):

wps	Indicates the number of write transfers per second.
avgserv	Indicates the average service time per write transfer. Different suffixes are used to represent the unit of time. Default is in milliseconds.
minserv	Indicates the minimum write service time. Different suffixes are used to represent the unit of time. Default is in milliseconds.
maxserv	Indicates the maximum write service time. Different suffixes are used to represent the unit of time. Default is in milliseconds.
timeouts	Indicates the number of write timeouts per second.
fails	Indicates the number of failed write requests per second.

Item	Description
Wait Queue Service Metrics (queue):	**Restriction:** These metrics are not applicable for tapes.
avgtime	Indicates the average time spent by a transfer request in the wait queue. Different suffixes are used to represent the unit of time. Default is in milliseconds.
mintime	Indicates the minimum time spent by a transfer request in the wait queue. Different suffixes are used to represent the unit of time. Default is in milliseconds.
maxtime	Indicates the maximum time spent by a transfer request in the wait queue. Different suffixes are used to represent the unit of time. Default is in milliseconds.
avgwqsz	Indicates the average wait queue size.
avgsqsz	Indicates the average service queue size.
sqfull	Indicates the number of times the service queue becomes full (that is, the disk is not accepting any more service requests) per second.

Suffix	Description
K	1000 bytes
M	1 000 000 bytes if displayed in xfer metrics. Minutes, if displayed in read/write/wait service metrics.
G	1 000 000 000 bytes.
T	1 000 000 000 000 bytes.
S	Seconds.
H	Hour.

Notes:

- For drives that do not support service time metrics, read, write and wait queue service metrics will not be displayed.
- Coherent Accelerator Processor Interface (CAPI) flash disk I/O generated by the user space programs in super pipe mode is not included in the iostat command output.

Statistics for CD-ROM devices are also reported.

Block IO Device Utilization Report

The Block IO Device Utilization report provides statistics per IO device. The report helps you in analyzing the IO statistics at VMM or filesystem, and disk layers of IO stack. The report also helps you in analyzing the performance of the IO stack. The default report has the following format:

Item	Description
device	Indicates the device name.
rbytes	Indicates the number of bytes read over the monitoring interval. Default unit is bytes; a suffix will be appended if required (1024 =K, 1024K =M).
wbytes	Indicates the number of bytes written over the monitoring interval. Default unit is bytes; a suffix will be appended if required.
rserv	Indicates the read service time per read over the monitoring interval. Different suffixes are used to represent unit, default unit is millisecond.
wserv	Indicates the write service time per write over the monitoring interval. Different suffixes are used to represent unit, default unit is millisecond.
rerr	Indicates the number of read errors over the monitoring interval. Default unit is numbers; a suffix will be appended if required (1000 = K, 1000K = M, 1000M = G).
werr	Indicates the number of write errors over the monitoring interval. Default unit is numbers; a suffix will be appended if required (1000 = K, 1000K = M, 1000M = G).
reads	Indicates the number of read requests over the monitoring interval. Default unit is numbers; a suffix will be appended if required (1000 = K, 1000K = M, 1000M = G).
writes	Indicates the number of write requests over the monitoring interval. Default unit is numbers; a suffix will be appended if required (1000 = K, 1000K = M, 1000M = G).

System Throughput Report

This report is generated if the **-s** flag is specified. This report provides statistics for the entire system. This report has the following format:

Item	Description
Kbps	Indicates the amount of data transferred (read or written) in the entire system in KB per second.
tps	Indicates the number of transfers per second issued to the entire system.
Kb_read	The total number of KB read from the entire system.
Kb_wrtn	The total number of KB written to the entire system.

Tip: The **-s** flag, when used with the **-@** or **-f** flag, displays logical and physical volume throughput, which corresponds to File Systems and Disks respectively.

Adapter Throughput Report

This report is generated if the **-a** flag is specified. This report provides statistics on an adapter-by-adapter basis (for both physical and virtual adapters). This report has the following format for a physical adapter report:

Item	Description
Kbps	Indicates the amount of data transferred (read or written) in the adapter in KB per second.
tps	Indicates the number of transfers per second issued to the adapter.
Kb_read	The total number of KB read from the adapter.
Kb_wrtn	The total number of KB written to the adapter.

The virtual adapter's default throughput report has the following format:

Item	Description
Kbps	Indicates the amount of data transferred (read or written) in the adapter in KB per second.
tps	Indicates the number of transfers per second issued to the adapter.
bkread	Number of blocks received per second from the hosting server to this adapter.
bkwrtn	Number of blocks per second sent from this adapter to the hosting server.
partition-id	The partition ID of the hosting server, which serves the requests sent by this adapter.

The virtual adapter's extended throughput report (-D option) has the following format:

Metrics related to transfers (xfer:)	
Kbps	Indicates the amount of data transferred (read or written) in the adapter in KB per second.
tps	Indicates the number of transfers per second issued to the adapter.
bkread	Number of blocks received per second from the hosting server to this adapter.
bkwrtn	Number of blocks per second sent from this adapter to the hosting server.
partition-id	The partition ID of the hosting server, which serves the requests sent by this adapter.

Adapter Read Service Metrics (read:)	
rps	Indicates the number of read requests per second.
avgserv	Indicates the average time to receive a response from the hosting server for the read request sent. Different suffixes are used to represent the unit of time. Default is in milliseconds.
minserv	Indicates the minimum time to receive a response from the hosting server for the read request sent. Different suffixes are used to represent the unit of time. Default is in milliseconds.
maxserv	Indicates the maximum time to receive a response from the hosting server for the read request sent. Different suffixes are used to represent the unit of time. Default is in milliseconds.

Adapter Write Service Metrics (write:)	
wps	Indicates the number of write requests per second.
avgserv	Indicates the average time to receive a response from the hosting server for the write request sent. Different suffixes are used to represent the unit of time. Default is in milliseconds.
minserv	Indicates the minimum time to receive a response from the hosting server for the write request sent. Different suffixes are used to represent the unit of time. Default is in milliseconds.
maxserv	Indicates the maximum time to receive a response from the hosting server for the write request sent. Different suffixes are used to represent the unit of time. Default is in milliseconds.

Adapter Wait Queue Metrics (queue:)	
avgtime	Indicates the average time spent by a transfer request in the wait queue. Different suffixes are used to represent the unit of time. Default is in milliseconds.
mintime	Indicates the minimum time spent by a transfer request in the wait queue. Different suffixes are used to represent the unit of time. Default is in milliseconds.
maxtime	Indicates the maximum time spent by a transfer request in the wait queue. Different suffixes are used to represent the unit of time. Default is in milliseconds.
avgwqsz	Indicates the average wait queue size.
avgsqsz	Indicates the average service queue size.
sqfull	Indicates the number of times the service queue becomes full (that is, the hosting server is not accepting any more service requests) per second.

Suffix	Description
K	1000 bytes.
M	1 000 000 bytes if displayed in xfer metrics. Minutes, if displayed in read/write/wait service metrics.
G	1 000 000 000 bytes.
T	1 000 000 000 000 bytes.
S	Seconds.
H	Hours.

Asynchronous I/O Report

The asynchronous I/O report has the following column headers :

Item	Description
avgc	Average global AIO request count per second for the specified interval.
avfc	Average fastpath request count per second for the specified interval.
maxgc	Maximum global AIO request count since the last time this value was fetched.
maxfc	Maximum fastpath request count since the last time this value was fetched.
maxreqs	Specifies the maximum number of asynchronous I/O requests that can be outstanding at one time.

File System Utilization Report

The file system utilization report provides statistics on a per-file-system basis. The default report has the following format:

Item	Description
Filesystem	Indicates the file system name.
% tm_act	Indicates the percentage of time the file system is active.
Kbps	Indicates the amount of data transferred (read or written) to the file system in KB per second.

Item	Description
Tps	Indicates the number of transfers per second that are issued to the file system. A transfer is of indeterminate size.
Kb_read	The total number of KBs read.
Kb_wrtn	The total number of KBs written.

Important: You must specify the disk names before you invoke the **-f** or **-F** flag. If you specify the **-f** or **-F** flag, separate file system names to be monitored by commas.

Disk Input/Output History

To improve performance, the collection of disk input/output statistics is disabled by default. To enable the collection of this data, enter the following command:

```
chdev -l sys0 -a iostat=true
```

To display the current settings, enter the following command:

```
lsattr -E -l sys0 -a iostat
```

If the collection of disk input/output history is disabled and the **iostat** command is called without an interval, the **iostat** command output displays the message `Disk History Since Boot Not Available` instead of disk statistics.

Flags

Item	Description
-a	Displays the adapter throughput report. The **-a** flag can be specified with the **-A** flag, but not when the **-q** or **-Q** flag is specified. The **-a** flag is mutually exclusive with the **-f** or **-F** flag.
-A	Displays the legacy asynchronous IO utilization report, and turns off the display of TTY utilization report.
-b	Displays the block I/O device utilization statistics. The **-b** flag is mutually exclusive to all flags, except the **-T** flag.
-d	Turns off the display of TTY utilization report or CPU utilization report. If you do not specify the **-d** or **-p** flag, then by default the **-d** flag is turned on. The **-t** and **-d** flags together turn off both disks and TTY or CPU statistics, allowed only with the **-a** or **-s** flags. The **-d** flag is mutually exclusive with the **-t** flag unless you specify the **-a** or **-s** flag, too. The **-d** flag is mutually exclusive with the **-p** flag unless you specify the **-a** or **-s** flag, too.
-D	Displays the extended tape/drive utilization report. Use the **-D** flag with the **-d** or **-p** flag. The **-D** flag is mutually exclusive with the **-t** flag unless you specify the **-a** or **-s** flag, too. The **-D** flag is mutually exclusive with the **-f** or **-F** flag.
-f	Displays the file system utilization report. The **-f** flag is mutually exclusive with the **-a** or **-D** flag. The **-f** flag can be specified with the **-A** flag, but not when the **-q** or **-Q** flag is specified.
-F	Displays the file system utilization report, and turns off other utilization reports. The **-F** flag is mutually exclusive with the **-a** or **-D** flag. The **-F** flag can be specified with the **-A** flag, but not when the **-q** or **-Q** flag is specified.
-l	Displays the output in long listing mode.
-m	Displays the path utilization report. The **-m** flag is mutually exclusive with the **-t** flag.

Item	Description
-O *Options*	Changes the content and presentation of the **iostat** report based on the values specified in option parameters. **Note:** For the *Options* description, see the Parameters section.
-p	Displays the tape utilization report. The **-p** flag is mutually exclusive with the **-d** flag unless you specify the **-a** or **-s** flag, too. **Note:** Only the Atape device utilization is reported.
`-P`	Displays the POSIX asynchronous IO utilization report, and turns off the display of TTY utilization report.
`-q`	Specifies AIO queues and their request counts. The **-q** flag can be specified only with **-A** or **-P** flag.
`-Q`	Displays a list of all the mounted file systems and the associated queue numbers with their request counts. The **-Q** flag can be specified only with **-A** or **-P** flag.
`-R`	Specifies that the reset of *min** and *max** values should happen at each interval. The default is to reset the values once when `iostat` is started. The **-R** flag can be specified only with the **-D** flag.
-s	Specifies the system throughput report. You can specify the **-a** flag with the **-A** flag, but not when you have specified the **-q** or **-Q** flag. Inside a workload partition, you can specify the **-s** flag only with the **-f** or **-F** flag.
-S *power*	Displays the processor statistics that are multiplied by a value of 10^{power}. The default value of the *power* parameter is 0. The following fields are scaled: • `% user` • `% sys` • `% idle` • `% iowait` • `physc` • `entc` **Note:** By default, the `%user`, `%sys`, `%idle`, and `%iowait` fields are relative to the processor consumption of a WPAR. When you specify the **-S** flag with a nonzero power, the `%user`, `%sys`, `%idle`, and `%iowait` fields are relative to system-wide processor consumption. **Note:** The value of power can only be between 0 and 3.
-t	Turns off the display of disk utilization report. The **-t** and **-d** flags together turn off both disks and TTY or CPU statistics, allowed only with the **-a** or **-s** flags. The **-t** flag is mutually exclusive with the **-d** flag unless you specify the **-a** or **-s** flag, too. The **-t** flag is mutually exclusive with the **-D** flag unless you specify the **-a** or **-s** flag, too. The **-t** flag is mutually exclusive with the **-m** flag.
-T	Displays the time stamp.
-V	Displays valid nonzero statistics.
-z	Resets the disk input/output statistics. Only root users can use this option.

Item	Description
-@	Reports I/O activities of a workload partition: • Specify **-@ ALL** to display the activity for the global environment and all workload partitions in the system. • Specify the **-@** flag with a list of workload partition names to display the activity for that workload partition. • Specify **-@ Global** to display the activity for the global environment only. • Specify the **-@** flag inside a WPAR to display system-wide statistics along with WPAR statistics. The **-@** flag can be specified only with **–d** and **–D**, **-f** or **-F** flags. All possible combinations of the **-s, -T, -f, -F, -d, -D** and **-l** flags are allowed. **Restriction:** The **-@** flag is mutually exclusive with **-a, -t, -z, -A, -P, -q, -Q**, and the **-m** flag.
-X	Generates the XML output. The default file name is **iostat_DDMMYYHHMM.xml** unless you specify a different file name by using the **–o** option.
-o	Specifies the file name for the XML output.

Parameters

Table 12. Parameters

Item	Description
Options	Specifies the content and presentation of each report. Use this parameter with the **-O** flag.
	`fullname=[on|off]`: Displays the full name or full path of the disk, adapter, file system path, vadapter, vdisk, and so on. Default value: off • **on**: displays the `Full_Name` column • **off**: does not display the `Full_Name` column
	`ellipsis=[on|off]`: Displays the first column name of the disk, adapter, file system, path, vadapter, vdisk and so on, in the ellipsis format. Default value: off • **on**: displays the ellipsis format • **off**: does not display the ellipsis format

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Examples

1. To display a single history since boot report for all TTY, CPU, and Disks, enter the following command:

```
iostat
```

2. To display a continuous disk report at two second intervals for the disk with the logical name `disk1`, enter the following command:

```
iostat -d disk1 2
```

3. To display six reports at two second intervals for the disk with the logical name `disk1`, enter the following command:

```
iostat disk1 2 6
```

4. To display six reports at two second intervals for all disks, enter the following command:

```
iostat -d 2 6
```

5. To display six reports at two second intervals for three disks named `disk1`, `disk2`, `disk3`, enter the following command:

```
iostat disk1 disk2 disk3 2 6
```

6. To print the System throughput report since boot, enter the following command:

```
iostat -s
```

7. To print the adapter throughput reports at 5-second intervals, enter the following command:

```
iostat -a 5
```

8. To print 10 system and adapter throughput reports at 20-second intervals, with only the TTY and CPU report (no disk reports), enter the following command:

```
iostat -sat 20 10
```

9. To print the system and adapter throughput reports with the disk utilization reports of hdisk0 and hdisk7 every 30 seconds, enter the following command:

```
iostat -sad hdisk0 hdisk7 30
```

10. To display time stamp next to each line of output of **iostat**, enter the following command:

```
iostat -T 60
```

11. To display 6 reports at 2-second intervals on AIO, enter the following command:

```
iostat -A 2 6
```

12. To display AIO statistics since boot for queues associated with all mounted file systems, enter the following command:

```
iostat -A -Q
```

13. To display extended drive report for all disks, enter the following command:

```
iostat -D
```

14. To display extended drive report for all tapes, enter the following command:

```
iostat -Dp
```

15. To display extended drive report for a specific disk, enter the following command:

```
iostat –D hdisk0
```

16. To reset the disk input/output statistics, enter the following command:

```
iostat –z
```

17. To display only file system statistics for all workload partitions, enter the following command:

```
iostat –F -@ ALL
```

18. To display system throughput of all workload partitions along with the system, enter the following command:

```
iostat -f -s -@ ALL
```

19. To display file system statistics that are appended with default O/P, enter the following command:

```
iostat –f
```

20. To display logical and physical system throughput, enter the following command:

```
iostat –s -f
```

21. To display throughput for user-specified drives and file systems, enter the following command:

```
iostat hdisk0 hdisk1 –f /dev/fslv00 /dev/fslv01 /dev/fslv02
```

22. To display the processor statistics that are multiplied by a factor of 10, enter the following command:

```
iostat -S 1
```

23. To display the full name along with the existing output, enter the following command:

```
iostat -a -O fullname=on
```

24. To display the name in the ellipsis format in the output, enter the following command:

```
iostat -a -O ellipsis=on
```

25. To display the name in the ellipsis format and as well as the fullname in the output, enter the following command:

```
iostat -a -O ellipsis=on,fullname=on
```

File

Item	Description
/usr/bin/iostat	Contains the **iostat** command.

ipcrm Command

Purpose

Removes message queue, semaphore set, or shared memory identifiers.

Syntax

ipcrm [**-m** *SharedMemoryID*] [**-M** *SharedMemoryKey*] [**-q** *MessageID*] [**-Q** *MessageKey*] [**-s** *SemaphoreID*] [**-S** *SemaphoreKey*] [**-@** *WparName*]

`ipcrm` `-r` {**-q**|**-m**|**-s**} [**-@** *WparName*] *Name*

ipcrm -r **-u** [**-o** *Owner*] [**-g** *Group*] [**-@** *WparName*]

Description

The **ipcrm** command removes one or more message queues, semaphore sets, or shared memory identifiers.

Note: The **-@** option is not supported when executed within a workload partition.

Flags

Item	Description
-g *Group*	Restricts the removal to unnamed semaphores matching the group specified.
-m *SharedMemory ID*	Removes the shared memory identifier *SharedMemoryID*. The shared memory segment and data structure associated with *SharedMemoryID* are also removed after the last detach operation.
-M *SharedMemoryKey*	Removes the shared memory identifier, created with the key *SharedMemoryKey*. The shared memory segment and data structure associated with it are also removed after the last detach operation.
-o *Owner*	Restricts the removal to unnamed semaphores matching the owner specified.
-q *MessageID*	Removes the message queue identifier *MessageID* and the message queue and data structure associated with it.
-Q *MessageKey*	Removes the message queue identifier, created with the key *MessageKey*, and the message queue and data structure associated with it.
-r	Removes named or unnamed real-time interprocess communication objects. The named real-time object is either a real-time message queue (-q), a real-time shared memory (-m), or a real-time semaphore (-s) and is identified by its *Name*.
-s *SemaphoreID*	Removes the semaphore identifier *SemaphoreID* and the set of semaphores and data structure associated with it.
-S *SemaphoreKey*	Removes the semaphore identifier, created with the key *SemaphoreKey*, and the set of semaphores and data structure associated with it.
-u	Removes all real-time unnamed semaphores. Using a descriptor on a destroyed unnamed semaphore can result in unspecified behavior.
-@ *WparName*	Removes the specified interprocess-communication construct within workload partition *WparName*.

The **msgctl**, **shmctl**, and **semctl** subroutines provide details of the remove operations. The identifiers and keys can be found by using the **ipcs** command.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To remove the shared memory segment associated with *SharedMemoryID* 18602, enter:

```
ipcrm -m 18602
```

ipcs Command

Purpose

Reports interprocess communication facility status.

Syntax

ipcs [**-m**] [**-q**] [**-s**] [**-S**] [**-P**] [**-1**] [**-a** | **-b** **-c** **-o** **-p** **-r** **-t**] [**-T**] [**-C** *CoreFile*] [**-N** *Kernel*] [**-X**] [**-@** [*WparName*]]

Description

The **ipcs** command writes to the standard output information about active interprocess communication facilities. If you do not specify any flags, the **ipcs** command writes information in a short form about currently active message queues, shared memory segments, semaphores, remote queues, and local queue headers.

The column headings and the meaning of the columns in an **ipcs** command listing follow. The letters in parentheses indicate the flags that cause the corresponding heading to appear. The designator **all** means the heading is always displayed. These flags only determine what information is provided for each facility. They do not determine which facilities are listed.

Item	Description
T	(**all**) the type of facility. There are three facility types: **q** message queue **m** shared memory segment **s** semaphore
ID	(**all**) the identifier for the facility entry.
KEY	(**all**) the key used as a parameter to the **msgget** subroutine, the **semget** subroutine, or the **shmget** subroutine to make the facility entry. **Note:** The key of a shared memory segment is changed to **IPC_PRIVATE** when the segment is removed until all processes attached to the segment detach from it.

Item	Description
MODE	(**all**) the facility access modes and flags. The mode consists of 11 characters that are interpreted as follows: The first two characters can be the following: **R** If a process is waiting on a **msgrcv** system call. **S** If a process is waiting on a **msgsnd** system call. **D** If the associated shared memory segment has been removed. It disappears when the last process attached to the segment detaches it. **C** If the associated shared memory segment is to be cleared when the first attach is run. **-** If the corresponding special flag is not set. The next nine characters are interpreted as three sets of 3 bits each. The first set refers to the owner's permissions; the next to permissions of others in the user group of the facility entry; and the last to all others. Within each set, the first character indicates permission to read, the second character indicates permission to write or alter the facility entry, and the last character is currently unused. The permissions are indicated as follows: **r** If read permission is granted. **w** If write permission is granted. **a** If alter permission is granted. **-** If the indicated permission is *not* granted.
OWNER	(**all**) The login name of the owner of the facility entry.
GROUP	(**all**) The name of the group that owns the facility entry.
CREATOR	(**a**,**c**) The login name of the creator of the facility entry.
CGROUP	(**a**,**c**) The group name of the creator of the facility entry. **Note:** For the **OWNER**, **GROUP**, **CREATOR**, and **CGROUP**, the user and group IDs display instead of the login names.
CBYTES	(**a**,**o**) The number of bytes in messages currently outstanding on the associated message queue.
QNUM	(**a**,**o**) The number of messages currently outstanding on the associated message queue.
QBYTES	(**a**,**b**) The maximum number of bytes allowed in messages outstanding on the associated message queue.
LSPID	(**a**,**p**) The ID of the last process that sent a message to the associated queue. If the last message sent was from a process in a node other than the node that holds the queue, **LSPID** is the PID of the kernel process that actually placed the message on the queue, not the PID of the sending process.

Item	Description
LRPID	(**a**,**p**) The ID of the last process that received a message from the associated queue. If the last message received was from a process in a node other than the node that holds the queue, **LRPID** is the PID of the kernel process that actually received the message on the queue, not the PID of the receiving process.
STIME	(**a**,**t**) The time when the last message was sent to the associated queue. For remote queues, this is the server time. No attempt is made to compensate for time-zone differences between the local clock and the server clock.
RTIME	(**a**,**t**) The time when the last message was received from the associated queue. For remote queues, this is the server time. No attempt is made to compensate for any time-zone differences between the local clock and the server clock.
CTIME	(**a**,**t**) The time when the associated entry was created or changed. For remote queues, this is the server time. No attempt is made to compensate for any time-zone differences between the local clock and the server clock.
NATTCH	(**a**,**o**) The number of processes attached to the associated shared memory segment.
SEGSZ	(**a**,**b**) The size of the associated shared memory segment in bytes.
CPID	(**a**,**p**) The process ID of the creator of the shared memory entry.
LPID	(**a**,**p**) The process ID of the last process to attach or detach the shared memory segment.
ATIME	(**a**,**t**) The time when the last attach was completed to the associated shared memory segment.
DTIME	(**a**,**t**) The time the last detach was completed on the associated shared memory segment.
NSEMS	(**a**,**b**) The number of semaphores in the set associated with the semaphore entry.
OTIME	(**a**,**t**) The time the last semaphore operation was completed on the set associated with the semaphore entry.
SID	(**S**) The shared memory segment id. **SID**s can be used as input to the **svmon -S** command.

This command supports multibyte character sets.

Flags

Item	Description
-a	Uses the **-b**, **-c**, **-o**, **-p** and **-t** flags.
-b	Writes the maximum number of bytes in messages on queue for message queues, the size of segments for shared memory, and the number of semaphores in each semaphores set.
-c	Writes the login name and group name of the user that made the facility.
-C*CoreFile*	Uses the file specified by the *CoreFile* parameter in place of the **/dev/mem** file. The *CoreFile* parameter is a memory image file produced by the Ctrl-(left)Alt-Pad1 key sequence.
-l	When used with the **-S** flag, writes the list of **SID**s unwrapped.
-m	Writes information about active shared memory segments.
-N*Kernel*	Uses the specified *Kernel* (the **/usr/lib/boot/unix** file is the default).

Item	Description
-o	Writes the following usage information: • Number of messages on queue • Total number of bytes in messages in queue for message queues • Number of processes attached to shared memory segments
-p	Writes process number information: • Process number of the last process to receive a message on message queues • Process number of last process to send a message on message queues • Process number of the creating process • Process number of last process to attach or detach on shared memory segments
-P	Writes the list of SIDs (segment IDs) associated with the shared memory ID, along with the number of bytes pinned to that segment and an indication of whether the segment is large-page enabled or not. If the segment is large-page enabled, a 'Y' is displayed, otherwise a '-' is displayed.
-q	Writes information about active message queues.
-r	Writes information about real-time interprocess communication objects.
-s	Writes information about active semaphore set.
-S	Writes the list of **SID** attached to shared memory id.
-t	Writes time information: • Time of the last control operation that changed the access permissions for all facilities • Time of the last **msgsnd** and **msgrcv** on message queues • Time of the last **shmat** and **shmdt** on shared memory • Time of the last **semop** on semaphore sets
-T	Writes the output of the **-t** flag with the date.
-X	Prints all available characters of each user name, group name of owner, creator, owner group, creator group instead of truncating to the first 8 characters.
-@ [*WparName*]	Reports the interprocess-communication facility status for workload partitions. If WparName is specified, the status of the interprocess communication facility is displayed for that particular workload partition. If no WparName is specified, the status of the interprocess communication facility is displayed for all active workload partitions. The name of the workload partition associated with the object is displayed. Specify **Global** as the *WparName* to display IPC object information for just that operating system environment, excluding any IPC information for workload partitions hosted from that environment.

Note:

1. If the user specifies either the **-C** or **-N** flag, the real and effective UID/GID is set to the real UID/GID of the user invoking **ipcs**.
2. Values can change while **ipcs** is running; the information it gives is guaranteed to be accurate only when it was retrieved.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

Example output from entering `ipcs` without flags:

```
IPC status from /dev/mem as of Mon Aug 14 15:03:46 1989
T    ID         KEY        MODE       OWNER     GROUP
Message Queues:
q       0    0x00010381 -Rrw-rw-rw-   root      system
q   65537    0x00010307 -Rrw-rw-rw-   root      system
q   65538    0x00010311 -Rrw-rw-rw-   root      system
q   65539    0x0001032f -Rrw-rw-rw-   root      system
q   65540    0x0001031b -Rrw-rw-rw-   root      system
q   65541    0x00010339--rw-rw-rw-    root      system
q       6    0x0002fe03 -Rrw-rw-rw-   root      system
Shared Memory:
m   65537    0x00000000 DCrw-------   root      system
m  720898    0x00010300 -Crw-rw-rw-   root      system
m   65539    0x00000000 DCrw-------   root      system
Semaphores:
s  131072    0x4d02086a --ra-ra----   root      system
s   65537    0x00000000 --ra-------   root      system
s 1310722    0x000133d0 --ra-------   7003      30720
```

Files

Item	Description
/usr/lib/boot/unix	Specifies the system kernel image.
/dev/mem	Specifies memory.
/etc/passwd	Specifies user names.
/etc/group	Specifies group names.
/usr/include/sys/ipc.h	Contains the header file.

ipfilter Command

Purpose

Extracts different operation headers from an ipreport output file and displays them in a table. Some customized nfs information regarding requests and replies is also provided.

Syntax

`ipfilter` [`-f` [`u n t x c a`]] [`-s` [`u n t x c a`]] [`-n` [`-d` *milliseconds*]] *ipreport_output_file*

Description

The **ipfilter** command extracts specific information from an ipreport output file and displays it to a table. The operation headers currently recognized are: udp, nfs, tcp, ipx, icmp, atm. The `ipfilter` command has three different types of reports:

- A single file (**ipfilter.all**) that displays a list of all the selected operations. The table displays packet number, Time, Source and Destination, Length, Sequence #, Ack #, Source Port, Destination Port, Network Interface, and Operation Type.

- Individual files for each selected header (**ipfilter.udp**, **ipfilter.nfs**, **ipfilter.tcp**, **ipfilter.ipx**, **ipfilter.icmp**, **ipfilter.atm**). The information is the same as **ipfilter.all**.
- A file **nfs.rpt** that reports on nfs requests and replies. The table contains: Transaction ID #, Type of Request, Status of Request, Call Packet Number, Time of Call, Size of Call, Reply Packet Number, Time of Reply, Size of Reply, and Elapsed millisecond between call and reply.

Flags

Item	Description
u n t x c a	Specifies operation headers (udp, nfs, tcp, ipx, and icmp and atm respectively).
-d *milliseconds*	Only Call/Reply pairs whose elapsed time is greater than *milliseconds* are to be shown.
-f [u n t x c a]	Selected operations are to be shown in **ipfilter.all**.
-n	Generates an **nfs.rpt**.
-s [u n t x c]	Separate files are to be produced for each of the selected operations.

ipreport Command

Purpose

Generates a packet trace report from the specified packet trace file.

Syntax

/usr/sbin/ipreport [**-e**] [**-r**] [**-n**] [**-s**] *LogFile*

/usr/sbin/ipreport [**-C**] [**-e**] [**-n**] [**-r**] [**-s**] [**-S**] [**-v**] [**-x**] [**-1**] [**-N**] [**-T**] [**-c count**] [**-j pktnum**] [**-X bytes**] *tracefile*

Description

The **/usr/sbin/ipreport** command generates a trace report from the specified trace file created by the **iptrace** command. The *LogFile* parameter specifies the name of the file containing the results of the Internet Protocol trace. This file is created by the **iptrace** command.

Flags

Item	Description
-c *count*	Displays the number of packets.
-C	Validates checksum.
-e	Generates the trace report in EBCDIC format. The default format is ASCII.
-j *pktnum*	Jumps to the packet number specified by the *pktnum* variable.
-n	Includes a packet number to facilitate easy comparison of different output formats.
-N	Does not resolve the names.
-r	Decodes remote procedure call (RPC) packets.
-s	Prepends the protocol specification to every line in a packet.
-S	Generates the input file on a sniffer.
-T	Represents the input file in the **tcpdump** format.

Item	Description
-v	Verbose.
-x	Prints the packets in the hexadecimal format.
-X *bytes*	Limits the hexadecimal dumps to the value determined by the *bytes* variable.
-1	Specifies the compatibility trace generated on the AIX Version 3.1 operating system.

ipsec_convert Command

Purpose

Converts IP Security tunnel export files to a format that can be imported by the IBM Secure Network Gateway.

Syntax

ipsec_convert SNG22 | **FW31** [**-f** *export_directory*]

Description

IP Security allows the importing of IBM Secure Network Gateway 2.2 and IBM Firewall 3.1 tunnels using the **imptun** command. However, these firewall products do not allow the reverse capability. The **ipsec_convert** command allows for this capability by translating exported IP Security tunnels to IBM Firewall tunnels. The translated files will be placed in the current directory.

Flags

Item	Description
SNG22 | FW31	Specifies whether the format of the resulting files will be in the format of IBM Secure Network Gateway 2.2 or IBM Firewall 3.1 format.
-f	Specifies the directory where the exported IPSec files are located.

ipsecstat Command

Purpose

Lists status of IP Security devices, IP Security crypto algorithms, and statistics of IP Security packets.

Syntax

ipsecstat [**-c**] [**-d**] [**-A**] [**-E**]

Description

The **ipsecstat** command, used without flags, displays the status of the IP Security devices, the crypto algorithms installed for IP Security, and the statistics of IP Security packets.

The command can be used with flags to only list the status of IP Security devices, to only list the installed algorithms, or to reset statistic counters (to zero).

Flags

Item	Description
-c	Resets statistics counters (after displaying current value). The **-c** flag cannot be used with any other flags.
-d	Lists only the status of the IP Security devices. The **-d** flag cannot be used with any other flags.
-A	Lists only the installed authentication algorithms. The **-A** flag cannot be used with any other flags.
-E	Lists only the installed encryption algorithms. The **-E** flag cannot be used with any other flags.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

ipsectrcbuf Command

Purpose

Lists the contents of tracing buffers in the IP Security subsystem.

Syntax

ipsectrcbuf [**-l** {**0**|**1**|**2**}]

Description

The IP Security subsystem maintains a memory resident trace buffer to help debug if there is a problem. The content of the buffer, a fixed number of the most recent trace messages, will be in a system dump or can be listed by running this command with no arguments.

Flags

Item	Description
-l	Sets the IP Security trace level. By default, of the nine IP Security trace hooks, only IPSEC_ERROR trace messages are put into the buffer. To enable or disable the other trace hooks, use the **-l** flag with one of the following values: **0** Only IPSEC_ERROR trace messages are written to the buffer. This is the default. **1** IPSEC_FILTER, IPSEC_CAPSUL, IPSEC_CRYPTO, IPSEC_TUNNEL, as well as IPSEC_ERROR trace messages are written to the buffer. **2** All IP Security trace messages are put into the buffer (that includes IPSEC_FILTER_INFO, IPSEC_CAPSUL_INFO, IPSEC_CRYPTO_INFO, and IPSEC_TUNNEL_INFO as well as those in level 1).

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

iptrace Daemon

Purpose

Provides interface-level packet tracing for Internet protocols.

Syntax

/usr/sbin/iptrace [**-a**] [**-b**] [**-e**] [**-u**] [**-P** *Protocol_list*] [**-i** *Interface*] [**-p** *Port_list*] [**-s** *Host* [**-b**]] [**-d** *Host*] [**-L** *Log_size*] [-B] [**-Q** [**-V**]] [-T] [-S *snap_length*] *LogFile*

Description

The **/usr/sbin/iptrace** daemon records Internet packets received from configured interfaces. Command flags provide a filter so that the daemon traces only packets that meeting specific criteria. Packets are traced only between the local host on which the **iptrace** daemon is started and the remote host.

If the **iptrace** process was started from a command line without the System Resource Controller (SRC), it must be stopped with the **kill -15** command. The kernel extension that is loaded by the **iptrace** daemon remains active in memory if **iptrace** is stopped in any other way.

The *LogFile* parameter specifies the name of a file to which the results of the **iptrace** command are sent. To format this file, run the **ipreport** command. The **ipreport** command might display the message `TRACING DROPPED xxxx PACKETS`. This count of dropped packets indicates only the number of packets that the **iptrace** command was unable to grab because of a large packet, the size of which exceeded the socket-receive buffer size. This message does NOT mean that the packets are being dropped by the system.

Note:

1. The file that is specified by the *LogFile* parameter must not exist on an NFS-mounted file system. Specifying an output file on an NFS-mounted file system can cause the **iptrace** daemon to hang. In this case, you might not be able to kill the **iptrace** daemon, thus, requiring that you restart the system.
2. If `iptrace` is killed with `kill -9`, it is required that you issue `iptrace -u` to unload the `bpf` kernel extensions or simply reboot. Sometimes, on a busy system, it is required that you issue `iptrace -u` multiple times because of the possibility that the kernel extension used by `iptrace` is busy processing packets.
3. The `iptrace` command supports `srcmstr` as well and can be started and stopped from the command line. If started from the command line, it can be stopped by using the `kill -9` command.

Flags

Item	Description
-a	Suppresses ARP packets.
-b	Changes the **-d** or **-s** flags to bidirectional mode.
-B	Uses BPF for packet capture. The **iptrace** command when used along with the -B option returns error if the command is run inside the WPAR.

Item	Description
-d *Host*	Records packets that are headed for the destination host-specified by the *Host* variable. The *Host* variable can be a host name or an IP address in dotted decimal format. If used with the **-b** flag, the **-d** flag records packets both going to and coming from the host-specified by the *Host* variable.
-e	Enables promiscuous mode on network adapters that support this function.
-i *Interface*	Records packets received on the interface that is specified by the *Interface* variable.
-L *Log_size*	This option causes **iptrace** to log data in such a way that the `LogFile` is copied to `LogFile.old` at the start and also every time it becomes approximately *Log_size* bytes long.
-P *Protocol_list*	Records packets that use the protocol that is specified by the *Protocol_list* variable which is a comma-separated list of protocols. The Protocols can be a decimal number or name from the **/etc/protocols** file.
-p *Port_list*	Records packets that use the port number that is specified by the *Port_list* variable which is a comma-separated list of ports. The *Port_list* variable can be a decimal number or name from the **/etc/services** file.
-Q	Enables filtered system tracing for the recorded packets. After the tracing feature is enabled, the AIX trace daemon is run to record the selected system events that are related to the network communication subsystem. **Note:** The tracing feature uses **Berkeley Packet Filter (BPF)** for packet capture.
-s *Host*	Records packets that come from the source that is host-specified by the *Host* variable. The *Host* variable can be a host name or an IP address in dotted decimal format. If used with the **-b** flag, the **-s** flag records packets both going to and coming from the host that is specified by the *Host* variable.
`-S` *snap_length*	Specifies the `snap` size (how much of each packet is actually captured from the wire) when you run the **iptrace** daemon with the **-B** flag (the **bpf** support). The command `iptrace -S 1500 /tmp/iptrace.dump` limits captured packet size to 1500 bytes. The default is 80 bytes.
`-T`	Creates a `tcpdump` compatible dump file. To read the output, use `ipreport -T` or `tcpdump -r`.
-u	Unloads the kernel extension that was loaded by the **iptrace** daemon at startup.
-V	Sets the socket debug flag (the **SO_DEBUG** socket option) and trace level on sockets. This flag must be used along with the **-Q** flag.

Exit Status

The command returns the following exit values:

Item	Description
0	The daemon has run successfully.

Item	Description
1	• No interfaces were found. • The **pcap_open_live** subroutine failed. • The **pcap_datalink** subroutine failed. • The **pcap_lookupnet** subroutine failed. • The **pcap_loop** subroutine failed. • The hostname was not found. • The address was formed incorrectly. • The WPAR did not permit the operation. • The **setpri** subroutine failed. • The **fopen** subroutine failed. • The **fstat** subroutine failed. • The interface is unknown when the daemon looks up the link type.
2	The **fread** subroutine on a trace file failed.
5	• Socket creation failed. • The specified file already exists, but the file is not a trace file.
9	• The protocol is not in the **/etc/protocols** file. • The service is not in the **/etc/services** file. • The daemon failed to load trace extension (**netintf**). • The daemon failed to unload trace extension.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To start the `iptrace` daemon with the System Resource Controller (SRC), enter the following command:

   ```
   startsrc -s iptrace -a "/tmp/nettrace"
   ```

   To stop the `iptrace` daemon with SRC enter the following command:

   ```
   stopsrc -s iptrace
   ```

2. To record packets that come in and go out to any host on every interface, enter the command in the following format:

   ```
   iptrace /tmp/nettrace
   ```

   The recorded packets are received on and sent from the local host. All packet flow between the local host and all other hosts on any interface is recorded. The trace information is placed into the `/tmp/nettrace` file.

3. To record packets that are received on an interface from a specific remote host, enter the command in the following format:

```
iptrace -i en0 -p telnet -s airmail /tmp/telnet.trace
```

The packets to be recorded are received on the en0 interface, from remote host airmail, over the telnet port. The trace information is placed into the /tmp/telnet.trace file.

4. To record packets that come in and go out from a specific remote host, enter the command in the following format:

```
iptrace -i en0 -s airmail -b /tmp/telnet.trace
```

The packets to be recorded are received on the en0 interface, from remote host airmail. The trace information is placed into the /tmp/telnet.trace file.

ipv6policy Command

Purpose

Configures or displays IPv6 policies for the default address selection that is based on RFC 3484.

Syntax

ipv6policy -add *address prefix precedence label*

ipv6policy -delete *address prefix precedence label*

ipv6policy -show

Description

You can use the **ipv6policy** command to configure IPv6 policies that override the default behavior of algorithms in RFC 3484.

Flags

Item	Description
-add	Adds new IPv6 policies to the system.
-delete	Deletes IPv6 policies from the system.
-show	Displays all existing IPv6 policies that are defined on the system.

Parameters

Item	Description
address	Specifies a valid IPv6 address.
prefix	Specifies an IPv6 prefix (a valid integer) according to RFC 3484.
precedence	Specifies a precedence value (a valid integer) according to RFC 3484.
label	Specifies a label value (a valid integer) according to RFC 3484.

Examples

To add a new ipv6 policy to the system for the address 2001:: with prefix=16, precedence=10, and label=20, enter the following command as a root user:

```
ipv6policy -add 2001:: 16 10 20
```

isC2host Command

Purpose

Determine the C2 status of a system.

Syntax

isC2host [**-i** | **-s**]

Description

The **isC2host** command returns the configuration status of the host machine. If the host has been configured to operate in C2 mode, the command exits with a zero (true) code. If the host has not been configured to operate in C2 mode, the command exits with a non-zero (false) code.

This command may be used in shell scripts where the security status of the host must be known.

The **-i** option is used to determine the installation status of the system. The C2 status of the system is determined by examining the ODM database, and the exit status indicates whether or not the system was installed in C2 mode.

The **-s** option is used to initialize AIX in C2 mode and may only be issued by the root user. The C2 status of the system is determined by examining the ODM database. On a system that has not been installed with C2, as indicated by the ODM, this option performs no operation.

Flags

Item	Description
-i	Determine the C2 installation status of the system.
-s	Set the C2 status of the system from the ODM.

Subcommands

Exit Status

0
When used with no options, the system has been initialized to operate in C2 mode. When used with the **-s** flag, the system was successfully initialized according to the C2 mode setting defined in the ODM database. When used with the **-i** flag, the system was installed with C2 enabled.

1
When used with no options, the system has not been initialized to operate in C2 mode. When used with the **-s** flag, the system could not be initialized to operate in the security mode that was defined in the ODM. When used with the **-i** flag, the system was installed with C2 enabled but is not currently operating in C2 mode.

2
When used with the **-s** option, the **isC2host** command was executed by a non-root user. When used with the **-i** option, the system was not installed with C2 enabled.

3
The **isC2host** command was executed with an invalid command line option.

Files

Item	Description
/usr/sbin/isC2host	Contains the **isC2host** command.

isCChost Command

Purpose

Determine the Common Criteria enabled status of a system.

Syntax

isCChost [**-i** | **-s**]

Description

The **isCChost** command returns the configuration status of the host machine. If the host has been configured to operate in Common Criteria enabled mode, the command exits with a zero (true) code. If the host has not been configured to operate in Common Criteria enabled mode, the command exits with a non-zero (false) code.

This command may be used in shell scripts where the security status of the host must be known.

The **-i** option is used to determine the installation status of the system. The Common Criteria enabled status of the system is determined by examining the ODM database, and the exit status indicates whether or not the system was installed in Common Criteria enabled mode.

The **-s** option is used to initialize AIX in Common Criteria enabled mode and may only be issued by the root user. The Common Criteria enabled status of the system is determined by examining the ODM database. On a system that has not been installed with Common Criteria enabled, as indicated by the ODM, this option performs no operation.

Flags

Item	Description
-i	Determine the Common Criteria enabled installation status of the system.
-s	Set the Common Criteria enabled status of the system from the ODM.

Subcommands

Exit Status

0
When used with no options, the system has been initialized to operate in Common Criteria enabled mode. When used with the **-s** flag, the system was successfully initialized according to the Common Criteria enabled mode setting defined in the ODM database. When used with the **-i** flag, the system was installed with Common Criteria enabled enabled.

1
When used with no options, the system has not been initialized to operate in Common Criteria enabled mode. When used with the **-s** flag, the system could not be initialized to operate in the security mode that was defined in the ODM. When used with the **-i** flag, the system was installed with Common Criteria enabled but is not currently operating in Common Criteria enabled mode.

2
When used with the **-s** option, the **isCChost** command was executed by a non-root user. When used with the **-i** option, the system was not installed with Common Criteria enabled.

3
The **isCChost** command was executed with an invalid command line option.

Files

Item	Description
/usr/sbin/isCChost	Contains the **isCChost** command.

isnstgtd Command

Purpose

Manages an Internet Storage Name Service (iSNS) daemon.

Syntax

isnstgtd -t *targetname* [-d debuglevel]

isnstgtd -s

isnstgtd -h

Description

The **isnstgtd** command manages the iSNS daemon. The iSNS daemon refreshes the iSNS registration with the period set into the iSNS configuration stanza file **/etc/tmiscsi/isns_servers**. It also responds to Heartbeat messages sent by a new iSNS server when it starts.

Use **isnstgtd** command with the -t flag to start the iSNS daemon. You can also start it with SRC through the following command:

```
startsrc -s isnstgtd -a '-t targetname'.
```

To kill the daemon, you have to kill the process or stop it with SRC.

Restriction
Do not run more than one isnstgtd daemons on the same machine.

Requirement
The -t or -s flags are mandatory.

The isnstgtd iSNS daemon can be used only if an iSCSI Target Mode Target is defined in the ODM database. This target must have the reg_policy attribute set to isns or slp&isns to be taken into account.

Note: When the command specifies to start the daemon in debug mode (isnstgtd -t targetname -d debuglevel) with a debuglevel greater than zero, the command is not run as a daemon.

Flags

Item	Description
-t *targetname*	Specifies the ODM defined iSCSI Target Mode Target use for iSNS communication.
-d *level*	Specifies the debug level use by isnstgtd. The debug level is between 0 (important) and 7 (debug).
-s	Prints the iSNS servers configuration (defined in the iSNS configuration stanza file **/etc/tmiscsi/isns_servers**) on stdout with the SMIT menu format.
-h	Display the help: command usage.

Examples

1. To run the command as a daemon for the defined iSCSI target tgt, enter the following command:

```
isnstgtd -t tgt
```

2. To run the command in debug mode with all debug traces, enter the following command:

```
isnstgtd -t target -d 7 &
```

System Ressource Controler (SRC)

The isnstgtd daemon can also be managed with SRC:

Item	Description
startsrc -s **isnstgtd** -a '-t *targetname* [-d *debuglevel*]'	Used to start the iSNS daemon under SRC control.
stopsrc -s **isnstgtd**	Used to stop the iSNS daemon started with SRC.
refresh -s **isnstgtd**	Used to ask the iSNS daemon under SRC control to refresh its iSNS registration refresh period set in the iSNS configuration stanza file **/etc/tmiscsi/isns_servers**.

istat Command

Purpose

Examines i-nodes.

Syntax

istat {*FileName* | *i-nodeNumber Device*}

Description

The **istat** command displays the i-node information for a particular file. You can specify the file either by providing a file or directory name with the *FileName* parameter or by providing an i-node number with the *i-nodeNumber* parameter and a device name with the *Device* parameter. You can specify the *Device* parameter as either a device name or as a mounted file system name.

If you specify the *FileName* parameter, the **istat** command writes the following information about the file:

- Device where the file resides
- i-node number of the file, on that device
- File type, such as normal, directory, and block device
- File access permissions
- Name and identification number of the owner and group

  Note: The owner and group names for remote files are taken from the local **/etc/passwd** file.
- Number of links to the file
- If the i-node is for a normal file, length of the file
- If the i-node is for a device, major and minor device designations
- Date of the last i-node update
- Date of the last file modification
- Date of the last reference to the file

If you specify the *i-nodeNumber* and *Device* parameters, the **istat** command also displays, in hexadecimal values, the block numbers recorded in the i-node.

Note: The *Device* parameter cannot refer to a remote device.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the information in the i-node corresponding to the /usr/bin/ksh file, enter:

```
istat /usr/bin/ksh
```

   This command displays the i-node information for the /usr/bin/ksh file. The information looks similar to the following:

```
Inode 10360 on device 10/6    File
Protection: r-xr-xr-x
Owner: 2(bin)     Group: 2(bin)
Link count: 2     Length 372298 bytes

Last updated:  Wed May 13 14:08:13 1992
Last modified: Wed May 13 13:57:00 1992
Last accessed: Sun Jan 31 15:49:23 1993
```

2. To display i-node information by specifying a file i-node number, enter:

```
istat 10360 /dev/hd2
```

   This command displays the information contained in the i-node identified by the number 10360 on the /dev/hd2 device. In addition to the information shown in Example 1, this displays:

```
Block pointers (hexadecimal):
2a9a   2a9b   2a9c   2a9d   2a9e   2a9f   2aa0   2aa1
```

   These numbers are addresses of the disk blocks that make up the **/usr/bin/ksh** file.

Files

Item	Description
/usr/bin/istat	Contains the **istat** command.

j

The following AIX commands begin with the letter *j*.

j2edlimit Command

Purpose

Manages quota Limits Classes for JFS2 file systems.

Syntax

To edit Quota Limits Classes:

`j2edlimit` [`-e`] [`-u` | `-g`] *Filesystem*

To list Quota Limits Classes:

`j2edlimit` `-l` [`-u` | `-g`] *Filesystem*

To Set an Existing Limits Class as the Default Limits Class:

`j2edlimit` `-d` *LimitsClassID* [`-u` | `-g`] *Filesystem*

To Assign a User or Group to a Limits Class:

`j2edlimit` `-a` *LimitsClassID* [`-u` *UserName* | `-g` *GroupName*] *Filesystem*

Description

Quotas are managed in JFS2 file systems through the use of Limits Classes. Each Limits Class has hard and soft limits for disk space and file, and grace periods for exceeding the soft limits. Individual users and groups may be assigned to a Limits Class and are then subject to the quotas defined by that class. Any user or group not assigned to a class is subject to the quotas defined by the default class (Class ID 0). Quota limits for all users or groups in a particular class can be changed by using `j2edlimit` to modify the Limits Class, without having to change or duplicate quotas for each user or group. By default, or when used with the `-e` flag, the `j2edlimit` command edits the User Limits Classes for the file system specified on the command line. When used with the `-g` flag, the `j2edlimit` command edits the Group Limits Classes for the specified file system. The command creates a temporary file that contains the file system's current limits classes, then invokes the `vi` editor (or the editor specified by the EDITOR environment variable) on the temporary file so that the limits classes can be added and modified. When the editor is exited, the command reads the temporary file and modifies the binary quota files to reflect any changes.

Note: If you specify an editor in the EDITOR environment variable, you must use the full pathname of the editor.

Fields displayed in the temporary file are:

Block Hard Limit
The total amount of 1KB blocks the user or group will be allowed to use, including temporary storage during a quota grace period.

Block Soft Limit
The number of 1KB blocks the user or group will be allowed to use during normal operations.

File Hard Limit
The total number of files the user or group will be allowed to create, including temporary files created during a quota grace period.

File Soft Limit
The number of files the user or group will be allowed to create during normal operations.

Block Grace Period
Amount of time a user can exceed the Block Soft Limit before it becomes enforced as a hard limit.

File Grace Period
Amount of time a user can exceed the File Soft Limit before it becomes enforced as a hard limit.

Note:

1. A hard limit with a value of 1 indicates that no allocations are permitted. A soft limit with a value of 1, in conjunction with a hard limit with a value of 0, indicates that allocations are permitted only on a temporary basis. Hard or soft limits can be specified in kilobytes (the default), megabytes, or gigabytes.
2. A user can exceed established soft limits for the length of the corresponding grace period. Upon expiration of the grace period, the soft limit is enforced as a hard limit. The grace period can be specified in days, hours, minutes, or seconds. A value of 0 indicates that the default grace period is imposed; a value of 1 second indicates that no grace period is granted.
3. After changing a grace period using the `j2edlimit` command, users who have already reached their old grace period must reduce their file system usage to a level below their soft limits in order to use the new grace period. In the future, when these users exceed their soft limits, the new grace period will be in effect.

Flags

Item	Description
`-a`	Assigns the User or Group specified by the -u or -g flag to the indicated Limits Class in the file system specified on the command line.
`-d`	Sets the indicated Limits Class as the default for the file system specified on the command line. By default, or with the `-u` flag, the default is set for User quotas. With the `-g` flag, the default is set for Group quotas.
`-e`	Edits the Limits Classes for the file system specified on the command line (this is the default operation for the `j2edlimit` command). By default, or with the `-u` flag, the default is set for User quotas. With the `-g` flag, the default is set for Group quotas.
`-g`	When used with the `-d`, `-l` or optional `-e` flag, performs the operation on the Group Limits Classes for the file system specified on the command line. When used with the `-a` flag, assigns the associated Group to the specified Limits Class. **Note:** If the parameter contains all numbers then it will be treated as a Group ID, and the Group ID will be assigned to the Limits Class.
`-l`	Lists the Limits Classes for the file system specified on the command line. By default, or with the `-u` flag, User limits classes are listed. With the `-g` flag, Group limits classes are listed. The format of the listing is the same as found in the temporary file when editing Limits Classes.
`-u`	When used with the `-d`, `-l` or optional `-e` flag, performs the operation on the User Limits Classes for the file system specified on the command line. When used with the `-a` flag, assigns the associated User to the specified Limits Class. **Note:** If the parameter contains all numbers then it will be treated as a User ID, and the User ID will be assigned to the Limits Class.

Security

Access Control: Only the root user can execute this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To edit User Limits Classes for the /home file system:

```
j2edlimit /home
```

2. To list Group Limits Classes for the /home file system:

```
j2edlimit -l -g /home
```

3. To set User Limits Class ID 2 as the default for the /foo file system:

```
 j2edlimit -d2 /foo
```

4. To assign user markg to Limits Class ID 1 in the /home file system:

```
 j2edlimit -a 1 -u markg /home
```

Files

Item	Description
quota.user	Contains usage and Limits information for users.
quota.group	Contains usage and Limits information for groups.
/etc/filesystems	Contains file system names and locations.

jobs Command

Purpose

Displays status of jobs in the current session.

Syntax

jobs [**-l** | **-n** | **-p**] [*JobID* ...]

Description

The **jobs** command displays the status of jobs started in the current shell environment. If no specific job is specified with the *JobID* parameter, status information for all active jobs is displayed. If a job termination is reported, the shell removes that job's process ID from the list of those known by the current shell environment.

The **/usr/bin/jobs** command does not work when operating in its own command execution environment, because that environment does not have applicable jobs to manipulate. For this reason, the **jobs** command is implemented as a Korn shell or POSIX shell regular built-in command.

If the **-p** flag is specified, output consists of one line for each process ID. If no flags are specified, standard output is a series of lines with the following fields:

Item	Description
job-number	Indicates the process group number to use with the **wait**, **fg**, **bg**, and **kill** commands. When used with these commands, prefix the job number with a % (percent sign).

Item	Description
`current`	A + (plus sign) identifies the job that will be used as a default for the **fg** or **bg** commands. This job ID can also be specified using the **%+** (percent sign, plus) or **%%** (double percent sign). A - (minus sign) identifies the job that becomes the default if the current default job exits. This job ID can also be specified using **%-** (percent sign, minus). For other jobs, the `current` field is a space character. Only one job can be identified with a +, and only one job can be identified with a -. If there is a single suspended job, that will be the current job. If there are at least two suspended jobs, then the previous job is also suspended.
`state`	Displays one of the following values (in the POSIX locale): **Running** Indicates that the job has not been suspended by a signal and has not exited. **Done** Indicates that the job completed and returned exit status 0. **Done (*code*)** Indicates that the job completed normally and that it exited with the specified non-zero exit status code. This code is expressed as a decimal number. **Stopped** Indicates that the job was suspended. **Stopped (SIGTSTP)** Indicates that the **SIGTSTP** signal suspended the job. **Stopped (SIGSTOP)** Indicates that the **SIGSTOP** signal suspended the job. **Stopped (SIGTTIN)** Indicates that the **SIGTTIN** signal suspended the job. **Stopped (SIGTTOU)** Indicates that the **SIGTTOU** signal suspended the job.
`command`	The associated command that was given to the shell.

If the **-l** flag is specified, a field containing the process group ID is inserted before the `state` field. Also, more processes in a process group may be output on separate lines, using only the `job-number` and `command` fields.

Flags

Item	Description
-l	(lowercase L) Provides more information about each job listed. This information includes the job number, current job, process group ID, state, and the command that initiated the job.
-n	Displays only jobs that have stopped or exited since last notified.
-p	Displays the process IDs for the process group leaders for the selected jobs.

By default the **jobs** command displays the status of all stopped jobs, all running background jobs, and all jobs whose status has changed but not been reported by the shell.

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To display the status of jobs in the current environment, enter:

```
jobs -l
```

   The screen displays a report similar to the following output:

```
+[4] 139  Running        CC - C foo c&
-[3] 465  Stopped        mail morris
 [2] 687  Done(1)        foo.bar&
```

2. To display the process ID for the job whose name begins with "m," enter:

```
jobs -p %m
```

   Using the jobs reported in Example 1, the screen displays the following process ID:

```
465
```

Files

Item	Description
/usr/bin/ksh	Contains the Korn shell **jobs** built-in command.
/usr/bin/jobs	Contains the **jobs** command.

join Command

Purpose

Joins the data fields of two files.

Syntax

join [**-a** *FileNumber* | **-v** *FileNumber*] [**-e** *String*] [**-o** *List*] [**-t** *Character*]
[**-1** *Field*] [**-2** *Field*] *File1 File2*

Description

The **join** command reads the files specified by the *File1* and *File2* parameters, joins lines in the files according to the flags, and writes the results to standard output. The *File1* and *File2* parameters must be text files. Both *File1* and *File2* must be sorted in the collating sequence of sort -b on the field that they are being joined by before invoking the **join** command.

One line appears in the output for each identical join field appearing in both files. The join field is the field in the input files examined by the **join** command to determine what will be included in the output. The output line consists of the join field, the rest of the line from the file specified by the *File1* parameter, and the rest of the line from the file specified by the *File2* parameter. Specify standard input in place of either the *File1* or *File2* parameter by substituting a **-** (dash) as the file name. Both input files cannot be specified with a **-** (dash).

Fields are usually separated by a space, a tab character, or a new-line character. In this case, the **join** command treats consecutive separators as one and discards leading separators.

Flags

Item	Description
-1 *Field*	Joins the two files using the field specified by the *Field* variable in the *File1* input file. The value of the *Field* variable must be a positive decimal integer.
-2 *Field*	Joins the two files using the field specified by the *Field* variable in the *File2* input file. The value of the *Field* variable must be a positive decimal integer.
-a *FileNumber*	Produces an output line for each line in the file specified by the *FileNumber* variable whose join fields do not match any line in the other input file. The output lines are produced in addition to the default output. The value of the *FileNumber* variable must be either 1 or 2, corresponding to the files specified by the *File1* and *File2* parameters, respectively. If this flag is specified with the **-v** flag, this flag is ignored.
-e *String*	Replaces empty output fields with the string specified by the *String* variable.
-o *List*	Constructs an output line to comprise the fields specified in the *List* variable. One of the following forms applies to the *List* variable: ***FileNumber.Field*** Where *FileNumber* is a file number and *Field* is a decimal-integer field number. Separate multiple fields with a **,** (comma) or space characters with quotation marks around the multiple fields. ***0* (zero)** Represents the join field. The **-o** *0* flag essentially selects the union of the join fields.
-t *Character*	Uses the character specified by the *Character* parameter as the field separator character in the input and the output. Every appearance of the character in a line is significant. The default separator is a space. With default field separation, the collating sequence is that of the **sort -b** command. If you specify **-t**, the sequence is that of a plain sort. To specify a tab character, enclose it in single quotation marks.
-v *FileNumber*	Produces an output line for each line in the file specified by the *FileNumber* variable whose join fields do not match any line in the other input file. Default output is not produced. The value of the *FileNumber* variable must be either 1 or 2, corresponding to the files specified by *File1* and *File2* parameters, respectively. If this flag is specified with the **-a** flag, the **-a** flag is ignored.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

Note: The vertical alignment shown in the following examples might not be consistent with your output.

1. To perform a simple join operation on two files where the first fields are the same, type:

```
join phonedir names
```

If the phonedir file contains the following names:

```
Adams A.        555-6235
Dickerson B.    555-1842
Erwin G.        555-1234
Jackson J.      555-0256
Lewis B.        555-3237
Norwood M.      555-5341
Smartt D.       555-1540
Wright M.       555-1234
Xandy G.        555-5015
```

and the names file contains these names and department numbers:

```
Erwin           Dept. 389
Frost           Dept. 217
Nicholson       Dept. 311
Norwood         Dept. 454
Wright          Dept. 520
Xandy           Dept. 999
```

the **join** command displays:

```
Erwin G.        555-1234        Dept. 389
Norwood M.      555-5341        Dept. 454
Wright M.       555-1234        Dept. 520
Xandy G.        555-5015        Dept. 999
```

Each line consists of the join field (the last name), followed by the rest of the line found in the phonedir file and the rest of the line in the names file.

2. To display unmatched lines with the **join** command, type:

   ```
   join  -a2 phonedir names
   ```

   If the phonedir and names files are the same as in Example 1, the **join** command displays:

   ```
   Erwin G.        555-1234         Dept. 389
   Frost                            Dept. 217
   Nicholson                        Dept. 311
   Norwood M.      555-5341         Dept. 454
   Wright M.       555-1234         Dept. 520
   Xandy G.        555-5015         Dept. 999
   ```

   This command performs the same join operation as in Example 1, and also lists the lines of names that have no match in the phonedir file. The names Frost and Nicholson are included in the listing, even though they do not have entries in the phonedir file.

3. To display selected fields with the **join** command, type:

   ```
   join  -o 2.3,2.1,1.2,1.3 phonedir names
   ```

   This displays the following fields in the order given:

Item	Description
Field 3 of names	Department number
Field 1 of names	Last name
Field 2 of phonedir	First initial
Field 3 of phonedir	Telephone number

   If the phonedir file and names files are the same as in Example 1, the **join** command displays:

   ```
   389             Erwin G.        555-1234
   454             Norwood M.      555-5341
   520             Wright M.       555-1234
   999             Xandy G.        555-5015
   ```

4. To perform the join operation on a field other than the first, type:

```
sort -b +2 -3 phonedir | join  -1 3 - numbers
```

This command combines the lines in the `phonedir` and `numbers` files, comparing the third field of the `phonedir` file to the first field of the `numbers` file.

First, this command sorts the `phonedir` file by the third field, because both files must be sorted by their join fields. The output of the **sort** command is then piped to the **join** command. The **-** (dash) by itself causes the **join** command to use this output as its first file. The `-1 3` flag defines the third field of the sorted `phonedir` file as the join field. This is compared to the first field of `numbers` because its join field is not specified with a **-2** flag.

If the `numbers` file contains:

```
555-0256
555-1234
555-5555
555-7358
```

then this command displays the names listed in the `phonedir` file or each telephone number:

```
555-0256        Jackson J.
555-1234        Erwin G.
555-1234        Wright M.
```

Note that the **join** command lists all the matches for a given field. In this case, the **join** command lists both `Erwin G.` and `Wright M.` as having the telephone number `555-1234`. The number `555-5555` is not listed because it does not appear in the `phonedir` file.

Files

Item	Description
/usr/bin/join	Contains the **join** command.
/usr/lib/nls/loc/*.src	Contains collation information.

joinvg Command

Purpose

Joins a snapshot volume group back into its original volume group.

Syntax

```
joinvg [ -f ] vgname
```

Description

Joins a snapshot volume group that was created with the **splitvg** command back into its original volume group. The snapshot volume group is deleted and the disks reactivated in the original volume group. Any stale partitions will be resynchronized by a background process.

Flags

Item	Description
-f *vgname*	Forces the join when disks in the snapshot volume group are missing or removed. The mirror copy on the missing or removed disks will be removed from the original volume group. The *vgname* parameter specifies the original volume group name with the **splitvg** command.

Security

Access Control: You must have `root` authority to run this command.

Examples

To join the original volume group, **testvg**, with the snapshot volume group **snapvg**, enter the following command:

```
joinvg testvg
```

Files

Item	Description
/usr/sbin	Directory where the **joinvg** command resides.

k

The following AIX commands begin with the letter *k*.

kdb Command

Purpose

Allows for the examining of a system or live dump or a running kernel.

Syntax

kdb -h

kdb [-c *CommandFile*] [-cp] [-i *HeaderFile*] [-l] [-script] -w -u *KernelFile*

kdb [-c *CommandFile*] [-cp] [-i *HeaderFile*] [-l] [-script] [-v] [*SystemImageFile* [*KernelFile* [*KernelModule* ...]]]

kdb [-c *CommandFile*] [-cp] [-i *HeaderFile*] [-l] [-script] [-v] [-m *SystemImageFile*] [-u *KernelFile*] [-k *KernelModule*]

Description

The **kdb** command is an interactive utility for examining an operating system image or the running kernel. The **kdb** command interprets and formats control structures in the system and provides miscellaneous functions for examining a dump.

Root permissions are required to use the **kdb** command on the active system because the **/dev/pmem** special file is used. To run the **kdb** command on the active system, type the following:

```
kdb
```

Note: Stack tracing of the current process on a running system does not work.

To invoke the **kdb** command on a system image file, type the following:

```
kdb SystemImageFile
```

When kdb starts, it looks for a **.kdbinit** file in the user's home directory and in the current working directory. If a **.kdbinit** file exists in either of these locations, kdb will execute all of the commands inside the file as if they were entered at the interactive kdb prompt. If a **.kdbinit** file exists in both of these locations, the file in the home directory will be processed first, followed by the file in the current working directory (unless the current directory is the home directory, in which case the file is processed only once).

Flags

Item	Description
-c *CommandFile*	Specifies a different name for the startup script file. If this option is used, then kdb searches for the *CommandFile* parameter in the home and current directories instead of the `.kdbinit` file.
-cp	Causes kdb to print out each command in the startup script files as the command is run. This can be used to help debug the `.kdbinit` files, or any other file specified with the -c flag. Each command is printed with a plus (+) sign in front of it.

Item	Description
`-h`	Displays a short help message in regard to command line usage and a brief listing of the available command line options.
`-i` *HeaderFile*	Makes all of the C structures defined in the *HeaderFile* parameter available for use with the kdb `print` subcommand. This option requires a C compiler to be installed on the system. If the *HeaderFile* variable needs additional `.h` files to compile, these might have to be specified with separate `-i` options as well.
`-k` *Module*	Instructs kdb to use the specified *Module* parameter as an additional kernel module for resolving symbol definitions not found in the kernel itself. Using this option is equivalent to specifying the kernel module with the *KernelModule* parameter.
`-l`	Disables the inline pager (that is, the `more (^C to quit) ?` prompt) in kdb. In this case, the `set scroll` subcommand in kdb has no effect, and the inline pager is always disabled regardless of the scroll setting.
`-m` *Image*	Instructs kdb to use the specified *Image* parameter as the system image file. Using this option is equivalent to specifying the system image file with the *SystemImageFile* parameter.
`-script`	Disables the inline pager (that is, the `more (^C to quit) ?` prompt) and disables printing of most status information when kdb starts. This option facilitates parsing of the output from the kdb command by scripts and other programs that act as a front end for kdb.
`-u` *Kernel*	Instructs kdb to use the specified *Kernel* as the kernel file for resolving symbol definitions. Using this option is equivalent to specifying the kernel with the *KernelFile* parameter.
`-v`	Displays a list of all component dump tables (CDTs) in the dump file when the **kdb** command starts. CDTs list the memory regions that are actually included in the dump. If the **kdb** command is used on a live system, this option is ignored.
`-w`	Examines a kernel file directly instead of a system image. All kdb subcommands which normally display memory locations from the system image file will instead read data directly from *KernelFile*. Subcommands which write memory are not available.

Parameters

Item	Description
KernelFile	Specifies the AIX kernel that the **kdb** command uses to resolve kernel symbol definitions. A kernel file must be available. When examining a dump, the kernel file must be the same as the kernel that was used to take the system or live dump. The default value is **/unix**.
KernelModule	Specifies the file names of any additional kernel modules that kdb uses to resolve symbol definitions not found in the kernel file itself.
SystemImageFile	Specifies the file that contains the system image. The value can indicate a system or live dump, the name of a dump device, or the **/dev/pmem** special file. The default value is **/dev/pmem**.

Examples

The following examples demonstrate invocation options for the **kdb** command:

1. To invoke the **kdb** command with the default system image and kernel image files, type the following:

   ```
   kdb
   ```

   The **kdb** program returns a (0)> prompt and waits for the entry of a subcommand.
2. To invoke the **kdb** command using a dump file named `/var/adm/ras/vmcore.0` and the UNIX kernel file named `/unix`, type the following:

   ```
   kdb /var/adm/ras/vmcore.0 /unix
   ```

   The **kdb** program returns a (0)> prompt and waits for the entry of a subcommand.
3. To invoke the **kdb** command using a live dump file named `/var/adm/ras/livedump/trc1.nocomp.200705222009.00` and the kernel file `/unix`, type the following:

   ```
   kdb /var/adm/ras/livedump/trc1.nocomp.200705222009.00
   ```

   Note: The default kernel file is **/unix**. Unlike a system dump, in a live dump, only selected data is present. For example, only the kernel thread data for threads explicitly included in the dump is present.

Files

Item	Description
/usr/sbin/kdb	Contains the **kdb** command.
/dev/pmem	Default system image file.
/unix	Default kernel file.

kdestroy Command

Purpose

Destroys a Kerberos credentials cache.

Syntax

kdestroy [**-q**] [**-c** *cache_name* | **-e** *expired_time*]

Description

The **kdestroy** command deletes a Kerberos credentials cache file.

If you specify the **-e** flag, the command checks all of the credentials cache files in the default cache directory (**/var/krb5/security/creds**) and deletes any file which contains only expired tickets, provided the tickets have been expired for the specified *expired_time*.

Flags

Flags Description

Item	Description
-c *cache_name*	Specifies the name of the credentials cache you want to destroy. The default credentials cache is destroyed if you do not specify a command flag. If the KRB5CCNAME environment variable is set, its value is used to name the default credentials (ticket) cache. This flag is mutually exclusive with the **-e** flag.
-e *expired_time*	Specifies that all credentials cache files containing expired tickets be deleted if the tickets have been expired at least as long as the *expired_time* value. The *expired_time* is expressed as *nwndnhnmns*, where: ***n*** represents a number ***w*** represents weeks ***d*** represents days ***h*** represents hours ***m*** represents minutes ***s*** represents seconds You must specify the *expired_time* components in this order but you can omit any component. For example, `4h5m` represents four hours and 5 minutes and `1w2h` represents 1 week and 2 hours. If you only specify a number, the default is hours.
-q	Suppress the beep when **kdestroy** fails to destroy the ticket.

Security

To delete a credentials cache, the user must be the owner of the file or must be a root (uid 0) user.

Examples

1. To delete the default credentials cache for the user, type:

```
kdestroy
```

2. To delete all credentials cache with expired tickets older than one day, type:

```
kdestroy -e 1d
```

Files

Files

Item	Description
/usr/krb5/bin/kdestroy	Contains the **kdestroy** command.

Files *(continued)*

Item	Description
/var/krb5/security/creds/krb5cc_[uid]	Default credentials cache (`[uid]` is the UID of the user).

keyadd Command

Purpose

keyadd retrieves objects from the source keystore and adds them to the destination keystore.

Syntax

keyadd [**-S** *servicename*] **-l** *label* **-s** *source_keystore* [**-d** *destination_keystore*] [*username*]

Description

The **keyadd** command retrieves the objects named by label from the source keystore and adds them to the destination keystore. In a keystore, a user may have the private key, public key and the certificate stored using the same label. All objects matching a label are copied regardless of the object type. If an object with the same label already exists in the destination keystore, the command returns an error. This forces the user to explicitly remove an existing object instead of blindly destroying it.

Attention: Generally, there is no way to recover a destroyed object.

The **-S** option specifies which end-entity services and libraries to use while adding the objects from the keystore. Available services are defined in **/usr/lib/security/pki/ca.cfg**. When invoked without **-S**, **keydelete** will use the default service, which is **local**. It is an error to specify a servicename which does not have an entry in the **/usr/lib/security/ pki/ca.cfg** file.

The **-l** option must be specified. This label uniquely identifies an object in the keystore to be copied. The **-s** option must also be specified.

If the **-d** option is not given, the username's default keystore file will be used as the destination keystore The user's default keystore location is **/var/pki/security/keys/<*username*>**.

If no ***username*** is given, the currend user's username will be used. The user will be prompted for the password of the destination keystore and the source keystore. If the destination keystore does not exist, one will be created and the user will be asked to enter the destination keystore password again for confirmation.

Flags

Item	Description
-S *servicename*	Specifies which service module to use.
-l *label*	Specifies the label associated with the key to be added.
-s *source_keystore*	Species the location of the source keystore.
-d *destination_keystore*	Specifies the location of the destination keystore.

Exit Status

Item	Description
0	The command completed successfully.

Item	Description
>0	An error occurred.

Security

This is a **setuid** command. In order to list the contents of a keystore the user must know the password of the private keystore.

Root and invokers belonging to group security are allowed to list anybody's keystore. However, they can only successfully complete this operation if they know the password to the keystore. A non-privileged user is only allowed to list the keystore that he owns.

Audit

This command records the following event information:

KEY_Add <*username*>

Examples

To copy a keystore object labeled as `label` from **/var/pki/security/keys/src.keystore** to **/var/pki/security/keys/dst.keystore**, enter:

```
$ keyadd -s /var/pki/security/keys/src.keystore -d /var/pki/
security/keys/dst.keystore -l label pkitest
```

Files

/usr/lib/security/pki/policy.cfg

/usr/lib/security/pki/ca.cfg

keycomp Command

Purpose

Compiles a keyboard mapping file into an input method keymap file.

Syntax

keycomp <*Infile* **>***Outfile*

Description

The **keycomp** command reads a textual description of the keyboard from standard input and produces a binary file that maps the keys to standard output. The binary file is used by the Input Method to translate key strokes into character strings.

You can *bind* characters and strings to keys on a keyboard with specified combinations of *modifier keys* called keyboard states, or you can specify particular key and state combinations as unbound (return nothing). All input keys are represented by *keysyms*, which stand for the key symbols that are usually used in the AIXwindows environment to represent keyboard input.

Any combination of modifier keys is possible when you press a key on the keyboard, but usually the keys are mapped into a smaller set of states. This state mapping can be specified.

Keycomp Source File

The input file used by the **keycomp** command consists of one or more lines. The items on the line are separated by a space. Each line begins with a keysym or a hexadecimal value for a keysym. The hexadecimal value represents keyboard input in the AIXwindows environment. Items following the

keysym represent the binding for a particular combination of the Ctrl, Alt, Shift, Lock, and Alt Graphic keys.

An item can be one of the following:

- Character surrounded by single quotation marks
- String surrounded by double quotation marks
- Keysym allowing mapping to other keysyms
- **U** indicating that the entry is unbound

Hexadecimal (**\x***XX*), octal (**\o***OOO*), and decimal (**\d***DDD*) notations of a byte can be contained in character and string items.

Keyboard States

Modifier keys (Shift, Lock, Ctrl, Alt, and Alt Graphics keys) change the state of the keyboard. They are used to select one item from a line corresponding to the input keysym. A value that is a combination of bits, each bit corresponding to a modifier key, indicates the state of a keyboard. The modifier keys increase in significance in the following order: Shift, Lock, Ctrl, Alt, and Alt Graphic modifier keys.

The bit combination or state value of a keyboard is mapped to one item of a line. The mapping is defined by the line beginning with the %M control, which can contain only numbers. The first number after the %M control is the item number. The numbers that follow the first number represent keyboard states, and they are all mapped to the item. See “Examples” on page 1823.

Flags

Item	Description
<*InFile*	Specifies a source file to be compiled by the **keycomp** command.
>*OutFile*	Specifies the name of the keymap file to be created.

Examples

1. The following is an example of a line for XK_a keysym input:

   ```
   XK_a'a'  XK_A  XK_A  XK_a  '\x01'  U  "hello"
   ```

   A , (comma) can, but need not, follow each item. Regardless of whether a comma follows an item, a space or tab must separate the items.

   Blank lines and lines beginning with the # character, except control statements, are ignored. All text between the # and the following line is ignored unless the # is part of a string enclosed in single or double quotation marks. Therefore, you can place comments at the end of a line that contains only a single item.
2. The following line shows that the keyboard states Ctrl, Ctrl+Shift, and Ctrl+Shift+Lock are all mapped to the third item:

   ```
   %M 3      4 5 7
   ```

Files

Item	Description
/usr/include/x11/keysymdef.h	Contains standard keysym definitions.
/usr/include/x11/aix_keysym.h	Contains unique keysym definitions.
/usr/bin/keycomp	Contains the **keycomp** command.
/usr/lib/nls/loc/*.imkeymap.src	Contains imkeymap source information.

Item	Description
/usr/lib/nls/loc/*.imkeymap	Maps a keysym/modifier to a string.

keydelete Command

Purpose

Deletes an object (key, certificate, etc) identified by the label from a keystore. If the label is ALL, all objects are deleted.

Syntax

keydelete [**-S** *ServiceName*] **-l** *Label* [**-p** *PrivateKeystore*] [*UserName*]

Description

The **keydelete** command deletes an object (key, certificate, etc) identified by the *Label*. If the *Label* is ALL, all objects are deleted. The **-S** flag specifies which end-entity services and libraries to use while deleting the objects from the keystore. Available services are defined in **/usr/lib/security/pki/ca.cfg**. When invoked without **-S**, **keydelete** uses the default service, which is **local**. An error is returned if a *ServiceName* is specified which does not have an entry in the **/usr/ lib/security/pki/ca.cfg** file.

The **-l** flag must be specified. The *Label* is a variable length text string that is used to map a key in the keystore to the certificate which contains the matching public key. If the *Label* is ALL, all the objects in the keystore are deleted.

If the **-p** flag is not given, the username's default keystore file is used. The user's default keystore location is **/var/pki/security/keys/**<*UserName*>.

If no *UserName* is given, the current user's user name is used. The user is prompted for the password of the keystore.

Flags

Item	Description
-S *ServiceName*	Specifies which service module to use.
-l *Label*	Specifies the label associated with the key to be added.
-p *PrivateKeystore*	Species the location of the source destination keystore.

Arguments

username - Specifies the user whose key is going to be deleted.

Security

This is a privileged (set-UID root) command.

In order to list the contents of a keystore, the user must know the password of the private keystore.

root and invokers belonging to group security are allowed to list anybody's keystore. However, they can only successfully complete this operation if they know the password to the keystore. A non-privileged user is only allowed to list the keystore that he owns.

Audit

This command records the following event information:

KEY_Delete <*UserName*>

Examples

1. To delete a keystore object with a label **signcert** from the invoker's default keystore, type:

```
keydelete -l signcert
```

2. To delete all the objects from the invoker's default keystore, type:

```
keydelete -l ALL
```

3. To delete a keystore object with a label **signcert** from the keystore **/home/bob/ bob.keystore**, type:

```
keydelete -p /home/bob/bob.keystore -l signcert
```

Files

/usr/lib/security/pki/ca.cfg

keyenvoy Command

Purpose

Acts as an intermediary between user processes and the **keyserv** daemon.

Syntax

/usr/sbin/keyenvoy

Description

The **keyenvoy** command acts as an intermediary by some Remote Procedure Call (RPC) programs between their user processes and the **keyserv** daemon. An intermediary is necessary because the **keyserv** daemon talks only to root processes. This program cannot be run interactively.

Files

Item	Description
/usr/sbin/keyenvoy	Contains the **keyenvoy** command.

keylist Command

Purpose

keylist lists the keystore labels in a private keystore.

Syntax

keylist [**-S** *servicename*] [**-v** | **-c**] [**-p** *privatekeystore*] [*username*]

Description

The **keylist** command lists the keystore labels in a private keystore. The **-S** option specifies which end-entity services and libraries to use while listing the labels in the keystore. Available services are defined in **/usr/lib/security/pki/ca.cfg**. When invoked without **-S**, **keylist** will use the default service, which is **local**. It is an error to specify a servicename which does not have an entry in the **/usr/lib/security/pki/ca.cfg** file. The user optionally may provide the location of the private keystore. If not given, the default

location will be used. If the **-c** option is given, the type of the keystore object corresponding to the label will be specified by one letter symbol. The following are the symbols denoting the keystore object types:

P = Public Key

p = Private Key

T = Trusted Key

S = Secret Key

C = Certificate

t = Trusted Certificate

U = Useful Certificate

If the **-v** option is used, the type of the object for a label will be given in non-abbreviated version (for example, Public Key, Secret Key).

If required, the user will be prompted for the password of the underlying service keystore.

Flags

Item	Description
-S *servicename*	Specifies which service module to use.
-p *privatekeystore*	Specifies the location of the keystore.
-v	Specifies that the output is in verbose mode.
-c	Specifies a concise output.

Arguments

Item	Description
username	Specifies the AIX user whose key labels is going to be queried.

Exit Status

Item	Description
0	Successful completion.
>0	An error occured.

Security

This is a privileged (set-UID root) command.

In order to list the contents of a keystore the user must know the password of the private keystore.

Root and invokers belonging to group security are allowed to list anybody's keystore. However, they can only successfully complete this operation if they have the knowledge of the password to the keystore.

A non-privileged user is only allowed to list the keystore that he owns.

Audit

This command records the following event information:

KEY_List <*username*>

Examples

1. To list the labels in keystore **/var/security/pki/keys/bob**, enter:

```
$ keylist -c -p /var/pki/security/keys/bob bob
PpC label1
PpC label2
```

2. To list labels/objects in verbose mode, enter:

```
$ keylist -v -p /var/pki/security/keys/bob bob
```

Files

/usr/lib/security/pki/policy.cfg

/usr/lib/security/pki/ca.cfg

keylogin Command

Purpose

Decrypts and stores the user's secret key.

Syntax

keylogin [**-r**]

Description

The **keylogin** command prompts users for their passwords. Then, the **keylogin** program decrypts the user's secret key, which is stored in the **/etc/publickey** file. The decrypted key is then stored by the local **keyserv** daemon to be used by any secure Remote Procedure Call (RPC) service, such as the Network File System (NFS).

The decrypted key given to the local **keyserv** daemon may eventually reach a time out and become invalid for that particular login session. The user can use the **keylogin** command again to refresh the key held by the **keyserv** daemon.

Flags

Item	Description
-r	Writes unencrypted secret key into a key file. Use the **-r** flag to store the root user's key in **/etc/.rootkey** on a host. Using this command, processes can run as a superuser task to issue authenticated requests. Therefore, processes do not need to explicitly run the **keylogin** command as a superuser task at system startup time.

Files

Item	Description
/etc/publickey	Contains public or secret keys for NIS maps.

keypasswd Command

Purpose

keypasswd manages the passwords which are used to access a user's private keystore.

Syntax

keypasswd [**-S** *servicename*] [**-p** *privatekeystore* | **-k** *username*]

Description

The **keypasswd** command allows a user to change the password of a private keystore. The user will be asked to enter the old and new password of the keystore. The **-S** option specifies which end-entity services and libraries to use while changing the password. Available services are defined in the **/usr/lib/security/pki/ca.cfg** file. When invoked without **-S**, **keypasswd** will use the **local** service. You will get an error if you specify a servicename which does not have an entry in the **/usr/lib/security/pki/ca.cfg** file. The **-p** option specifies the private keystore for which the password is going to be changed. The **-k** option specifies the user's default private keystore. You will get an error if you specify both the **-k** and **-p** options.

Flags

Item	Description
-S *servicename*	Specifies which service module to use.
-p *privatekeystore*	Specifies the private keystore whose password is going to be changed.
-k	Specifies that the keystore to be used is that of username.

Security

This is a privileged (set-UID root) command.

To change the password of a keystore one must know the password of the keystore.

Root and invokers belonging to group security are allowed to change the password of any keystore as long as they know the password of the keystore. A non-privileged user is allowed to change only the keystore file that they own.

Audit

This command records the following event information:

KEY_Password <*username*>

Examples

1. To change the password of the default private keystore that is owned by Bob, enter:

   ```
   $ keypasswd
   ```

   where the invoker is Bob.
2. To change the password of any other private keystore, enter:

   ```
   $ keypasswd -p bob.keystore
   ```

Files

/usr/lib/security/ca.cfg

/usr/lib/security/policy.cfg

keyserv Daemon

Purpose

Stores public and private keys.

Syntax

/usr/sbin/keyserv [**-n**]

Description

The **keyserv** daemon stores the private encryption keys of each user logged into the system. When a user types in a password during a **keylogin**, the secret key is decrypted. The decrypted key is then stored by the **keyserv** daemon. These decrypted keys enable the user to access secure network services such as secure Network File System (NFS).

When the **keyserv** daemon starts, it reads the key for the root directory from the **/etc/.rootkey** file. This daemon keeps the secure network services operating normally. For instance, after a power failure, when the system restarts itself, it gets the key for the root directory from the **/etc/.rootkey** file.

Flags

Item	Description
-n	Prevents the **keyserv** daemon from reading the key for the root directory from the **/etc/.rootkey** file. Instead, the **keyserv** daemon prompts the user for the password to decrypt the root directory's key stored in the network information service map and then stores the decrypted key in the **/etc/.rootkey** file for future use. This option is useful if the **/etc/.rootkey** file ever goes out of date or is corrupted.

Examples

1. To start the **keyserv** daemon enabling the system to get the key for the root directory from the **/etc/.rootkey** file, enter:

   ```
   /usr/sbin/keyserv
   ```

2. A System Resource Controller (SRC) command can also enable the system to get the key for the root directory from the **/etc/.rootkey** file as follows:

   ```
   startsrc -s keyserv
   ```

   This command sequence starts a script that contains the keyserv daemon.

3. To prevent the **keyserv** daemon from reading the key for the root directory from the **/etc/rootkey** file, enter:

   ```
   chssys -s keyserv -a '-n'
   ```

   This command passes the **-n** argument to the **keyserv** daemon if SRC is used to start the daemon.

Files

Item	Description
/etc/.rootkey	Stores the encrypted key for the root directory.

keysvrmgr Command

Purpose

Manages the Object Data Manager (ODM) database entries that are associated with the encryption key server when the logical volume uses the key server key-protection method for encryption.

Syntax

```
keysvrmgr action [-h] [flags]
```

Description

An encryption key server is used to securely store encryption key information. The access to the encryption key server is secured by certificate exchanges between the client and the server. When a logical volume (LV) uses the key server key-protection method for encryption, the information about the encryption key server is stored in the ODM database. You can use the **keysvrmgr** command to manage the ODM database entries that are associated with the encryption key server.

Starting from IBM AIX 7.2 with Technology Level 5, you can run the **keysvrmger** command by specifying the *action* parameter to perform one of the following operations:

- **add**: Adds a key server entry
- **modify**: Modifies an existing key server entry
- **remove**: Removes a key server entry
- **show**: Displays information about the key server entry

action parameters

add

Syntax:

```
keysvrmgr add [-h] -i server_ip [-p server_port] [-g sklm_device_group] -s server_cert_path
-c client_cert_path [-P type] server_id
```

Adds a key server entry to the ODM database. This *action* parameter can be specified with the following flags:

-i
Specifies the IP address of the encryption key server in the following format:

```
a.b.c.d
```

where each value of *a*, *b*, *c*, and *d* are in the range 0 - 255.

-p
(Optional) Specifies the port of the encryption key server. You can specify a port value in the range 0 – 65535. The default value is 5696.

-g
(Optional) Specifies the device group name associated with IBM Security Key Lifecycle Manager.

-s
Specifies the absolute path to the X.509 server certificate associated with the encryption key server.

-c
Specifies the absolute path of the Public Key Cryptography Standards #12 (PKCS #12) client certificate associated with your system.

-P

Specifies the type of password protection for the client certificate. You can specify the following values for this flag:

- **y**|**Y** – The password of the client certificate will be prompted during the command run time.
- **n**|**N** – The client certificate is not protected by a password. This is the default value.
- **p**|**P** – The password of the client certificate is stored in platform keystore (PKS).

server_id

Specifies the ID of the encryption key server entry that you want to create in the following format:

```
server_name[:device_group]
```

where *server_name* is the name of the key server entry and *device_group* is the name of the device group associated with IBM Security Key Lifecycle Manager.

modify

Syntax:

```
keysvrmgr modify [-h] -i server_ip [-p server_port] [-s server_cert_path] [-c
client_cert_path] [-P type] server_id
```

Modifies an existing key server entry in the ODM database. This *action* parameter can be specified with the following flags and values:

-i

Specifies the IP address of the encryption key server in the following format:

```
a.b.c.d
```

where each value of *a*, *b*, *c*, and *d* are in the range 0 - 255.

-p

(Optional) Specifies the port of the encryption key server. You can specify a port value in the range 0 – 65535. The default value is 5696.

-s

Specifies the absolute path to the X.509 server certificate associated with the encryption key server.

-c

Specifies the absolute path of the PKCS #12 client certificate associated with your system.

-P

Specifies the type of password protection for the client certificate. You can specify the following values for this flag:

- **y**|**Y** – The password of the client certificate will be prompted during the command run time.
- **n**|**N** – The client certificate is not protected by a password. This is the default value.
- **p**|**P** – The password of the client certificate is stored in platform keystore (PKS).

server_id

Specifies the ID of the key server entry that you want to modify in the following format:

```
server_name[:device_group]
```

where *server_name* is the name of the encryption key server and *device_group* is the name of the device group associated with IBM Security Key Lifecycle Manager.

remove

Syntax:

```
keysvrmgr remove [-h] server_id
```

Removes a key server entry from the ODM database. You must specify the ID of the key server entry that you want to remove from the ODM database.

show
Syntax:

```
keysvrmgr show [-h] server_id
```

Displays information about the specified key server ID.

Examples

- To display information about the existing key server entries in the ODM database, run the following command:

```
# keysvrmgr show
List of key servers:
ID                          PWD     IP:PORT
sklm1                       Y       10.11.12.13:5696
sklm_server2                N       210.211.212.213:569
```

Files

/usr/sbin/keysvrmgr
Contains the **keysvrmgr** command.

kill Command

Purpose

Sends a signal to running processes.

Syntax

To Send Signal to Processes

kill [**-s** { *SignalName* | *SignalNumber* }] *ProcessID ...*

kill [**-** *SignalName* | **-** *SignalNumber*] *ProcessID ...*

To List Signal Names

kill -l [*ExitStatus*]

Description

The **kill** command sends a signal (by default, the **SIGTERM** signal) to a running process. This default action normally stops processes. If you want to stop a process, specify the process ID (PID) in the *ProcessID* variable. The shell reports the PID of each process that is running in the background (unless you start more than one process in a pipeline, in which case the shell reports the number of the last process). You can also use the **ps** command to find the process ID number of commands.

A root user can stop any process with the **kill** command. If you are not a root user, you must have initiated the process that you want to stop.

SignalName is recognized in a case-independent fashion, without the SIG prefix.

If the specified *SignalNumber* is 0, the **kill** command checks the validity of the specified PID.

Flags

Item	Description
-s{*SignalName* **|** *SignalNumber***}**	Specifies the signal as a signal number or a signal name, such as -9 or **KILL** for the **SIGKILL** signal.
-*SignalName*	Specifies a signal name, such as **HUP**.
-*SignalNumber*	Specifies a signal number. **Note:** To specify the negative PID with the default signal in this syntax, you must specify **- -** as a signal. Otherwise the first operand is interpreted as a *SignalNumber*.
ProcessID	Specifies a decimal integer that represents a process or process group to be signaled. If PID is a positive value, the **kill** command sends the process whose process ID is equal to the PID. If the PID value is 0, the **kill** command sends the signal to all processes that have a process group ID equal to the process group ID of the sender. The signal is not sent to processes with a PID of 0 or 1. If the PID is -1, the **kill** command sends the signal to all processes owned by the effective user of the sender. The signal is not sent to processes with a PID of 0 or 1. If it is a negative number but not -1, the **kill** command sends the signal to all processes that have a process group ID equal to the absolute value of the PID.
-l	Lists all signal names that are supported by the implementation
-l*ExitStatus*	Lists signal names that are stripped of the common SIG prefix. If *ExitStatus* is a decimal integer value, the signal name corresponding to that signal is displayed. If *ExitStatus* is a value of the exit status corresponding to a process that was terminated by a signal, the signal name corresponding to the signal that terminated the process is displayed.

Exit Status

This command returns the following exit values:

Item	Description
0	At least one matching process was found for each *ProcessID* operand, and the specified signal was successfully processed for at least one matching process.
>0	An error occurred.

Examples

1. To stop a given process, enter the following command:

   ```
   kill 1095
   ```

   This stops process 1095 by sending it the default **SIGTERM** signal. Note that process 1095 might not actually stop if it has made special arrangements to ignore or override the **SIGTERM** signal.
2. To stop several processes that ignore the default signal, enter the following command:

   ```
   kill -kill 2098 1569
   ```

This sends signal 9, the **SIGKILL** signal, to processes 2098 and 1569. The **SIGKILL** signal is a special signal that normally cannot be ignored or overridden.

3. To stop all of your processes and log yourself off, enter the following command:

```
kill -kill 0
```

This sends signal 9, the **SIGKILL** signal, to all processes that have a process group ID equal to the senders process group ID. Because the shell cannot ignore the **SIGKILL** signal, this command also stops the login shell and logs you off.

4. To stop all processes that you own, enter the following command:

```
kill -9 -1
```

This command sends signal 9, the **SIGKILL** signal, to all processes that are owned by the effective user, even those processes that are started at other workstations and that belong to other process groups. If a listing that you requested is being printed, it is also stopped.

5. To send a different signal code to a process, enter the following command:

```
kill  -USR1  1103
```

The name of the **kill** command is misleading because many signals, including **SIGUSR1**, do not stop processes. The action that is taken on **SIGUSR1** is defined by the particular application you are running.

> **Note:** To send signal 15, the **SIGTERM** signal with this form of the **kill** command, you must explicitly specify -15 or **TERM**.

Files

Item	Description
/usr/include/sys/signal.h	Specifies signal names.

killall Command

Purpose

Cancels all processes except the calling process.

Syntax

killall [**-**] [**-***Signal*]

Description

The **killall** command cancels all processes that you started, except those producing the **killall** process. This command provides a convenient means of canceling all processes created by the shell that you control. When started by a root user, the **killall** command cancels all cancellable processes except those processes that started it. If several Signals are specified, only the last one is effective.

If no signal is specified, the **killall** command sends a **SIGKILL** signal.

Flags

Item	Description
-	Sends a **SIGTERM** signal initially and then sends a **SIGKILL** signal to all processes that survive for 30 seconds after receipt of the signal first sent. This gives processes that catch the **SIGTERM** signal an opportunity to clean up. If both **-** and **-***Signal* are set, the **killall** command sends the specified signal initially and then sends a **SIGKILL** signal to all processes that survive for 30 seconds after receipt of the signal first sent.
-*Signal*	Sends the specified *Signal* number or *SignalName*.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To stop all background processes that have started, enter:

   ```
   killall
   ```

   This sends all background processes the **kill** signal 9 (also called the **SIGKILL** signal).
2. To stop all background processes, giving them a chance to clean up, enter:

   ```
   killall -
   ```

   This sends signal 15, the **SIGTERM** signal; waits 30 seconds, and then sends signal 9, the **SIGKILL** signal.
3. To send a specific signal to the background processes, enter:

   ```
   killall -2
   ```

   This sends signal 2, the **SIGINT** signal, to the background processes.

kinit Command

Note: The Kerberos ticket life time is calculated taking the DST changes into consideration, as per design. Hence Kerberos tickets issued during DST disabled time, if has validity that spans to reach the DST enabled time or vice versa can have a difference of 1 hour displayed in **klist**.

Purpose

Obtains or renews the Kerberos ticket-granting ticket.

Syntax

kinit [**-l** *lifetime*] [**-r** *renewable_life*] [**-f**] [**-p**] [**-A**] [**-s** *start_time*] [**-S** *target_service*] [**-k** [**-t** *keytab_file*]] [**-R**] [**-v**] [**-u**] [**-c** *cachename*] [principal]

Description

The **kinit** command obtains or renews a Kerberos ticket-granting ticket. The Key Distribution Center (KDC) options specified by the [kdcdefault] and [realms] in the Kerberos configuration file (**kdc.conf**) are used if you do not specify a ticket flag on the command line.

If you are not renewing an existing ticket, the command reinitializes the credentials cache and will contain the new ticket-granting ticket received from the KDC. If you do not specify the *Principal* name on the command line and you do specify the **-s** flag, the *Principal* name is obtained from the credentials cache. The new credentials cache becomes the default cache unless you specify the cache name using the **-c** flag.

The ticket *Time* value for the **-l**, **-r** and **-s** flags is expressed as *ndnhnmns* where:

n
represents a number

d
represents days

h
represents hours

m
represents minutes

s
represents seconds

You must specify the components in this order but you can omit any component, for example `4h5m` represents four hours and 5 minutes and `1d2s` represents 1 day and 2 seconds.

Flags

Flags Description

Item	Description
-A	Specifies that the ticket contain a list of client addresses. The ticket will contain the local host address list if this option is not specified. When an initial ticket contains an address list, it can be used only from one of the addresses in the the address list.
-c *cachename*	Specifies the name of the credentials cache to use. The default credentials cache is used if this flag is not specified. If the KRB5CCNAME environment variable is set, its value is used to name the default ticket cache. Any existing contents of the cache i are destroyed by **kinit**.
-f	Specifies that the ticket is to be forwardable. To forward the ticket, this flag must be specified.
-k	Specifies to obtain the key for the ticket principal from a key table. If you do not specify this flag, you are prompted to enter the password for the ticket principal.
-l *lifetime*	Specifies the ticket end time interval. The ticket cannot be used after the interval expires unless the ticket is renewed. The interval default time is 10 hours.
-p	Specifies that the ticket is to be proxiable. To make the ticket proxiable, this flag must be specified.
principal	Specifies the ticket principal. The principal is obtained from the credentials cache if the principal is not specified on the command line.
-r *renewable_life*	Specifies the renew time interval for a renewable ticket. The ticket cannot be renewed after the interval expires. The renew time must be greater than the end time. If this flag is not specified, the ticket is not renewable, although you can still generate a renewable ticket if the requested ticket lifetime exceeds the maximum ticket lifetime.
-R	Specifies to renew an existing ticket. No other flags may be specified when renewing an existing ticket.
-s *start_time*	Specifies a request for a postdated ticket, valid starting at *start_time*.

Flags Description *(continued)*

Item	Description
-S *target_service*	Specifies an alternate service name to use when getting initial tickets.
-t *keytab_file*	Specifies the key table name. The default key table is used if this flag is not specified and the **-k** flag is specified. The **-t** flag implies the **-k** flag.
-v	Specifies that the ticket granting ticket in the cache be passed to the kdc for validation. If the ticket is within its requested time range, the cache is replaced with the validated ticket.
-u	Specifies that the **kinit** command creates a credentials cache file that is unique to the process. If the **kinit** command is successful, the credentials cache file name includes a unique number (Process Authentication Group or PAG). In AIX Version 5.3 and later, the PAG is generated from an operating system service. The *KRB5CCNAME* environment variable is set to this credentials cache file, and the **kinit** command executes a new shell.

Examples

1. To obtain a ticket-granting ticket with a lifetime of 10 hours, which is renewable for five days, type:

```
kinit -l 10h -r 5d my_principal
```

2. To renew an existing ticket, type:

```
kinit -R
```

Files

Files

Item	Description
/usr/krb5/bin/kinit	-
/var/krb5/security/creds/krb5cc_[uid]	default credentials cache ([uid] is the UID of the user.)
/etc/krb5/krb5.keytab	default location for the local host's keytab file.
/var/krb5/krb5kdc/kdc.conf	Kerberos KDC configuration file.

klist Command

Purpose

Displays the contents of a Kerberos credentials cache or key table.

Syntax

klist [[**-c**] [**-f**] [**-e**] [**-s**] [**-a**] [**-n**]] [**-k** [**-t**] [**-K**]] [name]

Description

The **klist** command displays the contents of a Kerberos credentials cache or key table.

Flags

Flags Description

Item	Description
-a	Displays all tickets in the credentials cache, including expired tickets. Expired tickets are not listed if this flag is not specified. This flag is valid only when listing a credentials cache.
-c	Lists the tickets in a credentials cache. This is the default if neither the **-c** nor the **-k** flag is specified. This flag is mutually exclusive with the **-k** flag.
-e	Displays the encryption type for the session key and the ticket.
-f	Displays the ticket flags using the following abbreviations: **F** Forwardable ticket **f** Forwarded ticket **P** Proxiable ticket **p** Proxy ticket **D** Postdateable ticket **d** Postdated ticket **R** Renewable ticket **I** Initial ticket **i** Invalid ticket **H** Hardware preauthentication used **A** Preauthentication used **O** Server can be a delegate
name	Specifies the name of the credentials cache or key table. The default credentials cache or key table is used if you do not specify a filename. If you do not specify a name indicating a cache name or keytab name, klist displays the credentials in the default credentials cache or keytab file as appropriate. If the KRB5CCNAME environment variable is set, its value is used to name the default credentials (ticket) cache.
-k	Lists the entries in a key table. This flag is mutually exclusive with the **-c** flag.
-K	Displays the encryption key value for each key table entry. This flag is valid only when listing a key table.
-n	Displays the numerical internet address instead of the host name. The default without the **-n** is host name. This command is used in conjunction with the **-a** flag.
-s	Suppresses command output but sets the exit status to 0 if a valid ticket-granting ticket is found in the credentials cache. This flag is valid only when listing a credentials cache.

Flags Description *(continued)*

Item	Description
-t	Displays timestamps for key table entries. This flag is valid only when listing a key table.

Examples

1. To list all of the entries in the default credentials cache, type:

```
klist
```

2. To list all of the entries in the **etc/krb5/my_keytab** key table with timestamps, type:

```
klist -t -k etc/krb5/my_keytab
```

Files

Files

Item	Description
/usr/krb5/bin/klist	-
/var/krb5/security/creds/krb5cc_[uid]	default credentials cache ([uid] is the UID of the user.)
/etc/krb5/krb5.keytab	default location for the local host's keytab file.

kmodctrl Command

Purpose

Loads or unloads the kernel extension **/usr/lib/drivers/kmobip6**.

Syntax

kmodctrl [**-k** *kextname*] [**-luq**]

Description

The kernel extension **/usr/lib/drivers/kmobip6** contains support for the Mobile IPv6 functionality. This kernel extension must be loaded in order to configure the system as a mobile IPv6 home agent or correspondent node. Normally this command will be run automatically by the **/etc/rc.mobip6** script if mobile IPv6 has been enabled using system management.

Flags

Item	Description
-k	Specifies an alternate path for the mobility kernel extension.
-l	Loads the mobility kernel extension.
-q	Checks whether the kernel extension is loaded.
-u	Unloads the mobility kernel extension.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Security

You must be the root user or a member of the system group to execute this command.

Examples

1. The following example loads the kmobip6 kernel extension:

```
kmodctrl -l
```

2. The following example unloads the kmobip6 kernel extension. This will disable all mobile IPv6 functionality on the system:

```
kmodctrl -u
```

3. The following example queries whether the kmobip6 kernel extension is loaded:

```
kmodctrl -q
```

kpasswd Command

Purpose

Changes the password for a Kerberos principal.

Syntax

kpasswd [Principal]

Description

The **kpasswd** command changes the password for a specified Kerberos principal. It prompts for the current principals password, which is used to obtain a changepw ticket from the KDC for the user's Kerberos realm. If **kpasswd** successfully obtains the changepw ticket, the user is prompted twice for the new password and the password is changed.

If the principal is governed by a policy that specifies for example length and/or number of character classes required in the new password, the new password must conform to the policy.

You may not change the password for a ticket-granting service principal (krbtgt/domain) using the **kpasswd** command.

Parameters

Parameters

Item	Description
Principal	Specifies the principal for which password you want to change. If you do not specify the principal on the command line, the principal is obtained from the default credentials cache.

Security

When requesting a password change, you must supply both the current password and the new password.

Files

Files

Item	Description
/usr/krb5/bin/kpasswd	-
/var/krb5/security/creds/krb5cc_[uid]	default credentials cache ([uid] is the UID of the user.)

krlogind Daemon

Purpose

Provides the server function for the **rlogin** command.

Syntax

/usr/sbin/krlogind [-n] [-s]

> **Note:** The **krlogind** daemon is normally started by the **inetd** daemon. It can also be controlled from the command line, using SRC commands.

Description

The **/usr/sbin/krlogind** daemon is the server for the **rlogin** remote login command. The server provides a remote login facility.

Changes to the **krlogind** daemon can be made by using the System Management Interface Tool (SMIT) or System Resource Controller (SRC), by editing the **/etc/inetd.conf** or **/etc/services** file. Entering **krlogind** at the command line is not recommended. The **krlogind** daemon is started by default when it is uncommented in the **/etc/inetd.conf** file.

The **inetd** daemon get its information from the **/etc/inetd.conf** file and the **/etc/services** file.

After changing the **/etc/inetd.conf** or **/etc/services** file, run the **refresh -s inetd** or **kill -1 InetdPID** command to inform the **inetd** daemon of the changes to its configuration file.

Service Request Protocol

When the **krlogind** daemon receives a service request, the daemon initiates the following protocol:

1. The **krlogind** daemon checks the source port number for the request. If the port number is not in the range 512 through 1023, the **krlogind** daemon terminates the connection.
2. The **krlogind** daemon uses the source address of the initial connection request to determine the name of the client host. If the name cannot be determined, the **krlogind** daemon uses the dotted-decimal representation of the client host address.
3. The **krshd** daemon attempts to validate the user using the following steps:
   - makes sure that Kerberos 5 is a valid authentication method if the incoming ticket is a Kerberos 5 ticket. If the incoming ticket is a Kerberos 4 ticket, the connection fails. Kerberos 4 is not supported for **rlogin**.
   - calls **kvalid_user** with the local account name as well as the DCE principal.

Error Messages

The following error messages are associated with the **krlogind** daemon:

Item	Description
Try again	A fork command made by the server has failed.
/usr/bin/shell:	No shell. The shell specified for the shell variable cannot be started. The shell variable may also be a program.

Flags

Item	Description
-n	Disables transport-level keep-alive messages. The messages are enabled by default.
-s	Turns on socket level debugging.

Manipulating the krshd Daemon

The **krshd** daemon is a subserver of the **inetd** daemon, which is a subsystem of the System Resource Controller (SRC). The **krshd** daemon is a member of the tcpip SRC subsystem group. Using the **chauthent** command will comment/uncomment the kshell line in the **/etc/inetd.conf** file and restart the **inetd** daemon depending on whether Kerberos 5 or Kerberos 4 is configured/unconfigured. This daemon should be manipulated using the **chauthent**/**lsauthent** commands. Direct modification of the **inetd.conf** file's kshell entry in not recommended.

krshd Daemon

Purpose

Provides the server function for remote command execution.

Syntax

/usr/sbin/krshd

> **Note:** The **rshd** daemon is normally started by the **inetd** daemon. It can also be controlled from the command line, using SRC commands.

Description

The **/usr/sbin/krshd** daemon is the server for the **rcp** and **rsh** commands using Kerberos authentication. The **krshd** daemon provides remote execution of shell commands. These commands are based on requests from privileged sockets on trusted hosts. The shell commands must have user authentication. The **krshd** daemon listens at the kshell socket defined in the **/etc/services** file.

Changes to the **krshd** daemon can be made using the System Management Interface Tool (SMIT) or System Resource Controller (SRC), by editing the **/etc/inetd.conf** or **/etc/services** file. Entering **krshd** at the command line is not recommended. The **krshd** daemon is started by default when it is uncommented in the **/etc/inetd.conf** file.

The **inetd** daemon gets its information from the **/etc/inetd.conf** file and the **/etc/services** file.

After changing the **/etc/inetd.conf** or **/etc/services** file, run the **refresh -s inetd** or **kill 1 InetdPID** command to inform the **inetd** daemon of the changes to its configuration file.

Service Request Protocol

When the **krshd** daemon receives a service request, it initiates the following protocol:

1. The **krshd** daemon checks the source port number for the request. If the port number is not in the range 0 through 1023, the **krshd** daemon terminates the connection.
2. The **krshd** daemon reads characters from the socket up to a null byte. The string read is interpreted as an ASCII number (base 10). If this number is nonzero, the **krshd** daemon interprets it as the port number of a secondary stream to be used as standard error. A second connection is created to the specified port on the client host. The source port on the local host is also in the range 0 through 1023.
3. The **krshd** daemon uses the source address of the initial connection request to determine the name of the client host. If the name cannot be determined, the **krshd** daemon uses the dotted decimal representation of the client host's address.
4. The **krshd** daemon retrieves the following information from the initial socket:
   - A Kerberos service ticket.
   - A null-terminated string of at most 16 bytes interpreted as the user name of the user on the client host.
   - Another null-terminated string interpreted as a command line to be passed to a shell on the local server host.
   - A null-terminated string of at most 16 bytes interpreted as the user name to be used on the local server host.
   - If the service ticket was a Kerberos 5 ticket, the daemon will expect either a Kerberos 5 TGT or a null string.
5. The krshd daemon attempts to validate the user using the following steps:
   - makes sure that Kerberos 5 is a valid authentication method if the incoming ticket is a Kerberos 5 ticket. Likewise, if the incoming ticket is a Kerberos 4 ticket, the Kerberos 4 authentication method must be configured.
   - calls **kvalid_user** with the local account name as well as the DCE Principal.
6. Once **krshd** validates the user, the **krshd** daemon returns a null byte on the initial connection. If the connection is a Kerberos 5 ticket and the TGT is sent, the command line passes to the **k5dcelogin** command, (which upgrades it to full DCE credentials). If the TGT is not sent or if the connection is a Kerberos 4 ticket, the command line passes to the user's local login shell. The shell then inherits the network connections established by the **krshd** daemon.

   The **krshd** daemon is controlled by using the System Management Interface Tool (SMIT) or by changing the **/etc/inetd.conf** file. Entering **krshd** at the command line is not recommended.

Manipulating the krshd Daemon

The **krshd** daemon is a subserver of the **inetd** daemon, which is a subsystem of the System Resource Controller (SRC). The **krshd** daemon is a member of the tcpip SRC subsystem group. Using the **chauthent** command will comment/uncomment the kshell line in the **/etc/inetd.conf** file and restart the **inetd** daemon depending on whether Kerberos 5 or Kerberos 4 is configured/unconfigured. This daemon should be manipulated using the **chauthent**/**lsauthent** commands. Direct modification of the **inetd.conf** file's kshell entry in not recommended.

ksh Command

Purpose

Invokes the Korn shell.

Syntax

ksh [**-i**] [{ **+** | **-** } { **a e f h k m n p t u v x** }] [**-o** *Option ...*] [**-c** *String* | **-s** | **-r** | *File* [*Parameter*]]

Note: Preceding a flag with **+** (plus) rather than **-** (minus) turns off the flag.

Description

The **ksh** command invokes the Korn shell, which is an interactive command interpreter and a command programming language. The shell carries out commands either interactively from a terminal keyboard or from a file.

The Korn shell is backwardly compatible with the Bourne shell (invoked with the **bsh** command) and contains most of the Bourne shell features as well as several of the best features of the C shell.

Note: The **ksh** wait built in behaves in a manner similar to the **parent wait()** API.

An enhanced version of the Korn shell, called `ksh93`, is also available. The enhanced Korn shell has additional features that are not available in the default Korn shell.

Additionally, a restricted version of the Korn shell, called `rksh`, is available. The restricted Korn shell allows administrators to provide a controlled execution environment for the users.

Flags

Item	Description
-a	Exports automatically all subsequent parameters that are defined.
-c *String*	Causes the Korn shell to read commands from the *String* variable. This flag cannot be used with the **-s** flag or with the *File*[*Parameter*] parameter.
-e	Executes the **ERR** trap, if set, and exits if a command has a nonzero exit status, unless in the following conditions: • The simple command is contained in a "&&" or "||" list. • The simple command immediately follows "if", "while" or "until". • The simple command is contained in a pipeline following "!". This mode is disabled when profiles are read.
-f	Disables file name substitution.
-h	Designates each command as a tracked alias when first encountered.
-i	Indicates that the shell is interactive. An interactive shell is also indicated if shell input and output are attached to a terminal (as determined by the **ioctl** subroutine). In this case, the **TERM** environment variable is ignored (so that the **kill 0** command does not kill an interactive shell) and the **INTR** signal is caught and ignored (so that a wait state can be interrupted). In all cases, the **QUIT** signal is ignored by the shell.
-k	Places all parameter assignment arguments in the environment for a command, not just those arguments that precede the command name.
-m	Runs background jobs in a separate process and prints a line upon completion. The exit status of background jobs is reported in a completion message. On systems with job control, this flag is turned on automatically for interactive shells.
-n	Reads commands and checks them for syntax errors, but does not execute them. This flag is ignored for interactive shells.

Item	Description
-o *Option*	Prints the current option settings and an error message if you do not specify an argument. You can use this flag to enable any of the following options: **allexport** Same as the **-a** flag. **errexit** Same as the **-e** flag. **bgnice** Runs all background jobs at a lower priority. This is the default mode. **emacs** Enters an emacs-style inline editor for command entry. **gmacs** Enters a gmacs-style inline editor for command entry. **ignoreeof** Does not exit the shell when it encounters an end-of-file character. You must use the **exit** command, or override the flag and exit the shell by pressing the Ctrl-D key sequence more than 11 times. **keyword** Same as the **-k** flag. **markdirs** Appends a **/** (slash) to all directory names that are a result of filename substitution. **monitor** Same as the **-m** flag. **noclobber** Prevents redirection from truncating existing files. When you specify this option, use the redirection symbol **>|** (right caret, pipe symbol) to truncate a file. **noexec** Same as the **-n** flag. **noglob** Same as the **-f** flag. **nolog** Prevents function definitions from being saved in the history file. **nounset** Same as the **-u** flag. **privileged** Same as the **-p** flag. **verbose** Same as the **-v** flag. **trackall** Same as the **-h** flag. **vi** Enters the insert mode of a vi-style inline editor for command entry. Entering escape character 033 puts the editor into the move mode. A return sends the line. **viraw** Processes each character as it is typed in vi mode. **xtrace** Same as the **-x** flag. You can set more than one option on a single **ksh** command line.

Item	Description
-p	Disables the processing of the **$HOME/.profile** file when you use the shell as a login shell.
-r	Runs a restricted shell. With a restricted shell you cannot: • Change the current working directory. • Set the value of the **SHELL**, **ENV**, or **PATH** variable. • Specify the pathname of a command that contains a **/** (slash). • Redirect output of a command with **>** (right caret), **>|** (right caret, pipe symbol), **<>** (left caret, right caret), or **>>** (two right carets). Using this flag is the same as issuing the `rksh` command.
-s	Causes the **ksh** command to read commands from the standard input. Shell output, except for the output of the special commands, is written to file descriptor 2. This parameter cannot be used with the **-c** flag or with the *File*[*Parameter*] parameter.
-t	Exits after reading and executing one command.
-u	Treats unset parameters as errors when substituting.
-v	Prints shell input lines as they are read.
-x	Prints executed commands and their arguments.

Files

Item	Description
/usr/bin/ksh	Contains the path name to the Korn shell.
/tmp/sh*	Contains temporary files that are created when a shell is opened.

ksh93 Command

Purpose

Invokes the Enhanced Korn shell.

Syntax

ksh93 [+ | - a b c C e f h i k m n **o** p r s t u v x **BDP**] [+-R *file*] [+-o *Option*] [*arg*...].

Note: Preceding a flag with + (plus) rather than - (minus) turns off the flag.

Description

The ksh93 command invokes the Enhanced Korn shell, which is an interactive command interpreter and a command programming language. The shell carries out commands either interactively from a terminal keyboard or from a file.

The Enhanced Korn shell has additional features that are not available in the default Korn shell.

Note: The ksh93 built-in wait behaves in a manner similar to the parent `wait` subroutine.

Flags

Item	Description
-B	Enables brace pattern field generation, and brace group expansion. This is set by default.
-D	Does not execute the script, but results in the set of output strings in double quotation marks preceded by $. These strings are needed for localization of the script in different languages.
-P	If the **-P** or **-o** profiles are present, the shell is called a profile shell.
-a	Exports automatically all subsequent parameters that are defined.
-b	Job completion messages are printed as soon as a background job changes state rather than waiting for the next prompt.
-c	Causes commands to be read from the first argument. The remaining arguments become positional parameters starting from 0.
-C	Prevents existing files from getting truncated when redirection > is used. O_EXCL mode is used to create files. Requires >| to truncate a file when -C option is used.
-e	If not contained within a **||** or **&&** command, or following an **if while** or **until** command, or in the pipeline following **!**, executes the ERR trap, if set, and exits if a command has a nonzero exit status. This mode is disabled while reading profiles.
-f	Disables file name generation.
-h	Designates each command as a tracked alias when first encountered. **Note:** The tracked alias feature is now obsolete.
-i	Indicates that the shell is interactive. An interactive shell is also indicated if shell input and output are attached to a terminal (as determined by the ioctl subroutine). In this case, the TERM environment variable is ignored (so that the kill 0 command does not kill an interactive shell) and the INTR signal is caught and ignored (so that a wait state can be interrupted). In all cases, the QUIT signal is ignored by the shell.
-k	(obsolete) Places all parameter assignment arguments in the environment for a command, not just those arguments that precede the command name.
-m	Runs background jobs in a separate process and prints a line upon completion. The exit status of background jobs is reported in a completion message. On systems with job control, this flag is turned on automatically for interactive shells.
-n	Reads commands and checks them for syntax errors, but does not execute them. This flag is ignored for interactive shells. **Note:** ksh93 -n outputs a warning message for certain syntax. These messages are warnings. Even though these warnings are issued, the execution of the scripts is unaltered. The following are known warning messages: `...` obsolete, use $(...). -a obsolete, use -e. '=' obsolete, use '=='. %s within [[...]] obsolete, use ((...)). set %s obsolete. `{' instead of `in' is obsolete. "obsolete -j must be 1 or 2.

Item	Description
-o *Option*	Prints the current option settings and an error message if you do not specify an argument. You can use this flag to enable any of the following options: **allexport** Same as the -a flag. **errexit** Same as the -e flag. **bgnice** Runs all background jobs at a lower priority. This is the default mode. **Braceexpand** Same as the **-B** flag. **emacs** Enters an emacs-style inline editor for command entry. **gmacs** Enters a gmacs-style inline editor for command entry. **ignoreeof** Does not exit the shell when it encounters an end-of-file character. You must use the `exit` command, or override the flag and exit the shell by pressing the Ctrl-D key sequence more than 11 times. **interactive** Same as the -i flag. **keyword** Same as the -k flag. **markdirs** Appends a / (slash) to all directory names that are a result of filename substitution. **monitor** Same as the -m flag. **multiline** The built-in editor will use multiple lines of the screen that are longer than the width of the screen. This might not work on all terminals. **noclobber** Same as the -C flag. **noexec** Same as the -n flag. **noglob** Same as the -f flag. **nolog** Prevents function definitions from being saved in the history file.

Item	Description
	notify Same as the -b flag. **nounset** Same as the -u flag. **pipefail** A pipeline will not complete until all the components of the pipeline are complete. The return value will be that of the last non-zero return value of the last command to fail, or 0 if all return values are 0. **showme** Single commands or pipelines preceded by a ; (semicolon) will be displayed as if the xtrace option were enabled but will not be executed. Otherwise the leading ; (semicolon) will be ignored. **privileged** Same as the -p flag. **verbose** Same as the -v flag. **trackall** Same as the -h flag. **vi** Enters the insert mode of a vi-style inline editor for command entry. Entering escape character 033 puts the editor into the move mode. A return sends the line. **viraw** Processes each character as it is typed in vi mode. **xtrace** Same as the -x flag. You can set more than one option on a single ksh93 command line.
-p	Disables processing of the $HOME/.profile file and uses the /etc/suid_profile file instead of the ENV file. This mode is on whenever the effective uid (gid) is not equal to the real uid (gid). Turning this off causes the effective uid and gid to be set to the real uid and gid.
-r	Runs a restricted shell. With a restricted shell you cannot: • Change the current working directory. • Set the value of the SHELL, ENV, or PATH variable. • Specify the path name of a command that contains a / (slash). • Redirect output of a command with > (right caret), >| (right caret, pipe symbol), <> (left caret, right caret), or >> (two right carets).

Item	Description
-R *File*	A cross reference database is generated when the -R *File* option is used. This can be used to find definitions and references for variables and commands by a separate utility.
-s	Causes the ksh93 command to read commands from the standard input. Shell output, except for the output of the special commands, is written to file descriptor 2. This parameter cannot be used with the -c flag or with the *File*[*Parameter*] parameter.
-t	Exits after reading and executing one command.
-u	Treats unset parameters as errors when substituting.

Item	Description
`-v`	Prints shell input lines as they are read.
`-x`	Prints executed commands and their arguments.

Exit Status

Item	Description
`0`	Successful completion.
`>0`	An error occurred.

Location

`/usr/bin/ksh93`

kvno Command

Purpose

Displays the current key version number for a principal.

Syntax

kvno [**-e** *etype*] *service 1 service2*....

Description

The **kvno** command displays the current key version number for a principal (*service 1 service2*...). The security policy must allow a service ticket to be obtained for the principal. The current network identity is used when requesting the service ticket.

Flags

Item	Description
-e *etype*	Specifies which encryption *type* to get the current key version.
service 1 service2...	Specifies the principal for which you want to display the current key version number.

Security

The security policy must allow a service ticket to be obtained for the principal.

Files

/usr/krb5/bin/kvno

l

The following AIX commands begin with the letter *l*.

labcat Command

Purpose

Prints a process's Sensitivity Label (SL) on the banner, and at the top and bottom of each printed page.

Syntax

`labcat` *files*

`labcat` [`-P` *pagetype*] [`-U`] [`-p` *lines*] [`-c`] [`-f`] [*files*]

Description

The `labcat` command generates secure binary labels, in human-readable format, for the System V print subsystem with Trusted AIX installed using two modes of operation. This command uses the DIA label-encodings software to produce the labels that appear on the banner and pages.

In general, the `labcat` command parses each printer command sent to the printer. Those commands, which cannot corrupt internal page labeling or affect the permanent state of the printer, is passed through unaffected. Suspect commands are modified if possible, but the `labcat` command resets the printer and then exits with an error code if the use of a particular command is unacceptable, thereby aborting the print job. Thus, dangerous commands are not sent to the printer. The next printer reset (preceding the next print job) re-establishes the printer default state.

The `labcat` command examines the shell environment variable **TERM** (set by the `lpsched` command to indicate the printer type) to determine the language of the printer commands that it receives from standard input. A value of hplaserjet or hplaser implies PCL language (standard configuration); PS, PS-b, or PSR indicates PostScript language (postscript configuration).

Flags

Item	Description
`-c`	Adds a carriage return to the output.
`-f`	Indicates that standard input rather than a file is used.
`-p` *lines*	Indicates the number of lines of text per page.

Item	Description
-P *pagetype*	Determines the correct location for the labels to appear at the top and bottom of various size pages. If you do not specify the *pagetype* parameter, the page-type *letter* is assumed. In a postscript configuration, the value for *pagetype* is sent to the printer after the printer is reset, but before the print-job object is sent. In standard configuration, the *pagetype* parameter must be one of the following: • **executive** • **letter** • **legal** • **a4** • **monarcenvelope** • **coml0envelope** • **dlenvelope** • **c5envelope** The *pagetype* parameter is converted into horizontal and vertical positions that are used for the placement of the trusted labels.
-U	Specifies that pages are not to be labeled.
files	Specifies the files to be printed. You can print multiple files. Separate the files with blank spaces.

Exit Status

Item	Description
0	The command completed successfully.
-1	The command failed.

Error codes

The `labcat` command returns a failure under one of the following conditions:

- It cannot audit the event.
- The page length or page type is not valid.
- It cannot open the label-encodings file.
- It cannot open the file to be printed.

labck Command

Purpose

Checks for the consistency of the label-encodings file.

Syntax

labck [**-l**] [**-f** *encodings_file*]

labck [{ **-c** | **-r** } *encodings_file*]

Description

You can use the **labck** command to verify that a label-encodings file is internally consistent. When specified without any flags, the **labck** command verifies the consistency of the system-default, label-encodings file and no message is displayed if the file is proper.

Flags

Item	Description
-l	Lists the system high sensitivity label, system low sensitivity label, system high integrity label, and system low integrity label as defined in the label-encodings file.
-f *encodings_file*	Uses the value that you specify for *encodings_file* instead of the system-default, label-encodings file.
-c *encodings_file*	Copies the contents of the system-default, label-encodings file into the *encodings_file* that you specify. If the file already exists, the command exits with an error.
-r *encodings_file*	Replaces the contents of the system-default, label-encodings file with the contents in the file that you specify using *encodings_file*. The contents are replaced only if the file name that you specify is a valid label-encodings file.

Security

Only authorized users can run the **labck** command.

Item	Description
aix.mls.lef	Required to perform the above operations on the label encodings file.

Files Accessed:

Item	Description
Mode	**File**
r	**/etc/security/enc/LabelEncodings**

Exit Status

The **labck** command returns the following exit values:

Item	Description
0	The command completed successfully.
>0	An error occurred.

Examples

1. To check the consistency of the system-default, label-encodings file, enter the following command:

   ```
   labck
   ```

2. To check the consistency of the system-default, label-encodings file and print the system the high and low labels, enter the following command:

   ```
   labck -l
   ```

3. To check the consistency of a label-encodings file that is stored in the current directory, enter the following command:

```
labck –f ./labelencodingsfile
```

4. To copy the system-default, label-encodings file to a file with the name of **/tmp/lef**, enter the following command:

```
labck –c /tmp/lef
```

5. To replace the contents of system-default, label-encodings file with the contents of the **/tmp/lef** file, enter the following command:

```
labck –r /tmp/lef
```

Files

Item	Description
/usr/sbin/labck	Contains the **labck** command.
/etc/security/enc/LabelEncodings	System default label encodings file.

last Command

Purpose

Displays information about previous logins.

Syntax

last [-X] [**-f** *FileName*] [**-t** *Time*] [**-n** *Number* | **-***Number*] [*Name ...*] [*Terminal ...*]

Description

The **last** command displays, in reverse chronological order, all previous logins and logoffs still recorded in the **/var/adm/wtmp** file. The **/var/adm/wtmp** file collects login and logout records as these events occur and holds them until the records are processed by the **acctcon1** and **acctcon2** commands as part of the daily reporting procedures. When the time daemon, timed, changes the system time, it logs entries in wtmp under the pseudo-user "date". An entry starting with "date |" is logged before the change, and one starting with "date {" is logged after the change. This allows for accurate accounting of logins that span a time change.

The list can be restricted to:

- The number of lines specified either with the **-***Number* parameter or with the **-n** flag.
- Logins or logoffs by the users specified by the *Name* parameter.
- Logins or logoffs from the terminals specified by the *Terminal* parameter.
- A terminal can be named fully or abbreviated as a tty. For example, you can specify either the `tty0` terminal or the `0` terminal.

  Note: If you specify both a *Name* and *Terminal* parameter, the **last** command displays all logins and logoffs meeting either criterion.

For each process, the **last** command displays the:

- Time the session began
- Duration

- Terminal (tty) used

If applicable, the following information is included:

- Terminations due to rebooting
- Sessions that are still continuing

If the **last** command is interrupted, it indicates how far the search has progressed in the **/var/adm/wtmp** file. If interrupted with a **quit** signal, the command indicates how far the search has progressed and then continues the search. The **quit** signal can be any one of the following:

```
#define SIGQUIT 3 /* (*) quit,
generated from terminal special char */

#define SIGKILL 9 /* kill (cannot be caught or ignored) */

#define SIGTERM 15 /* software termination signal */
```

The **kill** command sends the default SIGTERM signal when it is invoked without any option. If you want to send the SIGQUIT signal, enter the following:

```
kill -3 (Process ID)
```

Flags

Item	Description
-f *FileName*	Specifies an alternate file from which to read logins and logoffs.
-n	Specifies the number of lines to be displayed on the list.
-t *Time*	Displays users logged in at the given Time value. The Time variable is specified in the decimal form [[CC]YY]MMDDhhmm[.SS] where: **CC** Specifies the first two digits of the year. **YY** Specifies the last two digits of the year. **MM** Specifies the month of the year (01 to 12). **DD** Specifies the day of the month (01 to 31). **hh** Specifies the hour of the day (00 to 23). **mm** Specifies the minute of the hour (00 to 59). **SS** Specifies the second of the minute (00 to 59).
-X	Prints all available characters of each user name instead of truncating to the first 8 characters.

Examples

1. To display all the recorded logins and logoffs by user `root` or from the `console` terminal, type:

   ```
   last root console
   ```

2. To display the time between reboots of the system, type:

```
last reboot
```

The `reboot` pseudo-user logs in when the system starts again.

3. To display all the users still logged in at 10.30 am on April 15th, enter:

```
last -t 04151030
```

4. To display 10 lines in the list, type:

```
last -n 10
```

5. To display all the recorded logins and logoffs without truncating the user name, type:

```
last -X
```

Files

Item	Description
/usr/bin/last	Contains the **last** command.
/var/adm/wtmp	Contains connect-time accounting data, including login, logoff, and shutdown records.

lastcomm Command

Purpose

Displays information about the last commands executed.

Syntax

lastcomm [-X][*Command*] [*Name*] [*Terminal*]

Description

The **lastcomm** command displays information, in reverse chronological order, about all previously executed commands that are still recorded in the **/var/adm/pacct** summary file. You need to run the **/usr/sbin/acct/startup** command before you can execute the **lastcomm** command.

The list the **lastcomm** command displays can be restricted to:

- Commands specified by the *Command* parameter.
- Commands executed by the user specified by the *Name* parameter.
- Commands from the terminal specified by the *Terminal* parameter.

  A terminal can be named fully or abbreviated as a tty. For example, you can specify either the `tty0` terminal or the `0` terminal.

For each process, the following information is displayed:

- The name of the user who ran the process.
- Any flags the accounting facilities collected when the command executed. The following are valid flags:

Item	Description
S	The root user executed the command.
F	The command ran after a fork, but without a following subroutine.
C	The command ran in PDP-11 compatibility mode.

Item	Description
D	The command terminated with the generation of a core file.
X	The command was terminated with a signal.

- The name of the command under which the process was called.
- The seconds of CPU time used by the process.
- The time the process was started.

Flags

Item	Description
-X	Prints all available characters of each user name instead of truncating to the first 8 characters.

Examples

1. To display information about all previously executed commands recorded in the **/var/adm/pacct** file, enter:

   ```
   lastcomm
   ```

2. To display information about commands named `a.out` executed by the `root` user on the `ttyd0` terminal, enter:

   ```
   lastcomm a.out root ttyd0
   ```

3. To display information about all previously executed commands recorded in the `/var/adm/pacct` file without truncating the user name, enter:

   ```
   lastcomm -X
   ```

Files

Item	Description
/usr/bin/lastcomm	Contains the **lastcomm** command.
/var/adm/pacct	The directory that contains the current accounting summary files.

lastlogin Command

Purpose

Reports the last login date for each user on the system.

Syntax

/usr/sbin/acct/lastlogin [**-X**]

Description

The **lastlogin** command updates the **/var/adm/acct/sum/loginlog** file to show the last date each user logged in. Normally, the **runacct** command, running under the **cron** daemon, calls this command and adds the information to the daily report. However, the **lastlogin** command can also be entered by a user who is a member of the ADM group.

Note: You should not share accounting files among nodes in a distributed environment. Each node should have its own copy of the various accounting files.

Flags

Item	Description
-X	Processes all available characters for each user name instead of truncating to the first 8 characters. This flag will also cause the **lastlogin** command to write to the **/var/adm/acct/sumx/loginlog** file instead of the **/var/adm/acct/sum/loginlog** file.

Security

Access Control: This command should grant execute (x) access only to members of the ADM group.

Files

Item	Description
/usr/sbin/acct	The path to the accounting commands.
/var/adm/wtmp	The login and logout history file.
/var/adm/acct/sum	Cumulative directory for daily accounting records.

lbxproxy Command

Purpose

Low BandWidth X proxy.

Syntax

lbxproxy [**:<***display*>] [**-help**] [**-display** *Display*] [**-motion** *Number*] [**-terminate** | **-reset**] [**-reconnect**] [**-I**] [**-nolbx**] [**-nocomp**] [**-nodelta**] [**-notags**] [**-nogfx**] [**-noimage**] [**-nosquish**] [**-nointernsc**] [**-noatomsfile**] [**-atomsfiles** *File*] [**-nowinattr**] [**-nograbcmap**] [**-norgbfile**] [**-rgbfile** *Path*] [**-tagcachesize**] [**-zlevel** *Level*] [**-compstats**] [**-nozeropad**] [**-cheaterrors**] [**-cheatevents**]

Description

The **lbxproxy** command accepts client connections, multiplexes them over a single connection to the X server, and performs various optimizations on the X protocol to make it faster over low bandwidth and/or high latency connections. Applications that would like to take advantage of the Low Bandwidth extension to X (LBX) must make their connections to an **lbxproxy**. These applications need to know nothing about LBX, they simply connect to the **lbxproxy** as if were a regular server.

For authentication/authorization, **lbxproxy** passes the credentials presented by the client along to the server. Since X clients connect to **lbxproxy**, it is important that the user's **.Xauthority** file contain entries with valid keys associated with the network ID of the proxy. **lbxproxy** does not get involved with how these entries are added to the **.Xauthority** file. The user is responsible for setting it up.

The **lbxproxy** program has various flags, all of which are optional.

If **:<***Display***>** is specified, the proxy uses the *Display* port when listening for connections. The display port is an offset from port 6000, identical to the way in which regular X display connections are specified. If no port is specified on the command line, **lbxproxy** defaults to port 63. If the port that the proxy tries to listen on is in use, the proxy exits with an error message.

At startup, **lbxproxy** pre-interns a configurable list of atoms. This allows **lbxproxy** to intern a group of atoms in a single round trip and immediately store the results in its cache. While running, **lbxproxy** uses

heuristics to decide when to delay sending window property data to the server. The heuristics depend on the size of the data, the name of the property, and whether a window manager is running through the same **lbxproxy**. Atom control is specified in the **AtomControl** file, set up during installation of **lbxproxy**, with command line overrides.

The file is a simple text file. There are three forms of lines: comments, length control, and name control. Lines starting with a **!** (exclamation point) are treated as comments. A line of the form z length specifies the minimum length in bytes before property data is delayed. A line of the form options atomname controls the given atom, where options is any combination of the following characters: **i** means the atom should be pre-interned; and **w** means data for properties with this name should be delayed only if a window manager is also running through the same **lbxproxy**.

Flags

Item	Description
-atomsfile *File*	Overrides the default **AtomControl** file.
-cheaterrors	Allows cheating on X protocol for the sake of improved performance. The X protocol guarantees that any replies, events or errors generated by a previous request are sent before those of a later request. This puts substantial restrictions on when **lbxproxy** can short circuit a request. The **-cheaterrors** flag allows **lbxproxy** to violate X protocol rules with respect to errors. Use at your own risk.
-cheatevents	The **-cheatevents** flag allows **lbxproxy** to violate X protocol rules with respect to events as well as errors. Use at your own risk.
-compstats	Reports stream compression statistics every time the proxy resets or receives a SIGHUP signal.
-display *Display*	Specifies the address of the X server supporting the LBX extension. If this flag is not specified, the display is obtained by the DISPLAY environment variable.
-help	Prints a brief help message about the command line flags.
-I	Causes all remaining arguments to be ignored.
-motion *Number*	Specifies the maximimum *Number* of events that can be in flight. A limited number of pointer motion events are allowed to be in flight between the server and the proxy at any given time. The default is 8.
-noatomsfile	Disables reading of the **AtomControl** file.
-nocomp	Disables stream compression.
-nodelta	Disables delta request substitutions.
-nogfx	Disables reencoding of graphics requests (not including image related requests).
-nograbcmap	Disables colormap grabbing.
-noimage	Disables image compression.
-nointernsc	Disables short circuiting of **InternAtom** requests.
-nolbx	Disables all LBX optimizations.
-norgbfile	Disables color name to RGB resolution in proxy.
-nosquish	Disables squishing of X events.
-notags	Disables usage of tags.
-nowinattr	Disables **GetWindowAttributes**/**GetGeometry** grouping into one round trip.
-nozeropad	Indicates to not zero out unused pad bytes in X requests, replies, and events.

Item	Description
-reconnect	Causes **lbxproxy** to reset (see -reset) and attempts to reconnect to the server when its connection to the server is broken. The default behavior of **lbxproxy** is to exit.
-rgbfile *Path*	Specifies an alternate RGB database *Path* for color name to RGB resolution.
-tagcachesize	Sets the size of the proxy's tag cache (in bytes).
-[terminate|reset]	The default behavior of **lbxproxy** is to continue running as usual when it's last client exits. The **-terminate** option will cause **lbxproxy** to exit when the last client exits. The **-reset** option will cause **lbxproxy** to reset itself when the last client exits. Resetting causes **lbxproxy** to clean up it's state and reconnect to the server.
-zlevel *Level*	Set the Zlib compression level (used for stream compression). The default is 9. 1 = worst compression, fastest. 9 = best compression, slowest.

ld Command

Purpose

Links object files.

Syntax

ld [**-D***Number*] [**-e***Label*] [**-G**] [**-H***Number*] [**-K**] [**-m**] [**-M**] [**-o***Name*] [**-r**] [**-s**] [**-S***Number*] [**-T***Number*] [**-u** *Name*] ... [**-v**] [**-V**] [**-z**] [**-Z***String*] ... [**-b***Option*] ... [**-L***Directory*] ... { **-f***FileID* ... **-l***Name* ... *InputFile* ... }

or

ld -bsvr4 [**-d**[**y** | **n**]] [**-D** *Number*] [**-e** *Label*] [**-G**] [**-H***Number*] [**-K**] [**-m**] [**-M**] [**-o***Name*] [**-r**] [**-R** *Path*] [**-s**] [**-S***Number*] [**-T***Number*] [**-u** *Name*] ... [**-v**] [**-V**] [**-z** [**defs** | **nodefs**]] [**-z multidefs**] [**-z** [**text** | **nowarntext** | **warntext**]]] [**-Z***String*] ... [**-b***Option*] ... [**-L***Directory*] ... { **-f***FileID* ... **-l***Name* ... *InputFile* ... }

Description

The **ld** command, also called the linkage editor or binder, combines object files, archives, and import files into one output object file, resolving external references. It produces an executable object file that can be run. In addition, if you specify the **ld** command without the **-s** flag, you can use the output file as an *InputFile* parameter in another call to the **ld** command. By default, the **ld** command creates and places its output in the **a.out** file.

The **ld** command can relink a program without requiring that you list all input object files again. For example, if one object file from a large program has changed, you can relink the program by listing the new object file and the old program on the command line, along with any shared libraries required by the program. See “Examples” on page 1884.

The **ld** command links input files in the order you specify on the command line. If you specify a file more than once, only the first occurrence of the file is processed. You must specify at least one input file, either with the **-bI** (uppercase letter i), **-bimport**, **-bkeepfile**, **-f**, or **-l** (lowercase letter L) flag or as an *InputFile* parameter. (The **-bI**, **-bimport**, or **-bkeepfile** flag is the **-b** flag used with the **I**, **import**, or **keepfile** option.)

Use the **cc** command to link files when you are producing programs that run under the operating system. Because the **cc** command calls the **ld** command with common options and necessary support libraries, you do not need to specify them on the command line. (This information is read from the **/etc/xlC.cfg** or **/etc/vac.cfg** configuration file.)

Linking Mode

The **ld** command can link 32-bit objects and programs as well as 64-bit objects and programs, but 32-bit and 64-bit objects may not be linked together. To specify the mode for linking, you can use the **OBJECT_MODE** environment variable or the **-b32** or **-b64** options.

Archive Files

Archive files are composite objects, which usually contain import files and object files, including shared objects. If an archive file contains another archive file or a member whose type is not recognized, the **ld** command issues a warning and ignores the unrecognized member. If an object file contained in an archive file has the **F_LOADONLY** bit set in the XCOFF header, the **ld** command ignores the member. This bit is usually used to designate old versions of shared objects that remain in the archive file to allow existing applications to load and run. New applications link with the new version of the shared object, that is, another member of the archive.

Shared Objects

A shared object, usually created by another call to the **ld** command, is an object file with the **F_SHROBJ** bit set in the XCOFF header. A shared object defines external symbols that are resolved at run time. If you specify the **-bnso** or **-bnoautoimp** option, the **ld** command processes a shared object as an ordinary object file, and if the file is stripped, the link fails.

Ordinarily, a shared object used as input is only listed in the loader section of the output file if a symbol in the shared object is actually referenced. When the run-time linker is used, however, you might want shared objects to be listed even if there are no symbols referenced. When the **-brtl** option is used, all shared objects listed on the command-line that are not archive members are listed in the output file. The system loader loads all such shared objects when the program runs, and the symbols exported by these shared objects may be used by the run-time linker. Shared objects that are archive members are not loaded automatically unless automatic loading is enabled by an import file in the archive. To enable automatic loading, see "Import and export File Format (-bI: and -bE: Flags)" on page 1881.

Import and Export Files

Import files are ASCII files that identify the external symbols to resolve at run time. An import file identifies the shared object defining the imported symbols. The system loader finds and resolves those symbols at run time. If the first line of an import file begins with #! (#, exclamation point), you can specify the file on the command line as an ordinary *InputFile*. Otherwise, you must use the **-bI** or **-bimport** option to specify the import file.

Export files are ASCII files that identify external symbols that are made available for another executable object file to import. The file format of an export file is the same as the file format of an import file.

Libraries

Libraries are files whose names end in **.a**, or possibly **.so**. To designate a library, you can specify an absolute or relative path name or use the **-l** (lowercase letter L) flag in the form **-l***Name*. The last form designates a **lib***Name***.a** file, or when the **rtl** option is used, a **lib***Name***.so** file to be searched for in several directories. These search directories include any directories that are specified by **-L** flags and the standard library directories **/usr/lib** and **/lib**.

Note: If you specify a shared object, or an archive file containing a shared object, with an absolute or relative path name, instead of with the **-l***Name* flag, the path name is included in the import file ID string in the loader section of the output file. You can override this behavior with the **-bnoipath** option.

Processing

The **ld** command processes all input files in the same manner, whether they are archives or not. It includes the symbol tables of all objects, discarding only symbol definitions that duplicate existing symbols. Unlike some other versions of the **ld** command, you do not need to order archive files so references precede definitions. Furthermore, you do not need to list an archive file more than once on the command line.

The order of the **ld** command flags does not affect how they are processed, except for the flags used with input object files, libraries, and import files. These flags are: **-L**, **-f**, **-l** (lowercase letter L), **-bkeepfile**, and **-bI** (uppercase letter i). The flags are processed in the following order:

1. The **-L** flag adds a directory to the list of search directories to locate libraries specified by the **-l** (lowercase letter L) flag. The directories are searched in the order specified. All **-L** flags are processed before any **-l** flags are processed.
2. The **ld** command processes the *InputFile* parameters, the files specified by the **-f** flag and libraries specified by the **-l** (lowercase letter L) flag in the order specified.
3. The **ld** command processes import files specified by the **-bI** (uppercase letter i) flag in the order specified after processing all other object files and libraries. You can specify an import file as an input file without the **-bI** flag if it is necessary to process the file before processing some object files. In this case, the first line of the import file must begin with the #! (#, exclamation point) symbols, and the import file is processed with other input files as described in step 2.
4. The **-bkeepfile** option names an input file on which the **ld** command does not perform garbage collection. If the specified input file is also specified as an *InputFile* parameter or listed in a file specified by the **-f** flag, the **-bkeepfile** option does not affect the order in which the file is processed. Otherwise, the file is processed in order along with other input files, as described in step 2.

An output file produced by the **ld** command has execute permission set, unless you specify the **-r** flag or **-bnox** option or errors were reported while linking. An existing output file is not overwritten if any severe errors occurred, or if the output file was specified as an input file and any errors occurred.

Symbols

The **ld** command uses the following predefined symbols to provide special address locations and can be declared in C syntax as **extern char** *name***[ ]**.The symbol names are:

Item	Description
_text	Specifies the first location of the program.
_etext	Specifies the first location after the program.
_data	Specifies the first location of the data.
_edata	Specifies the first location after the initialized data
_end or **end**	Specifies the first location after all data.

The only way to use these symbols is to take their addresses. If an input file redefines any of these symbols, there may be unpredictable results. An additional predefined symbol, **_ptrgl**, is used by compilers to implement calls using function pointers.

Garbage Collection

By default, the **ld** command performs garbage collection, deleting control sections (CSECTs) that are not referenced when generating the output file.

A CSECT is an indivisible unit of coding or data. A CSECT references another CSECT if it contains a relocation entry (RLD) referring to a symbol contained in the other CSECT. A referenced CSECT causes all CSECTs it references to be referenced as well. In addition, a CSECT is referenced if it contains exported symbols, symbols specified with the **-u** flag, or the symbol designated as the entry point with the **-e** flag.

If a symbol is not referenced but is needed in the output file, you can export the symbol, specify the symbol with the **-u** flag, or suppress garbage collection. To suppress garbage collection, use the **-r** flag or **-bnogc** option. To suppress garbage collection for individual object files, use the **-bkeepfile** option or the **-bgcbypass** option. Even when garbage collection is suppressed, unreferenced internal symbols are deleted.

Ignored and Unsupported Flags

For compatibility with other versions of the **ld** command, some flags are recognized but ignored. These flags produce a message stating that the flag and its operand were ignored. An ignored flag does not cause the **ld** command to stop without further processing. The following flags are ignored:

```
-ANumber          -bnostrcmpct      -n
-bfilelist        -bstrcmpct        -N
-bfl              -BNumber          -Q
```

```
-bforceimp      -d              -RNumber
-bi             -i              -VNumber
-binsert        -j[Key:]Number  -x
-bnoforceimp    -kKey:Path      -YNumber
```

Note: When the `-bsvr4` option is present on the `ld` command line, the `-R` and `-z` options are redefined.

Flags that the **ld** command does not support result in an error message. After all unsupported flags are diagnosed, the **ld** command stops without further processing.

Flags

The `ld` command conforms to the XPG Utility Syntax Guidelines, except that the argument — only applies to the next operand, not to the remaining operands on the command line. For example, in the command line:

```
ld -- -s -v
```

The `-s` is treated as a filename and the `-v` is treated as a flag. To have `-v` treated as a filename, specify:

```
ld -- -s -- -v
```

Note: Enter a flag with an operand with or without a space between the flag and the operand. You can specify numeric values in decimal, octal (with a leading 0), or hexadecimal (with a leading 0x or 0X) format. If you specify conflicting flags on the command line, the **ld** command accepts the latest flag and ignores earlier ones.

Item	Description
-b*Option*	Sets special processing options. This flag can be repeated. For more information on these options, see “Options (-bOptions)” on page 1866.
-d [**y** | **n**]	When **-dy** is specified, **ld** uses dynamic linking; this option is equivalent to the **-b so** option. When **-dn** is specified, **ld** uses static linking; this option is equivalent to the **-b nso** option. The default is **-dy**. This option is valid only when the **-bsvr4** option is specified.
-D*Number*	Sets the starting address for the initialized data (the data section) of the output file to *Number.* If the specified number is -1, the data section starts immediately after the text section. By default, the data section begins at location 0. If both the **-D** and **-bpD** flags are specified, the latter flag takes precedence. **Note:** The system loader relocates the data section at run time, so the specified number only affects addresses listed in address maps or printed by utilities such as the **dump** or **nm** command.
-e*Label*	Sets the entry point of the executable output file to *Label*. The default entry point is **__start** (double underscore **start**).
-f*FileID*	Specifies a file containing a list of input files to process. FileID must contain a list of input file names. Each line in FileID is treated as if it were listed separately on the **ld** command line. Lines in the file can contain shell pattern characters * (asterisk), [(left bracket),] (right bracket), and ? (question mark), which are expanded using the **glob** subroutine and can designate multiple object files.
-G	Produces a shared object enabled for use with the run-time linker. The **-G** flag is equivalent to specifying the **erok**, **rtl**, **nortllib**, **nosymbolic**, **noautoexp**, and **M:SRE** options with the **-b** flag. Subsequent options can override these options.

Item	Description
-H*Number*	Aligns the text, data, and loader sections of the output file so that each section begins on a file offset that is a multiple of *Number*. If the specified number is 1, no alignment occurs. If the specified number is 0, the loader section is aligned on a word boundary, and the text and data sections are aligned on a boundary so as to satisfy the alignment of all CSECTs in the sections. The default value is 0. If the specified *Number* causes any CSECTS to be unaligned within the output file, the **ld** command issues a warning and the output executable file may not load or run.
-K	Aligns the header, text, data, and loader sections of the output file so that each section begins on a page boundary. This flag is equivalent to specifying **-H***Number*, where *Number* is the page size of the machine on which **ld** is running.
-l*Name*	Processes the ***libName*.a** file. In dynamic mode, with the **rtl** option, processes the ***libName*.a** or ***libName*.so** file. In all cases, directories specified by the **-L** flag or in the standard library directories (**/usr/lib** and **/lib**) are searched to find the file. In dynamic mode with the **rtl** option, the first directory containing either ***libName*.so** or ***libName*.a** satisfies the search. If both files are found in the same directory, ***libName*.so** is used. You can repeat this flag. For more information about dynamic mode, see "Run-time Linking" on page 1880. **Note:** The first definition of a symbol is kept, even if no reference to the symbol has been seen when the archive is read. In other versions of the **ld** command, a symbol defined in an archive is ignored if no reference to the symbol has been seen when the archive is read.
-L*Directory*	Adds *Directory* to the list of search directories used for finding libraries designated by the **-l** (lowercase letter L) flag. The list of directories, including the standard library directories, is also recorded in the output object file loader section for use by the system loader unless you use the **-blibpath**, **-bnolibpath**, or `-bsvr4` option. You can repeat this flag.
-m or **-M**	Lists to standard output the names of all files and archive members processed to create the output file. Shared objects and import files are not listed.
-o*Name*	Names the output file *Name*. By default, the name of the output file is **a.out**.
-r	Produces a nonexecutable output file to use as an input file in another **ld** command call. This file may also contain unresolved symbols. The **-r** flag is equivalent to specifying the **erok**, **noglink, nox**, and **nogc** options with the **-b** flag. (Subsequent options can override these options.)
-R *Path*	Valid only when the **-bsvr4** option is present on the **ld** command line. It defines a colon-separated list of directories used to specify library search directories to the runtime linker. *Path*, if present and not NULL, is recorded in the output file's loader section. Then it is used when linking an executable with shared libraries at runtime. Multiple instances of this option are concatenated together with each *Path* separated by a colon.
-s	Strips the symbol table, line number information, and relocation information when creating the output file. Stripping saves space but impairs the usefulness of the debuggers. You can also strip an existing executable by using the **strip** command. **Note:** Non-shared objects cannot be linked if they are stripped. A shared object can be stripped, but a stripped, shared object cannot be used when linking statically.

Item	Description
-S*Number*	Sets the maximum size (in bytes) allowed for the user stack when the output executable program is run. This value is saved in the auxiliary header and used by the system loader to set the soft ulimit. The default value is **0**. For more information on large user stacks and 32-bit programs, see "Large Program Support Overview3" in *General Programming Concepts: Writing and Debugging Programs*.
-T*Number*	Sets the starting address of the text section of the output file to *Number*. The default value is **0**. If both the **-T** and **-bpT** flags are specified, the latter flag takes precedence. **Note:** The system loader relocates the text section at run time, so the specified number affects only addresses listed in address maps or printed by utilities such as the **nm** or the **dump** command. **Attention:** If *Number* is `0x1000xxxxxxx` and the program linked is a 64-bit program, the system loader loads the executable text into memory starting from the segment at the address `0x10000000000`.
-u*Name*	Prevents garbage collection of the external symbol *Name*. If the specified symbol does not exist, a warning is reported. You can repeat this flag.
-v	Writes additional information about binder command execution to the loadmap file.
-V	Writes the version string of **ld** to standard error (stderr).
-z	In the absence of the **-b svr4** option, functions the same as the **-K** flag.
-z defs	Forces a fatal error if any undefined symbols remain at the end of the link. This is the default when an executable is built. It is also useful when building a shared library to assure that the object is self-contained, that is, that all its symbolic references are resolved internally. This option is valid only when the **-b svr4** option is specified. It is equivalent to **-b ernotok** option.
-z nodefs	Allows undefined symbols. This is the default when a shared library is built. When used with executables, the behavior of references to such undefined symbols is unspecified. This option is valid only when the **-b svr4** option is specified. It is equivalent to **-b erok** option.
-z multidefs	Allows multiple symbol definitions. By default, multiple symbol definitions occurring between relocatable objects (**.o** files) will result in a fatal error condition. This option suppresses the error condition and allows the first symbol definition to be taken. This option is valid only when the **-b svr4** option is specified.
-z text	In dynamic mode only, forces a fatal error if any relocations against the **.text** section remain. This option is valid only when the **-b svr4** option is specified.
-z nowarntext	In dynamic mode only, allows relocations against all mappable sections, including the **.text** section. This is the default when building a shared library. This option is valid only when the **-b svr4** option is specified.
-z warntext	In dynamic mode only, warns if any relocations against the **.text** section remain. This is the default when building an executable. This option is valid only when the **-b svr4** option is specified.

Item	Description
-Z*String*	Prefixes the names of the standard library directories with *String* when searching for libraries specified by the **-l** (lowercase letter L) flag. For example, with the **-Z/***test* and **-l***xyz* flags, the **ld** command looks for the **/test/usr/lib/libxyz.a** and **/test/lib/libxyz.a** files. When the **-Z***String* flag is used, the standard library directories are not searched. This flag has no effect on the library path information saved in the loader section of the output file. This flag is useful when developing a new version of a library. You can repeat this flag.

The Binder

The **ld** command verifies the command-line arguments and calls the binder (by default the **/usr/ccs/bin/bind** file), passing a generated list of binder subcommands. The binder program actually links the files. Although the binder is usually called by the **ld** command, you can start the binder directly. In this case, the binder reads commands from standard input.

Two options affect the calling of the binder. The **binder** option specifies which binder to call, and the **nobind** option prevents the **ld** command from calling a binder. Other binder options affect the binder subcommands that are generated.

If the **ld** command does not detect any errors in the options or command-line arguments, it calls the binder. The binder is called with a command line of the form:

```
bind [quiet_opt] [loadmap_opt]
```

The default value for *quiet_opt* is `quiet` and the default value for the *loadmap_opt* is the null string, so the default command line is:

```
/usr/ccs/bin/bind quiet
```

Options (-bOptions)

The following values are possible for the *Options* variable of the **-b** flag. You can list more than one option after the **-b** flag, separating them with a single blank.

Note:

1. In the following list of binder options, two option names separated by the word *or* are synonymous.
2. The *FileID* indicates a path name. You can use either a relative or a full path name.
3. For a non-repeatable option that is followed by an argument, you can negate the option using a null argument. That is, specify only the option and the colon.
4. If you specify conflicting options, the last one takes precedence.

Item	Description
32	Specifies 32-bit linking mode. In this mode, all input object files must be XCOFF32 files, or an error is reported. XCOFF64 archive members are ignored. For import or export files specifying the mode of certain symbols, 64-bit symbols are ignored. If both **-b32** and **-b64** options are specified, the last specified option is used. If neither option is specified, the mode is determined from the value of environment variable **OBJECT_MODE**.
64	Specifies 64-bit linking mode. In this mode, all input object files must be XCOFF64 files, or an error will be reported. XCOFF32 archive members are ignored. For import or export files specifying the mode of certain symbols, 32-bit symbols are ignored. If both **-b32** and **-b64** options are specified, the last specified option is used. If neither option is specified, the mode is determined from the value of environment variable **OBJECT_MODE**.
asis	Processes all external symbols in mixed case. This is the default. To process all external symbols in uppercase, see the **caps** option that follows.

Item	Description
aslr or **aslr**:*[tdsmp]** or **aslr**:-	Specifies the address space layout randomization for the program. The **aslr** option enables all of the randomization attributes when only this option is used, and, `-baslr:-`, disables all of the randomization attributes. If the **aslr** option is followed by a colon, individual attributes can be enabled. The following attributes denote text, data, stack, mmap, and private-libraries: *t*, *d*, *s*, *m*, and *p*. For the attributes that are not listed, the randomization settings remain disabled. **Note:** The *m* and *p* attributes cannot be specified for 32-bit programs. Some programs are compiled and linked such that relocatable addresses are mapped into the text section. This requires the relocation of the addresses when the program is run. These programs fail if randomization is enabled for text or data. As a result, when text-section relocations exist, text and data randomization are not enabled unless the *t* and *d* attributes are explicitly specified when using the **aslr** option.
autoexp	Automatically exports some symbols from the output module without having to list them in an export file. (This option does not export all symbols from the output module. Use the **-bexpall** option to export all symbols.)This is the default. Use this option when linking a main program. The linker assumes that you are linking a main program when you do not specify a module type (with the **M** or **modtype** option) beginning with S and you do not use the **noentry** option. When you use the **autoexp** option, if any shared object listed on the command-line imports a symbol from the special file **.** (dot), and the module being linked contains a local definition of the symbol, the symbol is exported automatically. Other symbols are also exported automatically when you link with the **rtl** option. If a symbol defined in the module being linked has one or more additional definitions exported from a shared object listed on the command-line, and if any of the definitions is a BSS symbol, the symbol is exported automatically. If the definition in the module being linked is a BSS symbol, the symbol is exported with the `nosymbolic` attribute. Otherwise, the symbol is exported with the `symbolic` attribute. If the symbol is listed in an export file with another export attribute, the explicit attribute is used. If the **autoexp** option would automatically export a symbol, but the symbol is listed in an export file with the **list** attribute, the symbol is not exported.
autoimp or **so**	Imports symbols from any shared objects specified as input files. The shared objects are referenced but not included as part of the output object file. This is the default.
autoload: *path/file(member)*	Automatically load archive member.
bigtls	Generates extra code if the size of thread-local storage in the output object or program is larger than 64 KB and a compiler was used that generates direct references to local-exec or local-dynamic thread-local variables. Extra code is needed for every direct reference to a thread-local variable that cannot be addressed with a 16-bit offset. Because a program containing extra code might have poor performance, it is better to reduce the number of thread-local variables using direct references than to use the option. The default option is the **nobigtls** option.
bigtoc	Generates extra code if the size of the table of contents (TOC) is greater than 64KB. Extra code is needed for every reference to a TOC symbol that cannot be addressed with a 16-bit offset. Because a program containing generated code may have poor performance, reduce the number of TOC entries needed by the program before using this option. The default is the **nobigtoc** option.
bindcmds:*FileID*	Writes a copy of the binder commands generated by the **ld** command to *FileID*. You can redirect the resultant file as standard input to the binder program when the binder program is called as a standalone program. By default, no file is produced.
binder:*FileID*	Uses *FileID* as the binder called by the **ld** command. The default binder is the **/usr/ccs/bin/bind** file.
bindopts:*FileID*	Writes a copy of the binder program arguments to *FileID*. You can use the resultant file to start the binder program as a standalone program. By default, no file is produced.
C:*FileID* or **calls:***FileID*	Writes an address map of the output object file to *FileID*. Symbols are sorted by section and then by address. For each symbol listed in the map, references from the symbol to other symbols are listed. By default, no file is produced. To learn more about the **calls** option, see "Address Maps" on page 1883.
caps	Processes all external symbols in uppercase. The default is the **asis** option.

Item	Description
cdtors[**:**[*incl*][**:**[*nnn*] [**:***order*]]]	The linker gathers information about C++ static constructor or destructor functions and saves this information in the output file. The *incl* suboption tells the linker which archive members to search when creating the saved information. The possible values are: **all** Searches all members of all archives for constructor or destructor functions. This is the default. **mbr** Searches for constructor or destructor functions only if the member is included in the output file. Using the **mbr** value is equivalent to using the **-qtwolink** and **-bsvr4** options. **csect** Searches for the constructor or destructor functions only in **csects** included from the archive. Using the **csect** value is equivalent to using the **-qtwolink** option without the **-bsvr4** option. The *nnn* suboption specifies the priority of the output module. This priority is used to control the order in which modules are initialized, in case a program loads multiple modules at the same time. (The priority is ignored if the output file is a program and not a shared object.) The priority can be in the range from -2^{31} to $2^{31}-1$. The default priority is 0. Values in the range from -2^{31} to $-2^{31}+1023$ are reserved for C++ runtime initialization. The *order* suboption specifies the order in which individual constructor or destructor functions are called, for all functions with the same priority. The possible values are: **s** Sorts in an arbitrary order based on function names. This is the default, and is compatible with the order that the XL C++ compiler uses. **c** Sorts in link order. Constructor or destructor functions in the first input file are initialized first. In archives, functions in the first member are initialized first. **r** Sorts in reverse link order. Constructor or destructor functions in the last input file are initialized first. In archives, functions in the last member are initialized first. You can specify this option multiple times, but the last suboption that you specify is used. An unspecified suboption does not affect the current or default value. For example, **-bcdtors:csect:20:s -bcdtors:::r** is the same as **-bcdtors:csect::20:r**. The default is **-bnocdtors**. If **-bcdtors** is specified, this is equivalent to **-bcdtors:all:0:s**. **Note:** 1. If necessary, the XL C++ compiler produces this option automatically. 2. Functions specified with the **-binitfini** option are invoked independently of static constructor or destructor functions. For more information about the order of initialization, see the **dlopen** subroutine.
comprld or **crld**	Combines multiple relocation entries (RLDs) at the same address into a single RLD when possible. This is the default.
cror15	Uses the **cror 15,15,15** (`0x4def7b82`) instruction as the special no-op instruction following a call instruction. The default value is **ori 0, 0, 0** (`0x60000000`). See the **nop** option. Use this option when linking object files on the current level of the system that you intend to relink on AIX 3.1.
cror31	Uses the **cror 31,31,31** (`0x4ffffb82`) instruction as the special no-op instruction following a call instruction. The default value is **ori 0, 0, 0** (`0x60000000`). See the **nop** option. Use this option when linking object files on the current level of the system that you intend to relink on AIX 3.2.
D: *Number* [*/dsa*] or **maxdata:***Number*[*/dsa*]	Sets the maximum size (in bytes) that is allowed for the user data area (or user heap) when the executable program is run. This value is saved in the auxiliary header and used by the system loader to increase the soft data ulimit, if required. The default value is 0. When this option is used, the specified number of bytes are reserved for the user data area. The program might not explicitly map objects, by using **shmat** or **mmap** functions to virtual addresses that are reserved for the user data area. For 32-bit programs, the maximum value allowed by the system is `0x80000000` for programs that are running under Large Program Support and `0xD0000000` for programs that are running under Very Large Program Support. See "Large Program Support Overview" in *General Programming Concepts: Writing and Debugging Programs*. When a non-zero value is specified, the user data area begins in segment 3, and the program uses as many segments as neccessary to satisfy the specified **maxdata** value. For 64-bit programs the **maxdata** option provides a guaranteed maximum size for the programs data heap. Any value can be specified but the data area cannot extend past `0x06FFFFFFFFFFFFF8` regardless of the maxdata value specified.

Item	Description
datapsize:*psize*	Requests *psize* page sizes in bytes for data. The value can be specified as a decimal, hexadecimal, or octal number. The number specifications are the same as in C programming language. Additionally, the page sizes can be specified as a number followed by a one-character suffix: • k or K for kilo or 0x400 bytes • m or M for mega or 0x100000 bytes • g or G for giga or 0x40000000 bytes • t or T for tera or 0x10000000000 bytes • p or P for peta or 0x4000000000000 bytes • x or X for exo or 0x1000000000000000 bytes For example, either `-b datapsize:16k` or `-b datapsize:0x4000` will request 0x4000 for data and set the F_VARPG bit in the XCOFF header.
dbg:*Option* or **debugopt**:*Option*	Sets a special debugging or control option. By default, no debug option is set. The **dbg:loadabs** or **debugopt:loadabs** option is used to indicate that the output program is loaded at the same address as the address specified by the **-T** and **-D** flags. In this case, a branch-absolute instruction is never changed to a (relative) branch instruction even if its target is a relocatable symbol. Similarly, a branch instruction is never changed to a branch-absolute instruction.
delcsect	Deletes all symbols in a CSECT if any symbol in the CSECT was defined by a previously read object file. This option prevents more than one instance of the same function from existing in the same program. For example, if **a.o** defines function **a()** and **b.o** defines functions **a()** and **b()**, linking **a.o** and **b.o** with the **-bdelcsect** option deletes symbols **a()** and **b()** from **b.o**. Thus, two instances of **a()** do not exist. The default is the **nodelcsect** option.
dynamic or **shared**	Cause the linker to process subsequent shared objects in dynamic mode. This is the default. In dynamic mode, shared objects are not statically included in the output file. Instead, the shared objects are listed in the loader section of the output file. When you specify the **rtl** option and dynamic mode is in effect, files ending in **.so** as well as **.a** satisfy searches for libraries specified with the **-l** (lowercase L) flag. When both are in effect, preference is given to **.so** instead of **.a** when present in same directory. When you specify the **rtl** option and static mode is in effect, files ending in **.a** are processed.
E:*FileID* or **export:***FileID*	Exports the external symbols listed in the file *FileID.* Exported symbols are listed in the loader section of the output file. There is no default export file. When the `svr4` option is used, the **E:***FileID* option cancels any `expall` or `expfull` options.
ernotok or **f**	Reports an error if there are any unresolved external references. This is the default.
erok	Produces the output object file without errors even if there are unresolved external references. The default is the **ernotok** option.
errmsg	Writes error messages to standard error if the error level of the message is greater than or equal to the value of the **halt** option and the **quiet** option is used or standard output is redirected. This is the default.
ex1:*FileID*, **ex2:***FileID*, **ex3:***FileID*, **ex4:***FileID*, and **ex5:***FileID*	Provide user exits in the typical binder subcommand sequence. Each file specified by *FileID* must contain a list of binder subcommands, which will be run as follows: **ex1:*FileID*** Before reading any *InputFiles* **ex2:*FileID*** Immediately before symbol resolution **ex3:*FileID*** Immediately after symbol resolution **ex4:*FileID*** Immediately before writing the output file **ex5:*FileID*** Immediately after writing the output file
expall	Exports all global symbols, except imported symbols, unreferenced symbols defined in archive members, and symbols beginning with an underscore (_). You can export additional symbols by listing them in an export file or using the `expfull` option. This option does not affect symbols exported by the **autoexp** option. When you use this option, you might be able to avoid using an export file. On the other hand, using an export file provides explicit control over which symbols are exported, and allows you to use other global symbols within your shared object without worrying about conflicting with names exported from other shared objects. The default is **noexpall**.
expfull	Exports all global symbols other than imported symbols. Exported global symbols include unreferenced symbols defined in archive members, symbols beginning with an underscore (_), and the module's entry point. This option does not affect symbols exported by the `autoexp` option. The default is `noexpfull` unless the `svr4` option is used.
export:*FileID*	Functions the same as the **E:***FileID* option.
f	Functions the same as the **ernotok** option.
forceimprw	Forces read-only CSECTs that contain references to imported symbols to become read-write. The default is `noforceimprw`.

Item	Description
forkpolicy:*policy*	Sets the _AOUT_FORK_POLICY and _AOUT_FORK_COR flags in the XCOFF auxiliary header, when linking a 64-bit program. If *policy* is `cor`, the _AOUT_FORK_COR flag is also set, requesting the use of the copy-on-reference forktree policy when the program is run. If *policy* is `cow`, the _AOUT_FORK_COR flag is reset, requesting the use of the copy-on-write forktree policy when the program is run. When linking a 32-bit program, this flag is ignored. The default is `noforkpolicy`.
gc	Performs garbage collection. Use the **nogc**, **gcbypass**, or **keepfile** option to prevent garbage collection for some or all object files. This is the default.
gcbypass:*Number*	Specifies the number of files to bypass when garbage collecting if the **gc** option is specified. This option is ignored if the **nogc** option is used. If *Number* is 0, this option is equivalent to the **gc** option and garbage collection is performed for all files. The default value is **0**.
glink:*FileID*	Uses the global linkage prototype code specified by *FileID.* Global-linkage interface code is generated for each imported or undefined function. In 32-bit mode, the default is the **/usr/lib/glink.o** file. In 64-bit mode, the default is the **/usr/lib/glink64.o** file.
h:*Number* or **halt:***Number*	Specifies the maximum error level for binder command processing to continue. The default value is **4**. If any binder subcommand has a return value greater than *Number*, no additional binder subcommands are processed. If the halt level value is 8 or greater, the output file may not be executable if it is produced at all. Return values are: **0** No error **4** Warning **8** Error **12** Severe error **16** Internal program error
I:*FileID* or **import:***FileID*	(Uppercase i) Imports the symbols listed in *FileID.* There is no default import file.
initfini:[*Initial*] [**:***Termination*] [**:***Priority*]	Specifies a module initialization and termination function for a module, where *Initial* is an initialization routine, *Termination* is a termination routine, and *Priority* is a signed integer, with values from -2,147,483,648 to 2,147,483,647. You must specify at least one of *Initial* and *Termination*, and if you omit both *Termination* and *Priority*, you must omit the colon after *Initial* as well. If you do not specify *Priority*, 0 is the default. This option can be repeated. This option sorts routines by priority, starting with the routine with the smallest (most negative) priority. It invokes initialization routines in order, and termination routines in reverse order. This option invokes routines with the same priority in an unspecified order, but if multiple **initfini** options specify the same priority and both an initialization and termination routine, it preserves the relative order of the routines. For example, if you specify the options **initfini:i1:f1** and **initfini:i2:f2**, then function **i1** and **i2** are invoked in an unspecified order, but if **i1** is invoked before **i2** when the module is loaded, **f2** will be invoked before **f1** when the module is unloaded. **Note:** 1. The priorities in the following inclusive ranges are reserved: `-2,147,483,640 to -2,147,000,000` `-1,999,999,999 to -1,000,000,000` `-99,999,999 to -50,000,000` `0` `50,000,000 to 99,999,999` `1,000,000,000 to 1,999,999,999` `2,147,000,000 to 2,147,483,640` 2. Functions specified with the **-binitfini** option are invoked independently of static constructor or destructor functions. For more information about the order of initialization, see the **dlopen** subroutine.
ipath	For shared objects listed on the command-line, rather than specified with the **-l** flag, use the path component when listing the shared object in the loader section of the output file. This is the default.
keepfile:*FileID*	Prevents garbage collection of *FileID*. By default, the binder deletes unreferenced CSECTS in all files. You can repeat this option.

Item	Description
lazy	Enables lazy loading of a module's dependent modules. This option adds a **-lrtl** option following other flags and options. If the **-brtl** option is specified, the **-blazy** option is ignored and lazy loading is not enabled. When a module is linked, a list of its dependent modules is saved in the module's loader section. The system loader automatically loads the dependent modules after the module is loaded. When lazy loading is enabled, loading is deferred for some dependents until a function is called in the module for the first time. A module is lazy loaded when all references to the module are function calls. If variables in the module are referenced, the module is loaded in the typical way. **Note:** Be careful while comparing function pointers if you are using lazy loading. Usually a function has a unique address to compare two function pointers to determine whether they refer to the same function. When using lazy loading to link a module, the address of a function in a lazy loaded module is not the same address computed by other modules. Programs that depend upon the comparison of function pointers should not use lazy loading. For more information about lazy loading, refer to "Shared Libraries and Lazy Loading" in *General Programming Concepts: Writing and Debugging Programs*.
l:*FileID* or **loadmap:***FileID*	(Lowercase L)Writes each binder subcommand and its results to *FileID.* By default, no file is produced.
libpath:*Path*	Uses *Path* as the library path when writing the loader section of the output file. *Path* is neither checked for validity nor used when searching for libraries specified by the **-l** flag. *Path* overrides any library paths generated when the **-L** flag is used. If you do not specify any **-L** flags, or if you specify the **nolibpath** option, the default library path information is written in the loader section of the output file. The default library path information is the value of the **LIBPATH** environment variable if it is defined, and **/usr/lib:/lib**, otherwise.
loadmap:*FileID*	Functions the same as the **l:***FileID* option.
lpdata	Sets the F_LPDATA bit in the XCOFF header of the executable file. When this bit is set, the process is going to request large pages for its data.

Item	Description
M:*ModuleType* or **modtype:***ModuleType*	Sets the two-character module-type field and the shared object flag in the object file. The module type is not checked by the binder, but it should be set to one of the following values: **1L** Single use. Module requires a private copy of the data section for each load. **RE** Reusable. Module requires a private copy of the data area for each process dependent on the module. **RO** Read only. Module is read-only, and can be used by several processes at one time. If an **S** prefix is used on any of the preceding options, the shared flag in the object file is set. The system loader attempts to share a single instance of the data section of an RO module. Otherwise, the module type is ignored by the system loader. The default value is **1L**. **UR** Sets the **SGETUREGS** flag for the linker. When the **SGETUREGS** flag is set, the contents of the registers are stored in a buffer. This option is used by coredump system call.
map:*FileID* or **R:***FileID*	Writes an address map of the output object file to *FileID.* Symbols are sorted by section and then by address. By default, no file is produced. To learn more about the **map** option, see “Address Maps” on page 1883.
maxdata:*Number*[/*dsa*]	Functions the same as the **D:***Number*[/*dsa*] option.
maxstack:*Number* or **S:***Number*	Functions the same as the **-S** flag.
modtype:*ModuleType*	Functions the same as the **M:***ModuleType* option.

Item	Description
nl or **noloadmap**	Does not write the binder subcommands and their results to a load map file. This is the default.
noaslr	Cancels the effect of a previous **aslr** option on the command line.
noautoexp	Prevents automatic exportation of any symbols. The default is the **autoexp** option.
noautoimp or **nso**	Links any unstripped, shared objects as ordinary object files. When you use this option, the loader section of shared objects is not used. The default is the **autoimp** or **so** option. **Note:** By using either of these flags, you statically link a shared object file into an application. Any application that is statically linked is *not* binary portable from any fix or release level to any other fix or release level.
nobigtls	Generates a severe error message if the compiler generates direct references to thread-local variables and the relocation to one of the variables overflows because the size of the thread-local storage is greater than 64 KB. If an output file is produced, it will not run correctly. The **nobigtls** option is the default option.
nobigtoc	Generates a severe error message if the size of the TOC is greater than 64 KB. If an output file is produced, it will not execute correctly. This is the default.
nobind	Omits calling the binder. Instead, the **ld** command writes the generated list of binder subcommands to standard output. By default, the **ld** command calls the binder.
nocdtors	Does not gather static constructor or destructor functions. This is the default.
nocomprld or **nocrld**	Does not combine multiple relocation entries (RLDs) at the same address into a single RLD. The default is the **comprld** or **crld** option.
nodelcsect	Allows all symbols in the CSECT to be considered during symbol resolution, even if some symbol in the CSECT is defined in a previously read object file. For more information, see the **delcsect** option. The **nodelcsect** option is the default.
noexpall	Does not export symbols unless you list them in an export file or you export them with the **autoexp** option. This is the default.
noexpfull	Does not export symbols unless you list them in an export file or you export them with the `autoexp` option. This is the default, unless the `svr4` option is used.
noentry	Indicates that the output file has no entry point. To retain any needed symbols, specify them with the **-u** flag or with an export file. You can also use the **-r** flag or the **nogc** or **gcbtpass** options to keep all external symbols in some or all object files. If neither the **noentry** nor the **nox** option is used and the entry point is not found, a warning is issued. This warning is suppressed when the `svr4` option is used.
noerrmsg	Does not write error messages to standard error. Use this option if you specify the **noquiet** option and you pipe standard output to a command such as **tee** or **pg**.
noforceimprw	Allows read-only CSECTs to reference imported symbols. This is the default.

Item	Description
noforkpolicy	Clears the _AOUT_FORK_POLICY and _AOUT_FORK_COR flags in the XCOFF auxiliary header, when linking a 64-bit program. The default forktree policy is used, unless a forktree policy is specified with the VMM_CNTRL environment variable. When linking a 32-bit program, this flag is ignored. This is the default.
nogc	Prevents garbage collection. CSECTs in all object files that contain global symbols are kept, whether they are referenced or not. The default is the **gc** option.
noglink	Prevents the **ld** command from inserting global linkage code. By default, the binder inserts the global linkage code.
noipath	For shared objects listed on the command-line, rather than specified with the **-l** flag, use a null path component when listing the shared object in the loader section of the output file. A null path component is always used for shared objects specified with the **-l** flag. This option does not affect the specification of a path component by using a line beginning with #! in an import file. The default is the **ipath** option.
nolibpath	Overrides any previous library path generated by the **-L** flag or specified by the **libpath** option. Instead, the default library path information is written in the loader section of the output file. The default library path information is the value of the **LIBPATH** environment variable if it is defined, and **/usr/lib:/lib** otherwise.
noloadmap	Functions the same as the **nl** option.
nolpdata	Clears the F_LPDATA bit in the XCOFF header of the executable file. When this bit is not set, the process is going to use small (regular) pages for its data.
nom	Does not list the object files used to create the output file. This option overrides the **-m** flag. This is the default.
noobjreorder	Does not use the depth-first CSECT reordering logic. The CSECTs in the output file are arranged in the same order that the object files and library files were specified on the command line, except as follows: • CSECTs are placed in their correct text, data, or BSS section of the object file, based on the storage-mapping class field of each CSECT. • All CSECTs with a storage-mapping class of XMC_TC (TOC address constant) or XMC_TD (TOC variable) are grouped together. If both the **noobjreorder** and **noreorder** options are specified, the **noreorder** option takes precedence. The default is the **reorder** option.
noorder_file	Does not map symbols in a specified order. This flag negates the effect of a previous **-border_file** flag. This is the default.

Item	Description
nop:*Nop*	Specifies the no-op instruction used after branches to local routines. *Nop* can be one of the special values **cror15**, **cror31**, **ori**, or an eight-digit hexadecimal number. The **ori** instruction is the default. Specifying the **-bnop:cror15** option is equivalent to specifying the **-bcror15** option; specifying the **-bnop:cror31** option is equivalent to specifying the **-bcror31** option. If you specify one of the special **nop** options, all previous **nop** options are overridden If *Nop* is an eight-digit hexadecimal number, it specifies an arbitrary machine instruction. This machine instruction overrides any previously specified special value for *Nop* instruction. When you use this form, you can repeat this option. The last machine instruction specified is the instruction generated by the binder after intramodule branches. Other specified machine instructions are recognized as no-op instructions, but are converted to the preferred no-op instruction.
>|>|**noplugin_opt**	Discards any previous plug-in options that are specified with the **plugin_opt** option.|<|<
noquiet	Writes each binder subcommand and its results to standard output. The default is the **quiet** option.
noreorder	Does not reorder CSECTs, except to combine all XMC_TC (TOC address constant) and XMC_TD (TOC variable) CSECTs and place them in the data section, and combine all BSS symbols and place them in the bss section. All other CSECTs are placed in the text section, so text and data are mixed in the output file. When the **noreorder** option is used, the text section of the output file may no longer be position-independent and the system loader will not load a module if the text section is not position-independent. Therefore, avoid using this option for programs and kernel extensions. If both **noobjreorder** and **noreorder** options are specified, the **noreorder** option takes precedence. The default is the **reorder** option.
nortl	Disables run-time linking for the output file. This option implies the **nortllib** and **nosymbolic-** options. Furthermore, additional actions described under the **rtl** option are not taken. This is the default unless the `svr4` option is used.
nortllib	Does not include a reference to the run-time linker. If a main program is linked with this option, no run-time linking will take place in the program, regardless of the way any shared modules were linked that are used by the program. This is the default unless the `svr4` option is used.
norwexec	Specifies that if the system's `sed_config` setting is not off, the process' private data areas will have non-execute permission.
noshrsymtab	Prevents the _AOUT_SHR_SYMTAB flag from being set in the output object. This is the default state.
nostabsplit	Prevents the debug section to be written to an alternate output file with the extension *.stab*. This is the default setting.
nostrip	Does not generate a stripped output file. Thus, the symbol table and relocation information is written in the output file. This option overrides the **-s** flag. This is the default.
nosymbolic	Assigns the **nosymbolic** attribute to most symbols exported without an explicit attribute. For more information, see "Attributes of Exported Symbols" on page 1882. The default is the **nosymbolic-** option.

Item	Description
nosymbolic-	Assigns the **nosymbolic-** attribute to most symbols exported without an explicit attribute. For more information, see "Attributes of Exported Symbols." This is the default.
notextro or **nro**	Does not check to ensure that there are no load time relocation entries for the text section of the output object file. This is the default.
notmprelname	The binder does not check for general instantiations. **Note:** This option is only needed for 32-bit mode. This option is ignored when building 64-bit objects.
notypchk	Does not check function-parameter types between external functional calls. The default is the **typchk** option.
nov	Does not write additional information to the load map file. This option is the default and overrides the **-v** flag.
noweaklocal	Resolves weak symbols using normal search order. This option overrides the **weaklocal** option. It is the default option.
nox	Does not make the output file executable. Neither the auxiliary header nor the loader section is written. Flags and options that specify values written in the auxiliary header or loader section have no effect when this option is used. The default is the **x** option.
nro	Functions the same as the **notextro** option.
nso	Functions the same as the **noautoimp** option.
order_file:*FileID*	Maps symbols that are listed in *FileID* in the specified order. The symbols that are listed in the file are mapped before other symbols of the same storage-mapping class. Function names that are specified in the file must start with a dot because a function name without a dot denotes a function descriptor.
order:*Specification*	Controls the order in which some symbols are mapped in the output file. The specifications can be: **toc:fileref** If the **-border_file** flag is used, then the TOC symbols that are referenced by any function listed in the order file, are mapped before other TOC symbols. Otherwise, this specification is ignored. **toc:nofileref** Does not order TOC symbols that are based on the order file. **notoc** Does not order TOC symbols in any special way.
pD:*Origin*	Specifies *Origin* as the address of the first byte of the file page containing the beginning of the data section. For example, if the data section begins at offset `0x22A0` in the object file, and `pD:0x20000000` is specified, the first byte of the data section is assigned address `0x200002A0`. This assumes a page size of 4096 (`0x1000`) bytes. **Note:** If both the **-bpD** and **-D** flags are specified, the latter flag takes precedence.

Item	Description
>|>|**plugin:***path*	Specifies the full path of a compiler plug-in. This option is typically generated by a compiler, if required. If you want to cancel a previously specified plug-in path, do not specify the *path* variable, that is, `-bplugin:` If an input file is a bitcode file or an archive that contains a bitcode file, and a plug-in path is not specified, the **ld** command fails. |<|<
>|>|**plugin_opt:***plugin-option*	Specifies an option to be passed to the compiler plug-in. You can specify the `plugin_opt` option multiple times. This option is typically generated by a compiler.|<|<
pT:*Origin*	Specifies *Origin* as the address of the first byte of the file page containing the beginning of the text section. For example, if the text section begins at offset `0x264` in the object file, and `pT:0x10000000` is specified, the first byte of the text section is assigned address `0x10000264`. **Note:** If both the **-bpT** and **-T** flags are specified, the latter flag takes precedence. See the **-T** flag for additional information.
quiet	Does not write binder subcommands and their results to standard output. This is the default.

Item	Description
R:*FileID*	Functions the same as the **map:***FileID* option.
r or **reorder**	Reorders CSECTs as part of the **save** command processing. The reorder process arranges CSECTs of the same storage-mapping class by proximity of reference. This is the default.
ras	Sets a flag in the output module's auxiliary header to signify that the module is both storage key safe and recovery safe. For more information abput how to make a kernel extension be key safe and recovery safe, see*Kernel Extensions and Device Support Programming Concepts* .
rename:*Symbol, NewName*	Renames the external symbol *Symbol* to *NewName*. In effect, it is as if all definitions and references to *Symbol* in all object files were renamed to *NewName* before the files were processed. By default, symbols are not renamed.
reorder	Functions the same as the **r** option.
ro or **textro**	Ensures that there are no load time relocation entries for the text section of the resultant object file. The default is the **nro** option.

Item	Description
rtl	Enables run-time linking for the output file. This option implies the **rtllib** and **symbolic** options. When dynamic mode is in effect (see the **dynamic** and **static** options), the **rtl** option allows input files specified with the **-l** flag to end in **.so** as well as in **.a**. All input files that are shared objects are listed as dependents of your program in the output files loader section. The shared objects are listed in the same order as they were specified on the command line. A shared object contained in an archive is only listed if the archive specifies automatic loading for the shared object member. You specify automatic loading for an archive member **foo.o** by creating an import file with the following lines: `# autoload` `#! (foo.o)` and adding the import file as a member to the archive. You may also specify automatic loading for an archive member foo.o by using the **-bautoload** option: `-bautoload:<archive_name>(foo.o)` You may specify additional archive members with additional **-bautoloads**. If the first line of an import file begins with `#!` (number sign, exclamation point), you can specify the file on the command line as an ordinary *InputFile*. Otherwise, you must use the **-bI** or **-bimport** option to specify the import file.
rtllib	Includes a reference to the run-time linker. The run-time linker is defined in **librtl.a**, and an implicit **-lrtl** flag is added automatically to the command line. This option (implied by the **rtl** option) must be used when linking a main program or no run-time linking will occur. Shared objects do not have to be linked with this option. The default is the **nortllib** option unless the `svr4` option is used.
rwexec	Specifies that the execute permissions of the process' private data areas will be determined according to the system's `sed_config` setting. This is the default.
rwexec_must	Specifies that the process' private data areas will have execute permission, regardless of the system's `sed_config` setting.
S:*Number*	Functions the same as the **-S** flag.
scalls:*FileID*	Writes an address map of the object file to *FileID*. Symbols are listed alphabetically. For each symbol listed in the map, references from the symbol to the other symbols are listed. By default, no file is produced. To learn more about the **scalls** option, see “Address Maps” on page 1883.
shared	Functions the same as the **dynamic** option.
shrsymtab	In 64-bit mode, sets the _AOUT_SHR_SYMTAB flag in the XCOFF auxiliary header. If *File* is a 64-bit program, a shared symbol table is created when the program is run. If *Flag* is a 64-bit object but not a 64-bit program, the _AOUT_SHR_SYMTAB flag can be set, but has no effect at runtime. In 32-bit mode, this flag is ignored. The default is **noshrsymtab**.
smap:*FileID*	Writes an address map of the object file to *FileID*. Symbols are listed alphabetically. By default, no file is produced. To learn more about the **smap** option, see "Address Maps" that follows.

Item	Description
so	Functions the same as the **autoimp** option.
stabcmpct:*Level*	Specifies the level of compaction for stabstrings in the debug section. Stabstrings are strings that are longer than eight characters. Each substring in the symbol table has its own offset in the debug section. The following values are valid for *Level*: **0** Does not compact. Separate copies of duplicate stabstrings are written to the debug section. **1** Deletes duplicates. Each stabstring is written once to the `.debug` section. Duplicate stabstrings in the symbol table specifies the same offset into the debug section. **2** Renumbers the stabstrings and deletes most duplicates. (In some instances, multiple stabstrings can exist. They describe the same type but use different type numbers.) The scope of a type number is the entire output file, rather than a single input file as indicated by a C_FILE symbol table entry. If the binder does not recognize a stabstring, it returns an error message and the resulting executable file does not have valid stabstrings. The rest of the file is unaffected by the error.
stabsplit	Causes the debug section to be written to an alternate output file with the extension *.stab*.
stackpsize:*psize*	Requests *psize* page sizes in bytes for process main thread stack. The value can be specified as a decimal, hexadecimal, or octal number. The number specifications are the same as in C programming language. Additionally, the page sizes can be specified as a number followed by a one-character suffix: • k or K for kilo or 0x400 bytes • m or M for mega or 0x100000 bytes • g or G for giga or 0x40000000 bytes • t or T for tera or 0x10000000000 bytes • p or P for peta or 0x4000000000000 bytes • x or X for exo or 0x1000000000000000 bytes For example, either `-b stackpsize:16k` or `-b stackpsize:0x4000` will request 0x4000 for process main thread stack and set the F_VARPG bit in the XCOFF header.
static	Causes the linker to process subsequent shared objects in static mode. In static mode, shared objects are statically linked in the output file.

Item	Description
svr4	This option changes the meaning of some other options on the command line and the standard behavior of the linker. It has the following effect on the linker: • `-b rtl` is set • `-b rtllib` is set only when building an executable or if not set explicitly to `-b nortllib` • `-b` symbolic is set only when building an executable or if not set explicitly by one of `-b symbolic`, `-b nosymbolic`, or `-b nosymbolic-` • `-b expfull` is set only when neither `-b E` nor `-b` export are present • `-b noexpall` is set • `-d`, instead of being ignored, is redefined and can assume two values: `-dy` or `-dn` • `-R`, instead of being ignored, takes one suboption that defines the runtime library search path • `-z`, instead of being a synonym of the `-K` option, takes either `defs`, `nodefs`, `multidefs`, `text`, `nowarntext`, or `warntext`, as a suboption • directories specified with the `-L` option are not included in the runtime libraries search path
sxref:*FileID*	Writes an address map of the object file to *FileID*. Symbols are listed alphabetically. For each symbol listed in the map, references to the symbol from other symbols are listed. By default, no file is produced. To learn more about the **sxref** option, see "Address Maps." that follows.
symbolic	Assigns the **symbolic** attribute to most symbols exported without an explicit attribute. For more information, see "Attributes of Exported Symbols" that follows. This is the default when the `svr4` option is used; otherwise, the default is the **symbolic-** option.
textro	Same as the **ro** option.
textpsize:*psize*	Requests *psize* page sizes in bytes for text. The value can be specified as a decimal, hexadecimal, or octal number. The number specifications are the same as in C programming language. Additionally, the page sizes can be specified as a number followed by a one-character suffix: • k or K for kilo or 0x400 bytes • m or M for mega or 0x100000 bytes • g or G for giga or 0x40000000 bytes • t or T for tera or 0x10000000000 bytes • p or P for peta or 0x4000000000000 bytes • x or X for exo or 0x1000000000000000 bytes For example, either `-b textpsize:16k` or `-b textpsize:0x4000` will request 0x4000 for text and set the F_VARPG bit in the XCOFF header.
tmplrename	Specifies that the binder should check for general instantiations. The binder checks for any symbol of the form **__tfNNxxx_***name* and renames the symbol to *name*. The default is **-bnotmplrename**. **Note:** This option is only needed for 32-bit mode. This option is ignored when building 64-bit objects.

Item	Description
typchk	Performs function-parameter type checking between external functional calls. Parameter-type checking information can be included in object files by compilers and assemblers. This is the default. For more information on type checking, see the "XCOFF (a.out) File Format" in *Files Reference*.
weaklocal	Specifies that weak symbols are searched for first in the object files where they are referenced. If the symbols are not found there, the normal search order is resumed.
x	Makes the output file executable, if no errors exist. This is the default option.
X or **xref:***FileID*	Writes an address map of the object file to *FileID*. Symbols are sorted by section and then by address. For each symbol listed in the map, references to the symbol from other symbols are listed. By default, no file is produced. To learn more about the **xref** option, see "Address Maps" that follows.

Run-time Linking

By default, references to symbols in shared objects are bound at link time. That is, the output module associates an imported symbol with a definition in a specific shared object. At load time, the definition in the specified shared object is used even if other shared objects export the same symbol.

You can cause your program to use the run-time linker, allowing some symbols to be rebound at load time. To create a program that uses the run-time linker, link the program with the **-brtl** option. The way that shared modules are linked affects the rebinding of symbols.

You can build shared objects enabled for run-time linking by using the **-G** flag. You can fully enable run-time linking for existing shared objects by relinking them with the **rtl_enable** command, as long as they have not been stripped.

Symbol Visibility

Global and weak symbols in an input object file can be marked with a visibility. Four symbol visibilities are defined.

Table 13. Symbol Visibility

Symbol	Visibility
Internal	Symbol is not exported. The address of the symbol must not be provided to other programs or shared objects, but the linker does not verify this.
Hidden	Symbol is not exported
Protected	Symbol is exported but cannot be rebound (or preempted), even if runtime linking is being used.
Exported	Symbol is exported with the global export attribute.

The visibility of a symbol can be specified in an assembler source file. Some compilers support visibility as well. Consult your compiler documentation for details.

Export files can also be used to specify the visibility for a symbol. Ordinarily, the visibility specified in an export file takes precedence over the visibility specified in the object file. This linker considers symbol visibility when creating the export list for a program or shared object.

Import and export File Format (-bI: and -bE: Flags)

Each line within an import or export file contains the name of a symbol, optionally followed by an address or a keyword. Primary keywords are **svc**, **svc32**, **svc3264**, **svc64**, **syscall**, **syscall32**, **syscall3264**, **syscall64**, **symbolic**, **nosymbolic**, **nosymbolic-**, **list**, **cm**, **bss**, **internal**, **hidden**, **protected**, and **export**. Additional keywords are **weak** and **required**, which can be used in conjunction with another keyword.

In an import file, specifying an address allows a symbol to be mapped to a fixed address, such as an address in a shared memory segment. You can also use one of the keywords **cm**, **bss**, or **weak** to specify the storage class of an imported symbol. When the **autoexp** option is used, the storage class of an imported symbol affects which symbols are automatically exported. If any other keyword is specified in an import file, the keyword is ignored.

In an export file, you can use the **svc**, **svc32**, **svc3264**, **svc64**, **syscall**, **syscall32**, **syscall3264**, or **syscall64** keyword after a function name to indicate that the function is a system call. This is needed when linking kernel extensions. If the output file is not a kernel extension, these keywords are equivalent to the **symbolic** keyword.

You can use the **list** keyword to cause a symbol to be listed in the loader section of the output file, although it will not be marked as an exported symbols. This can be used for applications that want to process some symbols at run time. Listed symbols are not processed by the system loader or the runtime linker.

You can use the **symbolic**, **nosymbolic**, or **nosymbolic-** keyword to associate an attribute with an exported symbol. A symbol address in an export file is ignored. In an export file, the keywords **cm** and **bss** are equivalent to the **nosymbolic** keyword. The visibility of a symbol can be specified with the **internal, hidden**, **protected**, or **export** keywords. For more information, see.“Attributes of Exported Symbols” on page 1882

The **weak** keyword can be used to specify weak symbol binding, and may be used with another attribute.

Use the **required** keyword to verify that a symbol is defined and not imported. An error is printed for symbols not meeting these criteria.

The **ld** command treats import and export files according to the following guidelines:

- A blank line is ignored.
- A line beginning with an * (asterisk) is a comment and is ignored.
- A line beginning with a # (#, blank space) provides operands to the **setopt** binder subcommand (**-bdbg:***Option*). For example, a line containing `# verbose` causes the binder to list each symbol as it is read from the file. These option settings are active only while processing the file. The # 32, # 64, # no32, and # no64 options can be used to specify whether the listed symbols should be used for 32-bit links, 64-bit links, or both.

  32-bit and 64-bit Import File Options

Item	Description
32	This option is used in an import or export file to specify that subsequent symbols should be processed when linking in 32-bit mode, but ignored when linking in 64-bit mode. If no **32** or **64** option is specified, all symbols are processed in both 32- and 64-bit modes.
64	This option is used in an import or export file to specify that subsequent symbols should be processed when linking in 64-bit mode, but ignored when linking in 32-bit mode. If no **32** or **64** option is specified, all symbols are processed in both 32- and 64-bit modes.
no32 or **no64**	Override a previous **32** or **64**. Subsequent symbols are processed in both 32- and 64-bit modes.

- When processing an import file, a line beginning with a #! (#, exclamation point) provides the shared library name to be associated with subsequent import symbols. The line can occur more than once and

applies to subsequent symbols until the next line beginning with `#!` is read. This file name information is placed in the loader section of the XCOFF object file. It is used by the system loader to locate the appropriate object file at execution time. If the import file name is **ipath/ifile** (imember), the file name placed in the loader section is determined based on the import file name and the contents of the `#!` line of the import file, as follows:

Item	Description
#!	(Nothing after the `#!`) Use null path, null file, and null number. This is treated as a deferred import by the system loader.
#! ()	Use `ipath`, `ifile`, and `imember`. This line can be used if the import file is specified as an *InputFile* parameter on the command line. The file must begin with `#!` in this case. This line can also be used to restore the default name if it was changed by another `#!` line.
#! *path*/*file* **(***member***)**	Use the specified path, file, and member.
#! *path*/*file*	Use the specified path and file, and a null member.
#! *file*	Use a null path, the specified file, and a null member. At run time, a list of directories is searched to find the shared object.
#! (*member***)**	Use `ipath`, `ifile`, and the specified member. At run time, a list of directories is searched to find the shared object.
#! *file* **(***member***)**	Use a null path and specified file and member. At run time, a list of directories is searched to find the shared object.
#! .	(A single dot) This name refers to the main executable. Use this file name when you are creating a shared object that imports symbols from multiple main programs with different names. The main program must export symbols imported by other modules, or loading will fail. This import file name can be used with or without the run-time linker.
#! ..	(Two dots) Use this name to list symbols that will be resolved by the run-time linker. Use this file name to create shared objects that will be used by programs making use of the run-time linker. If you use a module that imports symbols from **..** in a program that was not linked with the **rtllib** option, symbols will be unresolved, and references to such symbols will result in undefined behavior.

To automatically load archive members when the **-brtl** option is used, you can create an import file as follows. If **shr.so** is a shared object in an archive, create an import file:

```
# autoload
#! (shr.so)
```

You can list additional member names on additional lines, if appropriate. You do not need to list symbol names in the import file because the symbols imported from **shr.so** will be read from **shr.so** itself.

For more information on creating a shared library, see "How to Create a Shared Library" in *General Programming Concepts: Writing and Debugging Programs*. For more information on loading and binding, see the **load** subroutine.

Attributes of Exported Symbols

When you use run-time linking, a reference to a symbol in the same module can only be rebound if the symbol is exported with the proper attribute. References to symbols with the **symbolic** attribute cannot be rebound. References to symbols with the **nosymbolic** attribute can be rebound. References to symbols with the **nosymbolic-** attribute can be rebound if the symbols are variables. For function symbols, calls using a function pointer can be rebound, while direct function calls cannot be rebound. The **nosymbolic-** attribute is the default and is provided for compatibility with previous versions of the operating system, but its use is not recommended.

If you are not using the run-time linker, *avoid* using the **nosymbolic** attribute because intra-module function calls will be made indirectly through a function descriptor using global-linkage code. Otherwise, the attribute of exported symbols has no effect for modules used with programs that do not use the run-time linker.

You can specify an explicit export attribute for symbols listed in an export file. Most symbols without an explicit attribute are exported with the default export attribute, as specified with the **symbolic**, **nosymbolic**, or **nosymbolic-** options.

If a symbol is listed in an export file without a keyword, and the visibility of the symbol is specified in an input file, the symbol's visibility is preserved. An input symbol's visibility can be overridden by using the **internal, hidden**, **protected**, or **export** keyword.

The **weak** export attribute will mark the associated symbol's mapping type with L_WEAK in the loader section.

Imported symbols may only have the **weak** export attribute. If a symbol is imported from another module, all references to the symbol can be rebound. However, if a symbol is imported at a fixed address, all references are bound to this fixed address and cannot be rebound by the run-time linker. The system loader must resolve deferred imports. The run-time linker never resolves or rebinds references to deferred imports.

For exports of non-imported symbols, the following rules are used.

- If a symbol has the **list** attribute, it is listed in the loader section symbol table, but the **L_EXPORT** flag is not set in the symbol table entry. The run-time linker ignores such symbols.
- If a symbol was exported with an explicit attribute, or with an explicit visibility, the explicit attribute or visibility is used.
- If the symbol is a BSS symbol, it is exported with the **nosymbolic** attribute.
- Otherwise, the symbol is exported with the global attribute, as specified by the **symbolic**, **nosymbolic**, or **nosymbolic-** option. The default global attribute is **nosymbolic-**.

Address Maps

The **ld** command generates address maps, listing the layout of symbols in the output object file. If you use the **map** (or **R**) option, unresolved symbols and imported symbols are listed first, followed by the symbols in each section in address order. If you use the **calls** (or **C**) option, each symbol that is listed is followed by a list of references from that symbol to other symbols. If you use the **xref** (or **X**) option, each symbol that is listed is followed by a list of references to that symbol from other symbols. If you use the **smap**, **scalls**, or **sxref** option, the address map contains the same information as listed by the **map**, **calls**, or **xref** option, respectively, but symbols are listed in alphabetical order.

Internal symbols, with a storage class C_HIDEXT, are printed with the characters < and > (angle brackets) surrounding the symbol name. Names of external symbols, with a storage class C_EXT, are printed without the angle brackets, and those with a storage class of C_WEAKEXT, are printed with the characters { and } surrounding the symbol name.

Information listed about each symbol includes:

- An indication of whether the symbol is imported, exported, or the entry point. An * (asterisk) is used to mark the entry point, **I** is used to mark imported symbols, and **E** is used to mark exported symbols.
- Its address (except for imported symbols)
- Length and alignment (for CSECTs and BSS symbols)
- Storage-mapping class
- Symbol type
- Symbol number (used to differentiate between symbols of the same name)
- Symbol name
- Input file information

Storage-mapping classes and symbol types are defined in the **/usr/include/syms.h** file. In the address maps, only the last two characters are shown, except that storage-mapping class **XMC_TC0** is shown as **T0**.

The input file information depends on the type of input file. For object files, source files names obtained from C_FILE symbols table entries are listed. If the object is from an archive file, the object file name is listed in the following format:

```
ArchiveFileName[ObjectName]
```

A shared object name is listed between { } (braces). If a shared object is defined by an import file, the name of the import file is listed before the shared object name.

Import symbols have a symbol type of ER, but they have associated file input information. Undefined symbols are also listed with a symbol type of ER, but all other columns, except the symbol number, are left blank.

The **-T** and **-D** flags (or **pT** or **pD** options) affect the addresses printed in these address maps. For machine-level debugging, it is helpful to choose address so that symbols are listed with the same addresses that they have at run time. For a 32-bit program that does not use privately loaded shared objects, you can choose the proper addresses by specifying the `-bpT:0x10000000` and `-bpD:0x20000000` options. These options are defined by default in the **/etc/xlC.cfg** or **/etc/vac.cfg** file.

Environment Variables

The following environment variables affect the execution of the **ld** command:

Item	Description
LIBPATH	If **LIBPATH** is defined, its value is used as the default library path information. Otherwise, the default library path information is **/usr/lib:/lib**. If no **-L** flags are specified and no **-blibpath** option is specified, the default library path information is written in the loader section of the output file. Regardless of any options specified, **LIBPATH** is not used when searching for libraries that are specified from the command line.
TMPDIR	If the output file already exists or it is on a remote file system, the **ld** command generates a temporary output file. The temporary output file is created in the directory specified by **TMPDIR**. If **TMPDIR** is not defined, the temporary output file is created in the **/tmp** directory if the output file is remote, or in the same directory as the existing output file.
OBJECT_MODE	If neither the **-b32** nor **-b64** option is used, the **OBJECT_MODE** environment variable is examined to determine the linking mode. If the value of **OBJECT_MODE** is **32** or **64**, 32-bit or 64-bit mode is used, respectively. If the value is **32_64** or any other value, the linker prints an error message and exits with a non-zero return code. Otherwise, 32-bit mode is used.

Examples

1. To link several object files and produce an **a.out** file to run under the operating system, type:

   ```
   ld /usr/lib/crt0.o pgm.o subs1.o subs2.o -lc
   ```

   The `-lc` (lowercase letter L) links the **libc.a** library. A simpler way to accomplish this is to use the `cc` command (the compiler) to link the files as follows:

   ```
   cc  pgm.o  subs1.o  subs2.o
   ```

2. To specify the name of the output file, type:

   ```
   cc  -o pgm  pgm.o  subs1.o  subs2.o
   ```

   This creates the output in the file `pgm`.

3. To relink `pgm` if only the object file **subs1.o** has changed, type:

   ```
   cc -o pgm subs1.o pgm
   ```

   The CSECTs that originally came from object files **pgm.o** and **subs2.o** are read from the file **pgm**. This technique can speed the linking process if a program consists of many input files, but only a few files change at a time.

4. To link with library subroutines, type:

```
cc  pgm.o  subs1.o  subs2.o  mylib.a  -ltools
```

This links the object modules `pgm.o`, `subs1.o`, and `subs2.o`, the subroutines from the `mylib.a` archive, and the subroutine from the library specified by **-l** (lowercase letter L) flag. (This means the **/usr/lib/libtools.a** file).

5. To generate a shared object, type:

```
ld -o shrsub.o subs1.o subs2.o -bE:shrsub.exp -bM:SRE -lc
```

This links the object files **subs1.o**, **subs2.o**, and the subroutines from the library **libc.a** specified by **-lc** flag. It exports the symbols specified in the file **shrsub.exp** and stores the linked shared object in file **shrsub.o.** The `-bM:SRE` sets the shared object flag in the linked object file.

6. To link with the shared object `shrsub.o` generated previously, type:

```
cc -o pgm pgm.o shrsub.o -L '.'
```

This links the object file **pgm.o** with the exported symbols of `shrsub.o`. The linked output is stored in the object file `pgm`. The `-L '.'` adds the current directory to the library search path that the system loader uses to locate the `shrsub.o` shared object. At run time, this program is loaded only if it is run from a directory containing an instance of the `shrsub.o` file or if the `shrsub.o` file is found in the **/usr/lib** standard library directory. To allow the program to be run from anywhere, use the option **-L** **`pwd`**.

The list of directories searched by the system loader can be seen using the **dump** command.

7. To link a program using the **libc.a** library as a non-shared library, type:

```
cc -o pgm pgm.o -bnso -bI:/lib/syscalls.exp
```

This links `pgm.o` with the necessary support libraries and names the output file `pgm`. For the `cc` command, the **libc.a** library is a necessary support library and is usually link-edited to the user's program as a shared library. In this example, the `-bnso` option directs the **ld** command to link with the **libc.a** library as a non-shared library, and the `-bI:/lib/syscalls.exp` directs the **ld** command to import the system call functions that are actually contained in the kernel or **/usr/lib/boot/unix** file. Whenever linking with the `-bnso` option, any symbols that were both imported and exported (that is, passed through) in a shared object must be explicitly imported, as is done by the `-bI:/lib/syscalls.exp` option in this example.

Note: Any time that **/usr/lib/libc.a** is linked non-shared, the flag **-bI:/lib/syscalls.exp** must be used. The application can also have to be linked again whenever an updated release of the operating system is installed. Any application that is statically linked is *not* binary portable from any fix or release level to any other fix or release level.

8. To enable all of the randomization attributes, enter:

```
-baslr
```

If text-section relocations exist, do not enable text and data randomization.

9. To enable randomization for text and stack only, enter:

```
-baslr:ts
```

Files

Item	Description
/usr/bin/ld	Contains the **ld** command.
/**usr**/**lib**/**lib*.a**	Specifies libraries used for linking programs.
a.out	Specifies the default output file name.

ldapgetusrattr Command

Purpose

Displays the value of an attribute for an LDAP user from the LDAP configured directory server.

Note: If an LDAP user is created with a UID value that is greater than 2^{31}, the **ldapgetusrattr** command displays it as a negative number.

Syntax

`ldapgetusrattr <user_name> <ldap_attribute_name>`

Description

The **ldapgetusrattr** command displays the value of an attribute for an LDAP user from the LDAP configured directory server. The **ldapgetusrattr** command queries the LDAP directory server by using the `secldapclntd` daemon and prints the result to the standard output (stdout) file.

Exit Status

This **ldapgetusrattr** command returns 0 after successful completion and returns nonzero value on failure. On failure, one for the following error messages are written to the standard error (stderr) file:

*Table 14. Exit values of the **ldapgetusrattr** command*

Item	Description
0	Indicates that the command completed successfully.
EIO	Indicates a connection error with LDAP.
EINVAL	Indicates that the command arguments are invalid or do not follow the expected usage.
EPERM	Indicates that you do not have permissions to run the command.
ENOMEM	Indicates insufficient memory to run the command.

Security

The **ldapgetusrattr** command is owned by the root user and the command has access permissions of 500. A root user or a user with the role that has the `aix.security.ldap` authorization can run the **ldapgetusrattr** command.

Example

1. To display the value of the `passwordminlength` attribute for the LDAP user `foo`, run the following command:

   ```
   ldapgetusrattr foo passwordminlength
   ```

   An output that is similar to the following example is displayed:

   ```
   8
   ```

2. To display the value of the `sshPublicKey` attribute for the LDAP user `foo`, run the following command:

```
ldapgetusrattr foo sshPublicKey
```

An output that is similar to the following example is displayed:

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDlXAIquGStc6P07u7Y+3e5BeP608AxxCvCICGd/
1V7jzzjKXI1o4ktFPqEUilHqw7RAgj
zdXRG9jMeo2rg8oKye10CtswZGYunCDiFrBtw7cPSHcE1DCFW0yVu
70I5pUwVgYeVzQIWI8t28PdAvJnfCmlQQZxQrgGk3RimNVrRIFFHKgvbvG3Ck32K
ChRSpzOFiI14ZaGgz1qvW1GAM4YD1zQ3pk/
E5Gs80FaEuqxiDhmWoW7joA5SmkBcmz4UZgPEns0nZnIPDAYPPHBD482rKf1e0qymr9F1p5gIPK70Qi6fr
ilRdYK9e7ybql6n8KzgJWGgBbZqkjyEJn/
XeOrLhMfiEeqcNC3Mq3lg2M0tBGLojWyZ4QSIUCQXsjeRV74E1SuBOzrl4EBhiqJ8VQNr4sMfb1wXKPF6DO
ivGY2w7tbthp7LE94fKAnYyMHEg67LQXVoaGW+EUcj6kJVnW1Hqly6Q2bMHmbiLHRudb+CAa8GUFuWsDxVmUn/
PjyIAWc= vc17user@alp052-vc17.aus.stglabs.ibm.com
```

Restrictions

The **ldapgetusrattr** command is dependent on the `secldapclntd` daemon to query the LDAP server.

ldd Command

Purpose

Lists dynamic dependencies.

Syntax

ldd *FileName*

Description

The **ldd** command lists the path names of all dependencies. The command will report dependencies on only valid XCOFF files.

Parameters

Item	Description
FileName	Specifies the file whose dependencies will be listed.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Examples

1. To display dependencies on **/usr/bin/dbx**, enter:

```
ldd /usr/bin/dbx
```

The output looks like the following:

```
/usr/bin/dbx needs:
        /usr/lib/libc.a(shr.o)
        /usr/lib/libdbx.a(shr.o)
        /unix
```

```
        /usr/lib/libcrypt.a(shr.o)
        /usr/lib/libpthdebug.a(shr.o)
```

Files

Item	Description
/usr/bin/ldd	Contains the **ldd** command.

ldedit Command

Purpose

Modifies an **XCOFF** executable file header.

Syntax

ldedit -b *Option* [**-V**] *File*

Description

You can use the **ldedit** command to modify various fields in an **XCOFF** header or the auxiliary header of an executable file. The **ldedit** command makes it possible to mark or unmark an application as a 'large page data' program. The **ldedit** command also makes it possible to add or modify the values of MAXDATA and MAXSTACK without relinking.

The format of the **-b** flag is similar to the format used by the link editor, the **ld** command. The **-b** flag can be used multiple times on the command line.

If no flags are specified, the **ldedit** command displays a usage message using the standard error output.

Flags

Item	Description
-b*Option*	Modifies an executable as specified by *Option*. The possible values for *Option* are:

aslr or aslr:*[[+-][tdsmp]*]**
aslr or **aslr**:*[[+-][tdsmp]*]* specifies the address space layout randomization for the program. The **aslr** option enables all of the randomization attributes when only this option is used.

If the **aslr** option is followed by a colon, individual attributes can be enabled. The plus sign (*+*) enables randomization and the minus sign (-) disables randomization. The following attributes can be used to specify text, data, stack, mmap, and private-libraries: *t*, *d*, *s*, *m*, and *p*. For the attributes that are not listed, the randomization setting remains disabled.

Note: The *m* and *p* attributes cannot be specified for 32-bit programs.

Some programs are compiled and linked such that relocatable addresses are mapped into the text section. This requires the relocation of the addresses when the program is run. These programs fail if randomization is enabled for text or data. As a result, when text-section relocations exist, text and data randomization are not enabled unless the *t* and *d* attributes are explicitly specified when using the **aslr** option.

noaslr
Clears the randomization attribute settings of the program.

datapsize:*psize*
Requests *psize* page sizes in bytes for data. The value can be specified as a decimal, hexadecimal, or octal number. The number specifications are the same as in C programming language. Additionally, the page sizes can be specified as a number followed by a one-character suffix:

- k or K for kilo or 0x400 bytes
- m or M for mega or 0x100000 bytes
- g or G for giga or 0x40000000 bytes
- t or T for tera or 0x10000000000 bytes
- p or P for peta or 0x4000000000000 bytes
- x or X for exo or 0x1000000000000000 bytes

For example, either `-b datapsize:16k` or `-b datapsize:0x4000` will request 0x4000 for data and set the F_VARPG bit in the XCOFF header. It is accomplished by setting the corresponding member of the auxiliary header to the logarithm base 2 of the given value *psize*. If the value is different from 0, the F_VARPG bit of the XCOFF header's `f_flags` member is also set. Otherwise, this bit is cleared.

forkpolicy:*policy*
Sets the _AOUT_FORK_POLICY flag in the XCOFF auxiliary header, if *File* is a 64-bit program. If *policy* is `cor`, the _AOUT_FORK_COW is also set, requesting the use of the copy-on-write forktree policy. If *policy* is `cow`, the _AOUT_FORK_COW flag is cleared, requesting the use of the copy-on-reference forktree policy. If *File* is a 32-bit program, no change is made.

noforkpolicy
Resets the _AOUT_FORK_POLICY and _AOUT_FORK_COR flags in the XCOFF auxiliary header, if *File* is a 64-bit program. The default forktree policy is used, unless a forktree policy is specified with the VMM_CNTRL environment variable. If *File* is a 32-bit program, no change is made.

lpdata
Marks a file as a 'large page data' executable.

nolpdata
Unmarks a file as a 'large page data' executable.

noshrsymtab
Clears the _AOUT_SHR_SYMTAB flag in the XCOFF auxiliary header. If *File* is a 32-bit object, no change is made.

M:*<modtype>*
Updates the module-type field and the shared object flag in the file. The **F_SHROBJ** flag is set in the XCOFF header when the module type begins with *S* character and is 3 characters long.

maxdata:*value*
Sets the MAXDATA value. *value* is an octal number when it starts with `0`, a hexadecimal number when it starts with `0x`, and a decimal number in all other cases.

maxdata:*value*/dsa
Sets the MAXDATA value and the DSA bit. *value* is an octal number when it starts with `0`, a hexadecimal number when it starts with `0x`, and a decimal number in all other cases.

maxstack:*value*
Sets the MAXSTACK value. *value* is an octal number when it starts with `0`, a hexadecimal number when it starts with `0x`, and a decimal number in all other cases.

norwexec
Marks a file's writable and mappable sections and stack as non-executable.

rwexec
Marks a file's writable and mappable sections and stack as executable.

shrsymtab
Sets the _AOUT_SHR_SYMTAB flag in the XCOFF auxiliary header. If *File* is a 64-bit program, a shared symbol table is created when the program is run. If *Flag* is a 64-bit object but not a 64-bit program, the _AOUT_SHR_SYMTAB flag can be set, but has no effect at runtime. If *File* is a 32-bit object, no change is made.

Item	Description
	stackpsize:*psize* Requests *psize* page sizes in bytes for process main thread stack. The value can be specified as a decimal, hexadecimal, or octal number. The number specifications are the same as in C programming language. Additionally, the page sizes can be specified as a number followed by a one-character suffix: • k or K for kilo or 0x400 bytes • m or M for mega or 0x100000 bytes • g or G for giga or 0x40000000 bytes • t or T for tera or 0x10000000000 bytes • p or P for peta or 0x4000000000000 bytes • x or X for exo or 0x1000000000000000 bytes For example, either `-b stackpsize:16k` or `-b stackpsize:0x4000` will request 0x4000 for process main thread stack and set the F_VARPG bit in the XCOFF header. It is accomplished by setting the corresponding member of the auxiliary header to the logarithm base 2 of the given value *psize*. If the value is different from 0, the F_VARPG bit of the XCOFF header's `f_flags` member is also set. Otherwise, this bit is cleared.
	textpsize:*psize* Requests *psize* page sizes in bytes for text. The value can be specified as a decimal, hexadecimal, or octal number. The number specifications are the same as in C programming language. Additionally, the page sizes can be specified as a number followed by a one-character suffix: • k or K for kilo or 0x400 bytes • m or M for mega or 0x100000 bytes • g or G for giga or 0x40000000 bytes • t or T for tera or 0x10000000000 bytes • p or P for peta or 0x4000000000000 bytes • x or X for exo or 0x1000000000000000 bytes For example, either `-b textpsize:16k` or `-b textpsize:0x4000` will request 0x4000 for text and set the F_VARPG bit in the XCOFF header. It is accomplished by setting the corresponding member of the auxiliary header to the logarithm base 2 of the given value *psize*. If the value is different from 0, the F_VARPG bit of the XCOFF header's `f_flags` member is also set. Otherwise, this bit is cleared.
-V	Prints the version of the **ldedit** command on the standard error output.

Examples

1. To request system-selected page sizes for text, data, and stacks, enter:

   ```
   ldedit -b textpsize:0 -b datapsize:0 -b stackpsize:0
   ```

   This clears the F_VARPG bit in the XCOFF header.
2. To enable all of the randomization attributes, enter:

   ```
   -baslr
   ```

   If text-section relocations exist, do not enable text and data randomization.
3. To enable the randomization for text, disable the randomization for stack, and leave the other values unchanged, enter:

   ```
   -baslr:+t-s
   ```

learn Command

Purpose

Provides computer-aided instruction for using files, editors, macros, and other features.

Syntax

learn[**-** *Directory*] [*Subject*[*LessonNumber*]]

Description

The **learn** command provides computer-aided instruction for using files, editors, macros, and other features. The first time you invoke the command, the system provides introductory information about the **learn** command. Otherwise, the **learn** command begins at the point where you left the last **learn** command session.

You can bypass the default action of the **learn** command by specifying the *Subject* parameter. The **learn** command starts with the first lesson of the subject you specify. You can specify any of the following subjects:

- Files
- Editors
- More files
- Macros
- EQN (the enquiry character)
- C (the language)

> **Note:** You can only run the EQN lesson on a hardcopy terminal that is capable of 1/2 line motion. The **/usr/share/lib/learn/eqn/Init** file contains a detailed list of the supported terminals.

When you enter the **learn** command, the system searches the **/usr/share/lib/learn** directory for the appropriate lesson file. Use the **-***Directory* flag to identify a different search directory.

Subcommands

- The **bye** subcommand terminates a **learn** command session.
- The **where** subcommand tells you of your progress; the **where m** subcommand provides more detail.
- The **again** subcommand re-displays the text of the lesson.
- The **again** *LessonNumber* subcommand lets you review the lesson.
- The **hint** subcommand prints the last part of the lesson script used to evaluate a response; the **hint m** subcommand prints the entire lesson script.

Parameters

Item	Description
-*Directory*	Allows you to specify a different search directory. By default, the system searches for lesson files in the **/usr/share/lib/learn** directory.
LessonNumber	Identifies the number of the lesson.
Subject	Specifies the subject you want instruction on.

Examples

To take the online lesson about files, enter:

```
learn files
```

The system starts the **learn** program and displays instructions for how to use the program.

Files

Item	Description
/usr/share/lib/learn	Contains the file tree for all dependent directories and files.

Item	Description
/tmp/pl*	Contains the practice directories.
$HOME/.learnrc	Contains the startup information.

leave Command

Purpose

Reminds you when you have to leave.

Syntax

leave [[**+**] *hhmm*]

Description

The **leave** command waits until the specified time and then reminds you that you have to leave. You are reminded at 5 minutes and at 1 minute before the actual time, again at that time, and at every minute thereafter. When you log off, the **leave** command exits just before it would have displayed the next message.

If you do not specify a time, the **leave** command prompts with `When do you have to leave?` A reply of `newline` causes the **leave** command to exit; otherwise, the reply is assumed to be a time. This form is suitable for inclusion in a **.login** or **.profile** file.

The **leave** command ignores interrupt, quit, and terminate operations. To clear the **leave** command, you should either log off or use the **kill-9** command and provide the process ID.

Flags

Item	Description
+	Specifies to set the alarm to go off in the indicated number of hours and minutes from the current time.
hhmm	Specifies a time of day in hours and minutes (based on a 12- or 24-hour clock) or, if preceded by the **+**, a set number of hours and minutes from the current time for the alarm to go off. All times are converted to a 12-hour clock and assumed to relate to the next 12 hours.

Examples

To remind yourself to leave at 3:45, enter:

```
leave 345
```

lecstat Command

Purpose

Displays operational information about an Asynchronous Transfer Mode network protocol (ATM) Local Area Network (LAN) Emulation Client.

Syntax

lecstat [**-a** **-c** **-q** **-r** **-s** **-t** **-v**] Device_Name

Description

This command displays ATM LAN Emulation Client (LEC) operational information gathered by a specified LEC device. If a LEC device name is not entered, statistics for all available LEC's appear. Select a flag to narrow down your search results. You can display specific categories of information such as Configuration, LE_ARP Cache Entries, Virtual Connections, and Statistics, or you can choose to display all of the information categories.

You can also toggle debug tracing on or off and reset statistics counters.

Parameters

Item	Description
Device_Name	The name of the LE Client device, for example, *ent1*.

Flags

Item	Description
-a	Requests all the LE Client information. This flag does not reset statistics counters or toggle trace.
-c	Requests the configuration.
-q	Requests the LE_ARP cache.
-r	Resets the statistics counters after reading.
-s	Requests the statistics counters.
-t	Toggles full debug trace on or off.
-v	Requests the list of virtual connections.

The following information appears for all valid calls and contains the following fields:

Device Type
Displays a description of the LAN Emulation Client (example: Ethernet or Token Ring)

LAN MAC Address
Displays the LAN Emulation Client's 6-byte Ethernet or Token Ring MAC address.

ATM Address
Displays the LAN Emulation Client's 20-byte Asynchronous Transfer Mode (ATM) address.

Elapsed Time
Displays the real time period which has elapsed since statistics were last reset.

Driver Flags
The current LAN Emulation Client(LEC) device driver NDD status flags. Example status flags:

```
Broadcast       Allowing broadcast packets.
Dead            Requires re-open.
Debug           Internal debug tracing enabled.
Limbo           Attempting ELAN recovery.
Running         Fully operational on the ELAN.
Up              Device has been opened.
```

Configuration Information
Selected with the **-a** or **-c** flags. Displays the network administrator's pre-configured attributes, as well as the current ELAN configuration values as defined by the LANE Servers.

Lane LE_ARP Table Entries
Selected with the **-a** or **-q** flags. Displays the current LE Client ARP cache. Included are the type of entry, it's state, the remote LAN MAC address or route descriptor, the remote ATM address and some descriptive values.

Example Types

```
BUS-PP              Broadcast and Unknown Server (point-to-point).
BUS-MP              Broadcast and Unknown Server (multi-point).
Data                Data (point-to-point).
LES-PP              LE Server (point-to-point).
LES-MP              LE Server (multi-point).
```

Example States

```
Arping          Attempting to locate remote client/server via LE_ARP.
Connected       Fully connected to the remote client/server.
Flushing        Flushing the data path to the client/server.
Known           Remote address is known but no connection yet.
Unknown         Remote address in unknown and not able to LE_ARP yet.
```

Lane Servers and Statistics
Selected with the **-a** or **-s** flags. Displays the current Transmit, Receive, and General statistics for this LE Client, as well as the ATM addresses of the current and available LANE Servers.

Lane connections
Selected with the **-a** or **-v** flags. Displays the current list of virtual connections in use by this LE Client. Included are virtual path and channel values, remote ATM address, and some descriptive values such as whether this connection was started by the remote, whether it is a duplicate connection, or whether the remote station is proxied by another LE Client.

Exit Status

If you specify an invalid *Device_Name*, this command produces error messages stating that it could not connect to the device. Examples of an invalid device error message might be:

```
LECSTAT: No LANE device configured.
LECSTAT: Device is not a LANE device.
LECSTAT: Device is not available.
```

lex Command

Purpose

Generates a C or C++ language program that matches patterns for simple lexical analysis of an input stream.

Syntax

lex [**-C**] [**-t**] [**-v**| **-n**] [*File*...]

Description

The **lex** command reads *File* or standard input, generates a C language program, and writes it to a file named **lex.yy.c**. This file, **lex.yy.c**, is a compilable C language program. A C++ compiler also can compile the output of the **lex** command. The **-C** flag renames the output file to **lex.yy.C** for the C++ compiler.

The C++ program generated by the **lex** command can use either STDIO or IOSTREAMS. If the cpp define _CPP_IOSTREAMS is true during a C++ compilation, the program uses IOSTREAMS for all I/O. Otherwise, STDIO is used.

The **lex** command uses rules and actions contained in *File* to generate a program, **lex.yy.c**, which can be compiled with the **cc** command. The compiled **lex.yy.c** can then receive input, break the input into the logical pieces defined by the rules in *File*, and run program fragments contained in the actions in *File*.

The generated program is a C language function called **yylex**. The **lex** command stores the **yylex** function in a file named **lex.yy.c**. You can use the **yylex** function alone to recognize simple one-word input, or you

can use it with other C language programs to perform more difficult input analysis functions. For example, you can use the **lex** command to generate a program that simplifies an input stream before sending it to a parser program generated by the **yacc** command.

The **yylex** function analyzes the input stream using a program structure called a finite state machine. This structure allows the program to exist in only one state (or condition) at a time. There is a finite number of states allowed. The rules in *File* determine how the program moves from one state to another.

If you do not specify a *File*, the **lex** command reads standard input. It treats multiple files as a single file.

> **Note:** Since the **lex** command uses fixed names for intermediate and output files, you can have only one program generated by **lex** in a given directory.

lex Specification File

The input file can contain three sections: *definitions*, *rules*, and *user subroutines*. Each section must be separated from the others by a line containing only the delimiter, %% (double percent signs). The format is:

```
definitions
%%
rules
%%
user subroutines
```

The purpose and format of each are described in the following sections.

Definitions

If you want to use variables in your rules, you must define them in this section. The variables make up the left column, and their definitions make up the right column. For example, if you want to define D as a numerical digit, you would write the following:

```
D   [0-9]
```

You can use a defined variable in the rules section by enclosing the variable name in {} (braces), for example:

```
{D}
```

Lines in the definitions section beginning with a blank or enclosed in %{ , %} delimiter lines are copied to the **lex.yy.c** file. You can use this construct to declare C language variables to be used in the **lex** actions or to include header files, for example:

```
%{
#include <math.h>
int count;
%}
```

Such lines can also appear at the beginning of the rules section, immediately after the first %% delimiter, but they should not be used anywhere else in the rules section. If the line is in the definitions section of *File*, the **lex** command copies it to the external declarations section of the **lex.yy.c** file. If the line appears in the rules section, before the first rule, the **lex** command copies it to the local declaration section of the **yylex** subroutine in **lex.yy.c**. Such lines should not occur after the first rule.

The type of the **lex** external, **yytext**, can be set to either a null-terminated character array (default) or a pointer to a null-terminated character string by specifying one of the following in the definitions section:

```
%array    (default)
%pointer
```

In the definitions section, you can set table sizes for the resulting finite state machine. The default sizes are large enough for small programs. You may want to set larger sizes for more complex programs.

Item	Description
%a *n*	Number of transitions is *n* (default 5000)
%e *n*	Number of parse tree nodes is *n* (default 2000)
%h *n*	Number of multibyte character output slots (default is 0)
%k *n*	Number of packed character classes (default 1000)
%m*n*	Number of multibyte "character class" character output slots (default is 0)
%n *n*	Number of states is *n* (default 2500)
%o *n*	Number of output slots (default 5000, minimum 257)
%p *n*	Number of positions is *n* (default 5000)
%v *p*	Percentage of slots vacant in the hash tables controlled by **%h** and **%m** (default 20, range 0 <= P < 100)
%z *n*	Number of multibyte character class output slots (default 0)

If multibyte characters appear in extended regular expression strings, you may need to reset the output array size with the **%o** argument (possibly to array sizes in the range 10,000 to 20,000). This reset reflects the much larger number of characters relative to the number of single-byte characters.

If multibyte characters appear in extended regular expressions, you must set the multibyte hash table sizes with the **%h** and **%m** arguments to sizes greater than the total number of multibyte characters contained in the **lex** file.

If no multibyte characters appear in extended regular expressions but you want '.' to match multibyte characters, you must set **%z** greater than zero. Similarly, for inverse character classes (for example, [^abc]) to match multibyte characters, you must set both **%h** and **%m** greater than zero.

When using multibyte characters, the **lex.yy.c** file must be compiled with the **-qmbcs** compiler option.

Rules

Once you have defined your terms, you can write the rules section. It contains strings and expressions to be matched by the **yylex** subroutine, and C commands to execute when a match is made. This section is required, and it must be preceded by the delimiter %% (double percent signs), whether or not you have a definitions section. The **lex** command does not recognize your rules without this delimiter.

In this section, the left column contains the pattern in the form of an extended regular expression, which will be recognized in an input file to the **yylex** subroutine. The right column contains the C program fragment executed when that pattern is recognized, called an *action*.

When the lexical analyzer finds a match for the extended regular expression, the lexical analyzer executes the action associated with that extended regular expression.

Patterns can include extended characters. If multibyte locales are installed on your system, patterns can also include multibyte characters that are part of the installed code set.

The columns are separated by a tab or blanks. For example, if you want to search files for the keyword **KEY**, you can write the following:

```
(KEY) printf ("found KEY");
```

If you include this rule in *File*, the **yylex** lexical analyzer matches the pattern **KEY** and runs the **printf** subroutine.

Each pattern can have a corresponding action, that is, a C command to execute when the pattern is matched. Each statement must end with a `;` (semicolon). If you use more than one statement in an action, you must enclose all of them in `{ }` (braces). A second delimiter, %%, must follow the rules section if you have a *user subroutine* section. Without a specified action for a pattern match, the lexical analyzer copies the input pattern to the output without changing it.

When the **yylex** lexical analyzer matches a string in the input stream, it copies the matched string to an external character array (or a pointer to a character string), **yytext**, before it executes any commands in the rules section. Similarly, the external int, **yyleng**, is set to the length of the matched string in bytes (therefore, multibyte characters will have a size greater than 1).

User Subroutines

The **lex** library defines the following subroutines as macros that you can use in the rules section of the **lex** specification file:

Item	Description
input	Reads a byte from **yyin**.
unput	Replaces a byte after it has been read.
output	Writes an output byte to **yyout**.
winput	Reads a multibyte character from **yyin**.
wunput	Replaces a multibyte character after it has been read.
woutput	Writes a multibyte output character to **yyout**.
yysetlocale	Calls the **setlocale (***LC_ALL***, " ");** subroutine to determine the current locale.

The **winput**, **wunput**, and **woutput** macros are defined to use the **yywinput**, **yywunput**, and **yywoutput** subroutines coded in the **lex.yy.c** file. For compatibility, these **yy** subroutines subsequently use the **input**, **unput**, and **output** subroutines to read, replace, and write the necessary number of bytes in a complete multibyte character.

You can override these macros by writing your own code for these routines in the user subroutines section. But if you write your own, you must undefine these macros in the definition section as follows:

```
%{
#undef input
#undef unput
#undef output
#undef winput
#undef wunput
#undef woutput
#undef yysetlocale
%}
```

There is no **main** subroutine in **lex.yy.c**, because the **lex** library contains the **main** subroutine that calls the **yylex** lexical analyzer, as well as the **yywrap** subroutine called by **yylex( )** at the end of *File*. Therefore, if you do not include **main( )**, **yywrap( )**, or both in the user subroutines section, when you compile **lex.yy.c**, you must enter `cc lex.yy.c -ll`, where `ll` calls the **lex** library.

External names generated by the **lex** command all begin with the preface **yy**, as in **yyin**, **yyout**, **yylex**, and **yytext**.

Finite State Machine

The default skeleton for the finite state machine is defined in **/usr/ccs/lib/lex/ncform**. The user can use a personally configured finite state machine by setting an environment variable **LEXER=***PATH.* The *PATH* variable designates the user-defined finite state machine path and file name. The **lex** command checks the environment for this variable and, if it is set, uses the supplied path.

Putting Blanks in an Expression

Normally, blanks or tabs end a rule and therefore, the expression that defines a rule. However, you can enclose the blanks or tab characters in " " (quotation marks) to include them in the expression. Use quotes around all blanks in expressions that are not already within sets of [ ] (brackets).

Other Special Characters

The **lex** program recognizes many of the normal C language special characters. These character sequences are:

Sequence	Meaning
\a	Alert
\b	Backspace
\f	Form Feed
\n	Newline character (Do not use the actual newline character in an expression.)
\r	Return
\t	Tab
\v	Vertical Tab
****	Backslash
\digits	The character with encoding represented by the one-, two-, or three-digit octal integer specified by **digits**.
\xdigits	The character with encoding represented by the sequence of hexadecimal characters specified by **digits**.
\c	Where **c** is none of the characters listed above, represents the character **c** unchanged.

Note: Do not use **\0** or **\x0** in **lex** rules.

When using these special characters in an expression, you do not need to enclose them in quotes. Every character, except these special characters and the operator symbols, is always a text character.

Matching Rules

When more than one expression can match the current input, the **lex** command chooses the longest match first. When several rules match the same number of characters, the **lex** command chooses the rule that occurs first. For example, if the rules

```
integer    keyword action...;
[a-z]+       identifier action...;
```

are given in that order, and `integers` is the input word, **lex** matches the input as an identifier, because `[a-z]+` matches eight characters while `integer` matches only seven. However, if the input is integer, both rules match seven characters. **lex** selects the keyword rule because it occurs first. A shorter input, such as `int`, does not match the expression integer, and so **lex** selects the identifier rule.

Matching a String Using Wildcard Characters

Because **lex** chooses the longest match first, do not use rules containing expressions like `.*`. For example:

```
'.*'
```

might seem like a good way to recognize a string in single quotes. However, the lexical analyzer reads far ahead, looking for a distant single quote to complete the long match. If a lexical analyzer with such a rule gets the following input:

```
'first' quoted string here, 'second' here
```

it matches:

```
'first' quoted string here, 'second'
```

To find the smaller strings, `first` and `second`, use the following rule:

```
'[^'\n]*'
```

This rule stops after `'first'`.

Errors of this type are not far reaching, because the . (period) operator does not match a new-line character. Therefore, expressions like `.*` (period asterisk) stop on the current line. Do not try to defeat this with expressions like `[.\n]+`. The lexical analyzer tries to read the entire input file and an internal buffer overflow occurs.

Finding Strings within Strings

The **lex** program partitions the input stream and does not search for all possible matches of each expression. Each character is accounted for once and only once. For example, to count occurrences of both `she` and `he` in an input text, try the following rules:

```
she         s++
he          h++
\n          |.          ;
```

where the last two rules ignore everything besides `he` and `she`. However, because `she` includes `he`, **lex** does *not* recognize the instances of `he` that are included in `she`.

To override this choice, use the action **REJECT**. This directive tells **lex** to go to the next rule. **lex** then adjusts the position of the input pointer to where it was before the first rule was executed and executes the second choice rule. For example, to count the included instances of `he`, use the following rules:

```
she                 {s++;REJECT;}
he                  {h++;REJECT;}
\n                  |.                  ;
```

After counting the occurrences of `she`, **lex** rejects the input stream and then counts the occurrences of `he`. Because in this case `she` includes `he` but not vice versa, you can omit the **REJECT** action on `he`. In other cases, it may be difficult to determine which input characters are in both classes.

In general, **REJECT** is useful whenever the purpose of **lex** is not to partition the input stream but to detect all examples of some items in the input, and the instances of these items may overlap or include each other.

Flags

Item	Description
-C	Produces the **lex.yy.C** file instead of **lex.yy.c** for use with a C++ compiler. To get the I/O Stream Library, use the macro, **_CPP_IOSTREAMS**, as well.
-n	Suppresses the statistics summary. When you set your own table sizes for the finite state machine, the lex command automatically produces this summary if you do not select this flag.
-t	Writes **lex.yy.c** to standard output instead of to a file.
-v	Provides a one-line summary of the generated finite-state-machine statistics.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To draw **lex** instructions from the file `lexcommands` and place the output in **lex.yy.c**, use the following command:

```
lex lexcommands
```

2. To create a **lex** program that converts uppercase to lowercase, removes blanks at the end of a line, and replaces multiple blanks by single blanks, including the following in a **lex** command file:

```
%%
[A-Z]   putchar(yytext[0]+ 'a'-'A');
[ ]+$ ;
[ ]+    putchar(' ');
```

Files

Item	Description
/usr/ccs/lib/libl.a	Contains the run-time library.
/usr/ccs/lib/lex/ncform	Defines a finite state machine.

line Command

Purpose

Reads one line from the standard input.

Syntax

line

Description

The **line** command copies one line from standard input and writes it to standard output. It returns an exit value of 1 on an end-of-file and always writes at least a new-line character. Use this command within a shell command file to read from the workstation.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	End-of-file occurred on input.

Examples

To read a line from the keyboard and append it to a file, create a script file as follows:

```
echo 'Enter comments for the log:'
echo ': \c'
line >>log
```

This shell procedure displays the message:

```
Enter comments for the log:
```

and then reads a line of text from the workstation keyboard and adds it to the end of the log. The `echo ': \c'` command displays a colon prompt.

link Command

Purpose

Performs a **link** subroutine.

Syntax

link *File1 File2*

Description

The **link** command performs the **link** subroutine on a specified file. The **link** command does not issue error messages when the associated subroutine is unsuccessful; you must check the exit value to determine if the command completed normally. It returns a value of 0 if it succeeds, a value of 1 if too few or too many parameters are specified, and a value of 2 if its system call is unsuccessful.

> **Attention:** The **link** command allows a user with root user authority to deal with unusual problems, such as moving an entire directory to a different part of the directory tree. It also permits you to create directories that cannot be reached or escaped from. Be careful to preserve the directory structure by observing the following rules:
>
> - Be certain every directory has a . (dot) link to itself.
> - Be certain every directory has a .. (dot dot) link to its parent directory.
> - Be certain every directory has no more than one link to itself or its parent directory.
> - Be certain every directory is accessible from the root of its file system.
>
> **Note:** If the . (dot) entry has been destroyed and the **fsck** command is unable to repair it (a rare occurrence), you can use the **link** command to restore the . (dot) entry of the damaged directory. Use the **link** *Dir Dir/*. command where the *Dir* parameter is the name of the damaged directory. However, use this only as a last resort when the directory is destroyed and the **fsck** command is unable to fix it.

Although the linked files and directories can be removed by the **unlink** command, it's safer to use the **rm** or **rmdir** command.

Examples

To create an additional link for an existing `file1`, enter:

```
link file1 file2
```

Files

Item	Description
/usr/sbin/link	Contains the **link** command.

lint Command

Purpose

Checks C and C++ language programs for potential problems.

Syntax

lint [**-a**] [**-b**] [**-c**] [**-C**] [**-h**] [**-l***Key*] [**-n**] [**-o***Library*] [**-q***DBCS*] [**-p**] [**-t**] [**-u**] [**-v**] [**-w** *Class* [*Class* ...]] [**-x**] [**-MA**] [**-Nd***Number*] [**-Nl***Number*] [**-Nn***Number*] [**-Nt***Number*] [**-I***Directory*] [**-D***Name* [**=***Definition*]] [**-U***Name*] *File* ...

Description

The **lint** command checks C and C++ language source code for coding and syntax errors and for inefficient or non-portable code. You can use this program to:

- Identify source code and library incompatibility.
- Enforce type-checking rules more strictly than does the compiler.
- Identify potential problems with variables.
- Identify potential problems with functions.
- Identify problems with flow control.
- Identify legal constructions that may produce errors or be inefficient.
- Identify unused variable and function declarations.
- Identify possibly non-portable code.

> **Note:** Checking of C++ language files by the **lint** command requires the presence of the C Set++ Compiler package.

The inter-file usage of functions is checked to find functions that return values in some instances and not in others, functions called with varying numbers or types of arguments, and functions whose values are not used or whose values are used but not returned.

The **lint** command interprets file name extensions as follows:

- *File* names ending in **.c** are C language source files.
- *File* names ending in **.C** are C++ language source files.
- *File* names ending in **.ln** are non-ASCII files that the **lint** command produces when either the **-c** or the **-o** flag is used.

The **lint** command warns you about files with other suffixes and ignores them.

The **lint** command takes all the **.c**, **.C**, and **.ln** files and the libraries specified by **-l** flags and processes them in the order that they appear on the command line. By default, it adds the standard **llib-lc.ln** lint library to the end of the list of files. However, when you select the **-p** flag, the **lint** command uses the **llib-port.ln** portable library. By default, the second pass of **lint** checks this list of files for mutual compatibility; however, if you specify the **-c** flag, the **.ln** and **llib-l***x***.ln** files are ignored.

The **-c** and **-o** flags allow for incremental use of the **lint** command on a set of C and C++ language source files. Generally, use the **lint** command once for each source file with the **-c** flag. Each of these runs produces a **.ln** file that corresponds to the **.c** file and writes all messages concerning that source file. After you have run all source files separately through the **lint** command, run it once more, without

the **-c** flag, listing all the **.ln** files with the needed **-l** flags. This writes all inter-file inconsistencies. This procedure works well with the **make** command, allowing it to run the **lint** command on only those source files modified since the last time that set of source files was checked.

The **lint** and **LINT** preprocessor symbols are defined to allow certain questionable code to be altered or removed for the **lint** command. Therefore, the **lint** and **LINT** symbols should be thought of as a reserved word for all code that is planned to be checked by **lint**.

The following comments in a C and C++ language source program change the way the **lint** command operates when checking the source program:

Item	Description
/*NOTREACHED*/	Suppresses comments about unreachable code.
/*VARARGS*Number***/**	Suppresses checking the following old style function declaration for varying numbers of arguments, but does check the data type of the first *Number* arguments. If you do not include a value for *Number*, the **lint** command checks no arguments (*Number*=0). The ANSI function prototypes should use the ellipsis to indicate unspecified parameters rather than this comment mechanism.
/*ARGSUSED*/	Suppresses warnings about function parameters not used within the function definition.
/*LINTLIBRARY*/	If you place this comment at the beginning of a file, the **lint** command does not identify unused functions and function parameters in the file. This is used when running the **lint** command on libraries.
/*NOTUSED*/	Suppresses warnings about unused external symbols, functions and function parameters in the file beginning at its point of occurrence. This is a superset of the **/*LINTLIBRARY*/** comment directive, but applies also to external symbols. It is useful for suppressing warnings about unused function prototypes and other external object declarations.
/*NOTDEFINED*/	Suppresses warnings about used, but undefined external symbols and functions in the file beginning at its point of occurrence.
/*LINTSTDLIB*/	Permits a standard prototype-checking library to be formed from header files by making function prototype declarations appear as function definitions. This directive implicitly activates both the **/*NOTUSED*/** and **/*LINTLIBRARY*/** comment directives to reduce warning noise levels.

The **lint** command warning messages give file name and line number. As each file goes through the first pass, warnings for each file and each line number are reported.

If you have not specified the **-c** flag, the **lint** command collects information gathered from all input files and checks it for consistency. At this point, if it is not clear whether a message stems from a given source file or from one of its included files, the **lint** command displays the source file name followed by a question mark.

ANSI programs that include many standard header files may wish to set the **-wD** flag to reduce the quantity of warnings about prototypes not used, and the **-n** flag to disable checking against the ANSI standard library. For non-ANSI programs, it is advisable to specify the **-wk** flag to reduce the amount of warnings concerning the absence of function prototypes.

Flags

Item	Description
-a	Suppresses messages about assignments of long values to variables that are not long.

Item	Description
-b	Suppresses messages about unreachable break statements.
-c	Causes the **lint** command to produce an **.ln** file for every **.c** file on the command line. These **.ln** files are the product of the first pass of the **lint** command only and are not checked for inter-function compatibility.
-C	Specifies to use the C++ libraries (in the **/usr/lpp//xlC/lib** directory).
-h	Does not try to detect bugs, improper style, or reduce waste.
-l*Key*	Includes the additional **llib-l***Key***.ln** lint library. You can include a lint version of the **llib-lm.ln** math library by specifying **-lm** on the command line or **llib-ldos.ln** library by specifying the **-ldos** flag on the command line. Use this flag to include local lint libraries when checking files that are part of a project having a large number of files. This flag does not prevent the **lint** command from using the **llib-lc.ln** library. The lint library must be in the **/usr/ccs/lib** directory.
-n	Suppresses the check for compatibility with either the standard or the portable lint libraries. This applies for both the ANSI and extended mode libraries.
-o*Library*	Causes the **lint** command to create the **llib-l***Library***.ln** lint library. The **-c** flag nullifies any use of the **-o** flag. The lint library produced is the input that is given to the second pass of the **lint** command. The **-o** flag simply causes this file to be saved in the named lint library. To produce a **llib-l***Library***.ln** without extraneous messages, use the **-x** flag. The **-v** flag is useful if the source files for the lint library are just external interfaces (for example, the way the **llib-lc** file is written). These flag settings are also available through the use of **lint** command comment lines.
-p	Checks for portability to other C language dialects.
-t	Checks for problematic assignments when porting from 32 to 64 bit. Only the following cases are checked: • all shift / mask operations are flagged because some operations that work well in 32-bit may cause problems in 64-bit. • warnings are given for the following type of assignments. `int = long` `int = ptr`
-u	Suppresses messages about functions and external variables that are either used and not defined or defined and not used. Use this flag to run the **lint** command on a subset of files of a larger program.
-v	Suppresses messages about function parameters that are not used.

Item	Description
-w *Class* [*Class...*]	Controls the reporting of warning classes. All warning classes are active by default, but can be individually deactivated by including the appropriate option as part of the *Class* argument. The individual options are listed as: **a** Non-ANSI features. **c** Comparisons with unsigned values. **d** Declaration consistency. **h** Heuristic complaints. **k** Use for K+R type source code. **l** Assignment of long values to variables that are not long. **n** Null-effect code. **o** Unknown order of evaluation. **p** Various portability concerns. **r** Return statement consistency. **s** Storage capacity checks. **u** Proper usage of variables and functions. **A** Deactivate all warnings. **C** Constants occurring in conditionals. **D** External declarations are never used. **O** Obsolescent features. **P** Function prototype presence. **R** Detection of unreachable code.
-x	Suppresses messages about variables that have external declarations but are never used.
-MA	Enforces the ANSI C language standard rules. The default mode is equal to the extended C mode. The ANSI mode prepends the standard ANSI library function prototypes in place of the default extended mode C lint library. The ANSI mode enforces a stricter inter-file object reference and provides definition linkage checks.
-Nd*Number*	Changes the dimension table size to *Number*. The default value of *Number* is 2000.
-Nl*Number*	Changes the number of type nodes to *Number*. The default value of *Number* is 8000.

Item	Description
-Nn*Number*	Increases the size of the symbol table to *Number*. The default value of *Number* is 1500.
-Nt*Number*	Changes the number of tree nodes to *Number*. The default value of *Number* is 1000.

In addition, the **lint** command recognizes the following flags of the **cpp** command (macro preprocessor):

Item	Description
-I*Directory*	Adds the *Directory* to the list of directories in which the **lint** command searches for the **#include** files.
-D*Name*[=*Definition*]	Defines the *Name*, as if by the **#define** file. The default of the *Definition* is the value of 1.
-q*DBCS*	Sets multibyte mode specified by the current locale.
-U*Name*	Removes any initial definition of the *Name*, where the *Name* is a reserved symbol that is predefined by the particular preprocessor.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To check a C program for errors, enter:

   ```
   lint command.c
   ```

2. To suppress some of the messages, enter:

   ```
   lint -v -x program.c
   ```

   This checks `program.c`, but does not display error messages about unused function parameters (**-v**) or unused externals (**-x**).

3. To check the program against an additional lint library, enter:

   ```
   lint -lsubs program.c
   ```

   This checks `program.c` against both the **/usr/ccs/lib/llib-lc.ln** standard lint library and **/usr/lib/llib-lsubs.ln** lint library.

4. To check against the portable library and an additional library, enter:

   ```
   lint -lsubs -p program.c
   ```

   This checks `program.c` against both the **/usr/ccs/lib/llib-port.ln** portable lint library and **/usr/lib/llib-lsubs.ln** lint library.

5. To check against a nonstandard library only, enter:

   ```
   lint -lsubs -n program.c
   ```

   This checks `program.c` against only **/usr/lib/llib-lsubs.ln**.

Files

Item	Description
/usr/lib/lint[12}	Programs
/usr/ccs/lib/llib-lansi	Declarations for standard ANSI functions (source)
/usr/ccs/lib/llib-lansi.ln	Declarations for standard ANSI functions (binary format)
/usr/ccs/lib/llib-lc	Declarations for standard functions (source)
/usr/ccs/lib/llib-lc.ln	Declarations for standard functions (binary format)
/usr/ccs/lib/llib-lcrses	Declarations for curses functions (source)
/usr/ccs/lib/llib-lcrses.ln	Declarations for curses functions (binary format)
/usr/ccs/lib/llib-lm	Declarations for standard math functions (source)
/usr/ccs/lib/llib-lm.ln	Declarations for standard math functions (binary format)
/usr/ccs/lib/llib-port	Declarations for portable functions (source)
/usr/ccs/lib/llib-port.ln	Declarations for portable functions (binary format)
/usr/lpp//xlC/lib	Directory containing C++ libraries
/var/tmp/*lint*	Temporary files

listdgrp Command

Purpose

Displays devices of a device class.

Syntax

listdgrp *DeviceClass*

Description

Lists information about devices where the *DeviceClass* parameter referes to a object class of Customized Devices in the Device Configuration database.

Parameters

Item	Description
DeviceClass	Specifies the device class whose members will be displayed.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Examples

1. To list the devices in the **adapter** class, enter:

```
listdgrp adapter
```

The output looks similar to the following:

```
a0
sa1
siokma0
fda0
scsi0
scsi1
bl0
sioka0
siota0
```

Files

Item	Description
/usr/bin/listdgrp	Contains the System V **listdgrp** command.

listvgbackup Command

Purpose

Lists or restores the contents of a volume group backup on a specified media.

Syntax

listvgbackup [**-b** *blocks*] [**-f** *device*] [**-a**] [**-c**] [**-l**] [**-n**] [**-r**] [**-s**] [**-d** *path*] [**-B**] [**-D**] [**-L**] [**-V**] [*file_list*]

Description

The **listvgbackup** command lists the contents of a volume group backup from tape, file, CD-ROM, or other source and can be used to restore files from a valid backup source. The **listvgbackup** command also works for multi-volume backups such as multiple CDs, DVDs, USB disks, or tapes.

The **listvgbackup -r** and **restorevgfiles** commands perform identical operations and should be considered interchangeable.

Flags

Item	Description
-a	Verifies the physical block size of the tape backup, as specified by the **-b** *block* flag. You may need to alter the block size if necessary to read the backup. The **-a** flag is valid only when a tape backup is used.
-b *blocks*	Specifies the number of 512-byte blocks to read in a single input operation, as defined by the *blocks* parameter. If the *blocks* parameter is not specified, the number of blocks read will default to 100.
-B	Prints the volume group backup log to **stdout**. This flag will display the past 256 backups (roughly). The log is in alog format and is kept in **/var/adm/ras/vgbackuplog**. Each line of the log is a semicolon-separated list of the file or device name, the command used to make backup, date, shrink size, full size of the backup, and recommended maintenance or technology level (if any). **Note:** The shrink size is the size of the data on all filesystems. The full size is total size of each filesystem (unused + data).
-c	Produces colon-separated output. This flag only works with the **-l** and **-L** flags.

Item	Description
-d *path*	Specifies the directory path to which the files will be restored, as defined by the *path* parameter. If the **-d** parameter is not used, the current working directory is used. This can be a problem if the current working directory is root. We recommend writing to a temporary folder instead of to root.
-D	Produces debug output.
-l	Displays useful information about a volume group backup. This flag requires the **-f** *device* flag. This flag causes **listvgbackup** to display information such as volume group, date and time backup was made, uname output from backed up system, oslevel, recommended maintenance or technology level, backup size in megabytes, and backup shrink size in megabytes. The shrink size is the size of the data on all filesystems. The full size is the total size of each filesystem (unused + data). The **-l** flag also displays the logical volume and filesystem information of the backed up volume group, equivalent to running "**lsvg -l** *vgname*".
-L	Displays lpp fileset information about a **mksysb** backup only. This flag requires the **-f** *device* flag and displays the equivalent information to that produced by invoking "**lslpp -l**" on the running backed up system. This flag does not produce output about any volume group backup other than that produced by **mksysb**.
-f *device*	Specifies the type of device containing the backup (file, tape, CD-ROM, or other source) as defined by the *device* parameter. When **-f** is not specified, *device* will default to **/dev/rmt0**.
-n	Does not restore ACLs, PCLs, or extended attributes.
-r	Specifies to restore the backup files, as defined by the *file-list* parameter. If the *file-list* parameter is not specified, then all files in the backup will be restored. If the **-r** flag is not used, then executing the **listvgbackup** command only lists the files in the specified backup.
-s	Specifies that the backup source is a user volume group and not rootvg.
-V	Verifies a tape backup. This flag requires the **-f** *device* flag and works for tape devices only. The **-V** flag causes **listvgbackup** to verify the readability of the header of each file on the volume group backup and print any errors that occur to **stderr**.

Parameters

Item	Description
file_list	Identifies the list of files to be restored. This parameter is used only when the **-r** flag is specified. The full path of the files relative to the current directory should be specified in the space-separated list. All files in the specified directory will be restored unless otherwise directed. If you are restoring all files in a directory, we recommend writing to a temporary folder instead of to root.

Examples

1. To list the contents of the system backup located on the default device **/dev/rmt0**, enter:

```
listvgbackup
```

2. To list the contents of the system backup located on device **/dev/cd1**, enter:

```
listvgbackup -f /dev/cd1
```

3. To list the contents of the system backup located on device **/dev/cd1**, which is a user volume group that is not rootvg, enter:

```
listvgbackup -f /dev/cd1 -s
```

4. To restore **/etc/filesystems** from the system backup located on device **/dev/cd1**, enter:

```
listvgbackup -f /dev/cd1 -r ./etc/filesystems
```

5. To restore all files in the **/myfs/test** directory of the non-rootvg backup, which is located on device **/dev/cd1**, and write the restored files to **/data/myfiles**, enter:

```
listvgbackup -f /dev/cd1 -r -s -d /data/myfiles ./myfs/test
```

6. To display colon separated lpp information about a **mksysb** backup tape located on **/dev/rmt0**, enter the following:

```
lsmksysb -Lc -f /dev/rmt0
```

7. To display the volume group backup log to **stdout**, enter:

```
lssavevg -B
```

8. To list volume group and general backup data about a backup located at **/tmp/mybackup**, enter:

```
listvgbackup -l -f /tmp/mybackup
```

9. To verify the readability of each header on a volume group backup tape in **/dev/rmt0**, enter:

```
lsmksysb -V -f /dev/rmt0
```

10. To list the contents of the system backup located on device **/dev/usbms0,** use the following command:

```
listvgbackup –f /dev/usbms0
```

Files

Item	Description
/usr/bin/listvgbackup	Contains the **listvgbackup** command

listX11input Command

Purpose

Lists X11 input extension records entered into the Object Data Manager (ODM) database.

Syntax

listX11input

Description

The **listX11input** command lists all X11 input extension records entered in the ODM database.

Error Codes

Item	Description
ODM could not open class	The ODM database is not stored in the **/usr/lib/objrepos** directory.

livedumpstart Command

Purpose

Initiates a live dump.

Syntax

`livedumpstart` [`-e`] [`-h`] [`-p` *pseudo-component*] [`-q`] [`-r`] [`-u`] [`-c` *component_path*] [`-l` *logical_alias*] [`-t` *type*] [`-C` *component_path* | `-L` *logical_alias* | `-T` *type*] *attribute* [...]

Description

The `livedumpstart` command is used to start a live dump. The dump can include one or more components. Only serialized dumps are used. It can be limited to one pass. The data acquired is dumped to the file system, and the dump is placed in a directory. The dump can be designated as informational or critical.

Components are dumped in the order that you specify. Specify the failing component with either the `-C`, `-L`, or `-T` flag. You cannot specify the name of a pseudo-component.

The data is dumped at the detail level that you set for that component, see the `dumpctrl` command for more information about managing system and live dumps.

If you do not specify the `-q` flag, the `livedumpstart` command displays a message containing the name of the dump.

Flags

Item	Description
`-c` *[+] component_path[+] [:parameter_list]*	Specifies a component by component path name. You can specify the `-c` flag more than once. If you precede a component name with a plus sign (+), the data from that component and its ancestors are dumped. If you follow a component name with a plus sign (+), the data from that component and its descendents are dumped. You can pass parameters to the component. Follow the component name and the optional "+" *:parameter_list*. A *parameter_list* consists of parameters separated by commas. It can also be groups separated by blanks. If a component and its ancestors or descendents are specified, parameters are passed only to the component, not to the ancestors or descendents.
`-C` *[+] component_path[+] [:parameter_list]*	Specifies a failing component by component path name. At most one failing component can be specified. Thus, only one of the `-C`, `-L`, and `-T` flags is allowed, and that component specification must refer to a single component. If **-C basecomp+** is specified, and **basecomp** is not live dump aware, then only one component among **basecomp** and its descendents can be live dump aware. However, if **basecomp** is live dump aware, **basecomp** is the failing component, and it might have multiple live-dump-aware descendents. **Tip:** These rules also apply to a component and its ancestors. If a component is preceded with a plus sign, "+", then that component and its ancestors are dumped. If a component is followed with a plus sign, "+", then that component and its descendents are dumped. If parameters are passed to the component, the component and the optional "+" are followed with *:parameter_list*. A *parameter_list* consists of parameters separated by commas, or keyword=parm_list pairs separated by blanks. See the section on specifying parameters from the command line for more information. Note that if a component and its ancestors and/or descendents are specified, parameters are passed only to the component, not to the ancestors or descendents.
`-e`	Displays an estimate for the size of the dump, which contains the specified components or pseudo-components. This flag obtains a size estimate for the dump without starting the dump. To get an accurate estimate, use the same components, parameters, and detail level that you intend to use for the dump. The estimate takes into account the compression factor.
`-h`	Shows help text. If the `-h` flag is specified with other components or pseudo-components, the help text for those components is shown.
`-l` *[+] logical_alias[+] [:parameter_list]*	Specifies a component by component logical alias. You can specify multiple `-c`, `-l`, and `-t` flags. If a logical alias is preceded with a plus sign "+", then that alias and its ancestors are dumped. If a logical alias is followed with a plus sign "+", then that alias and its descendents are dumped. If parameters are passed to the component, the component and the optional "+" are followed with *:parameter_list*. A *parameter_list* consists of parameters separated by commas, or keyword=parm_list pairs separated by blanks. See the section on specifying parameters from the command line for more information. Note that if a component and its ancestors, descendents, or both are specified, parameters are passed only to the component, not to the ancestors or descendents.

Item	Description
`-L` *[+] logical_alias[+] [:parameter_list]*	Specifies a failing component by component logical alias. At most one failing component can be specified. Thus, only one of the `-C`, `-L`, and `-T` flags is allowed, and that component specification must refer to a single component. If **-L basecomp+** is specified, and **basecomp** is not live dump aware, then only one component among **basecomp** and its descendents can be live dump aware. However, if **basecomp** is live dump aware, **basecomp** is the failing component, and it might have multiple live-dump-aware descendents. **Tip:** These rules also apply to a component and its ancestors. If a logical alias is preceded with a plus sign "+", then that alias and its ancestors are dumped. If a logical alias is followed with a plus sign "+", then that alias and its descendents are dumped. If parameters are passed to the component, the component and the optional "+" are followed with *:parameter_list*. A *parameter_list* consists of parameters separated by commas, or keyword=parm_list pairs separated by blanks. See the section on specifying parameters from the command line for more information. Note that if a component and its ancestors and/or descendents are specified, parameters are passed only to the component, not to the ancestors or descendents.
`-p` *pseudo-component [:parameter_list]*	Specifies a pseudo-component. **Note:** A pseudo-component (`-p`) cannot be a failing component. If parameters are to be passed to the pseudo-component, the pseudo-component must be followed by a *:parameter_list*. A *parameter_list* consists of parameters separated by commas, or keyword=parm_list pairs separated by blanks. See the section on specifying parameters from the command line for more information. The following table is the description of pseudo-components.

Specification	Parameters	Description
eaddr:hex,hex	address and length, hexadecimal values	Dumps memory by kernel effective address.
context:addr=hex-eaddr | tid_t=hex-tid_t | cpu=dec-lcpu | bid=dec-bid	hex-eaddr - context (MST) effective address, hex-tid_t - kernel thread id, dec-lcpu - logical cpu, dec-bid - cpu bind id	Dumps a kernel context. This includes lightweight memory trace data, stack and thread state information.
tid_t:hex-tid	hexadecimal kernel thread id	Dumps a kernel thread by kernel thread ID.
tid:dec-tid	decimal kernel thread id	Dumps a kernel thread by kernel thread ID, and the ID is decimal.
tslot:dec-slot	decimal kernel thread slot number	Dumps a kernel thread by kernel thread ID, and the thread is specified by decimal slot number.
pid_t:hex-pid	hexadecimal kernel process id	Dumps a process by process ID.
pid:dec-pid	decimal process id	Dumps a process by process ID, and the ID is decimal.
pslot:dec-slot	decimal process slot number	Dumps a process by process ID, and the process is specified by decimal slot number.
errbuf	no parameters	Dumps kernel error logging data.
mtrc:common-size, rare-size	common and rare decimal buffer sizes	Dumps lightweight memory trace data.
systrace:dec-size	decimal buffer size	Dump system trace data. If the buffer size is 0, the entire buffer is dumped.
comptrace:component, dec-length	component name and decimal amount of data. The component can be an alias, and the length can be zero to dump the entire buffer.	Dumps component trace data.
kernext:pathname	extension's full path name	Allows symbol resolution for this extension.

`-q`	Specifies quiet mode. No messages are displayed.
`-r`	Dumps data for any subcomponents of the specified components. Specifying this flag is equivalent to specifying every component followed by a "+".
`-t` *[+] type[+] [:parameter_list]*	Specifies a component by its type or subtype. You can specify multiple `-c`, `-l`, and `-t` flags. If a type or subtype is preceded with a plus sign (+), then that component and its ancestors are dumped. If a type or subtype is followed with a plus sign (+), then that component and its descendents are dumped. If parameters are passed to the component, the component and the optional "+" are followed with *:parameter_list*. A *parameter_list* consists of parameters separated by commas, or keyword=parm_list pairs separated by blanks. See the section on specifying parameters from the command line for more information. Note that if a component and its ancestors and/or descendents are specified, parameters are passed only to the component, not to the ancestors or descendents.
`-T` *[+] type[+] [:parameter_list]*	Specifies a failing component by component type or subtype. At most one failing component can be specified. Thus, only one of the `-C`, `-L`, and `-T` flags is allowed, and that component specification must refer to a single component. If **-T type+** is specified, and *type* is not live dump aware, then only one component among *type* and its descendents can be live dump aware. However, if a component of the type *type* is live dump aware, it is the failing component, and it might have multiple live-dump-aware descendents. **Tip:** These rules also apply to a component and its ancestors. If a type or subtype is preceded with a plus sign (+), then that component and its ancestors are dumped. If a type is followed with a plus sign (+), then that component and its descendents are dumped. If parameters are passed to the component, the component and the optional "+" are followed with *:parameter_list*. A *parameter_list* consists of parameters separated by commas, or keyword=parm_list pairs separated by blanks. See the section on specifying parameters from the command line for more information. Note that if a component and its ancestors, descendents, or both are specified, parameters are passed only to the component, not to the ancestors or descendents.
`-u`	Dumps the data for the components that are "above" the specified components in the component hierarchy. This is equivalent to specifying every component preceded by a "+".

You can use wildcard when you specify component names and aliases. Remember that any parameters that you specify are passed to all matching components. You cannot use **all** or an asterisk (*).

Restriction: You can only specify one failing component, so **-C comp*** can resolve to only one component.

Attributes

The dump attributes are specified with keyword=value pairs. They are used to configure dump parameters, construct dump headers and edit symptom information. You change attributes by specifying an *Attribute=Value* parameter. If you have the proper authority you can set the following required attribute:

Item	Description
symptom=*string*	Provides symptom string details that must be supplied to further qualify the dump. The maximum length of this string is 2047 characters.

If you have the proper authority you can set the following optional group attributes:

Item	Description
errcode=*code*	Specifies the error code for the symptom string. If it begins with 0x, the value is in hex; if it begins with 0, the value is octal; otherwise it is decimal.
force=yes|no	If yes, overrides duplicate checking, dumps the data regardless of whether it duplicates a previous dump. The default is **yes**, because any dump taken from the command line should not be treated as a duplicate.
log=yes|no	Specifies whether a log entry should be written when the dump completes. If you specify **yes**, a message is written to the error log. The default is **yes**.
noforce	For software initiated live dumps. Specifies whether to initiate the dump if it duplicates a previous dump that was initiated within the last day. The **noforce** attribute makes the dump subject to duplicate elimination.
nolog	Specifies whether to a message should be written to the error log when the dump is complete. If it is not specified, dump completion and errors are logged.
prefix=*prefix*	Specifies the file name prefix. The name can be no more than 63 characters.
priority=*priority*	Specifies the priority of the dump. You can specify **info** or **critical**. The default is **critical**. If you specify the value **info**, it indicates the dump is for informational purposes, while **critical** indicates the dump is necessary to debug a problem.
title=*string*	Specifies an optional dump title, which can be up to 127 characters.
type=*type*	Specifies whether data should be collected without freezing the system. **serialized|ser** The dump data is gathered when the system is frozen. It might be necessary to use multiple freezes to dump all the data. This is the default. **unserialized|unser** Data is gathered without freezing the system. It might be necessary to use multiple freezes to dump all the data. If you specify **unserialized**, the system is not frozen when gathering the data.
onepass	All data is gathered under one pass. The dump is truncated if all data does not fit in available memory. The default is to use multiple passes if required.

Exit Status

Item	Description
0 (zero)	The `livedumpstart` command completes successfully and produces a message containing the name of the dump.

Item	Description
nonzero	The `livedumpstart` command fails and produces an error message. This command fails under the following conditions: • One or more parameters are not valid. • One or more components are not valid. • None of these components can be specified for a live dump. • A component attempts to take a live dump from within a live dump. • Live dumps are disabled. • A dump already exists. This can occur when you specify the **force=no** attribute. • There is insufficient memory. • All the data cannot be buffered in this single-pass dump. • Too much time is spent while processors are disabled, and this dump is truncated.

Security

Only the root user can run this command.

Examples

1. To dump data for device `ent0`, and components above it in the component hierarchy, enter the following command:

   ```
   livedumpstart -L +ent0 symptom=foo
   ```

   The failing component is `ent0`. This creates a dump named `ent0.yymmddhhmm.00.DZ`. It is a serialized, critical dump.

   Tip: According to the rules for specifying the failing component, if `ent0` is not live dump aware, but multiple ancestors are, then this command fails. If `ent0` is not live dump aware, and only one ancestor is, this ancestor is used as the failing component.

2. To create an informational dump of the process management data for processes 856 and 10272, enter the following command:

   ```
   livedumpstart -p pid:856 -p pid:10272 \
                 info prefix=mydump title="process dump" symptom="foo"
   ```

   The dump is named `mydump.nocomp.yymmddhhmm.00.DZ`. Note there is no failing component.

3. To create a serialized, one-pass dump where `foo` is the failing component, enter the following command:

   ```
   livedumpstart -C foo+:block=45ab8 -pcontext:tid_t=57B29 onepass symptom=bar
   ```

   This command dumps `foo`, its descendents, and the context for kernel thread 57B29. The dump is named `foo.yymmddhhmm.00.DZ`.

4. A subsystem has the parent component with alias `subsyst`. It has only one live-dump-aware component. To create a serialized live dump of this subsystem, you might use the following command:

   ```
   livedumpstart -L subsyst+ title="Dump of subsystem subsyst" symptom=foo
   ```

5. To specify that process 1234 is dumped along with 0x400 bytes starting at 0x45928, enter the following command:

```
livedumpstart -p tid:1234 -p eadder:45928,400 symptom=foo
```

In this example, there is no failing component.

lkdev Command

Purpose

Locks a device. Any attempt to modify device characteristics fails.

Syntax

lkdev [**-l** *Name* **-a** | **-d** [**-c** *Text*]]

lkdev -h

Description

The **lkdev** command locks the specified device (the **-l** *Name* flag). Any attempt to modify device attributes by using the **chdev** or **chpath** command is denied. In addition, an attempt to delete the specified device or one of its paths from the Object Data Manager (ODM) by using either the **rmdev** or **rmpath** command is denied.

Flags

Item	Description
-h	Displays the command usage message.
-l *Name*	Specifies the logical device name of the target device for the paths affected by the change. This flag is required in all cases.
-a	Locks the specified device.
-d	Unlocks the specified device.
-c *Text*	Specifies a text string of up to 64 printable characters that contain no embedded spaces.

Security

Privilege control: Only the root user can execute this command.

Auditing events:

Event	Information
DEV_LOCK	The device command line

Examples

1. To enable the lock for the *hdisk1* disk device, enter the following command:

```
lkdev -l hdisk1 -a
```

2. To disable the lock for the *hdisk1* disk device, enter the following command:

```
lkdev -l hdisk1 –d
```

3. To enable the lock for the *hdisk1* disk device and create a text label, enter the following command:

```
lkdev -l hdisk1 -a –c test_string
```

4. To modify the text label for the *hdisk1* disk device, enter the following command:

```
lkdev -l hdisk1 –c new_test_string
```

Location

Item	Description
/usr/sbin/lkdev	Contains the **lkdev** command

ln Command

Purpose

Links files.

Syntax

To Link a File to a File

ln [**-f** | **-n**] [**-s**] [**-P** | **-L**] *SourceFile* [*TargetFile*]

To Link a File or Files to a Directory

ln [**-f** | **-n**] [**-s**] [**-P** | **-L**] *SourceFile* ... *TargetDirectory*

Description

The **ln** command links the file designated in the *SourceFile* parameter to the file designated by the *TargetFile* parameter or to the same file name in another directory specified by the *TargetDirectory* parameter. By default, the **ln** command creates hard links. To use the **ln** command to create symbolic links, designate the **-s** flag.

A symbolic link is an indirect pointer to a file; its directory entry contains the name of the file to which it is linked. Symbolic links may span file systems and may refer to directories.

If you are linking a file to a new name, you can list only one file. If you are linking to a directory, you can list more than one file.

The *TargetFile* parameter is optional. If you do not designate a target file, the **ln** command creates a new file in your current directory. The new file inherits the name of the file designated in the *SourceFile* parameter. See example 5.

If you specify the **-P** flag and if the source file is a symbolic link, the **ln** command operates similar to the **linkat()** subroutine, where the *SourceFile* parameter is *Path1* argument and the target destination path is the *Path2* argument of the **linkat()** subroutine. Both *DirFileDescriptor1* and *DirFileDescriptor2* parameter values are set as **AT_FDCWD**, and the Flag parameter of the **linkat()** subroutine is set as 0.

If you specify the **-L** flag and if the source file is a symbolic link, the **ln** command operates similar to the **linkat()** subroutine, where the *SourceFile* parameter is *Path1* argument and the target destination path is the *Path2* argument of the **linkat()** subroutine. Both *DirFileDescriptor1* and *DirFileDescriptor2* parameters values are set as **AT_FDCWD**, and the Flag parameter of the **linkat()** subroutine is set as **AT_SYMLINK_FOLLOW**.

If you specify the **-s** flag to create an alias to alter the default behavior when you create hard links (for example, alias ln=' ln -L'), the **-L** and **-P** flags are ignored because the *SourceFile* parameter value acts as a string that is used as the content for the created symbolic link and the source file is not required to exist as a file.

Notes:

1. You cannot link files across file systems without using the **-s** flag.
2. If *TargetDirectory* is already a symbolic link to a directory, then the **ln** command treats the existing target as a file. This means that a command such as **ln -fs somepath/lname symdir** will not follow the existing symbolic link of **symdir**, instead it will create a new symbolic link from **somepath/lname** to **symdir**.

Flags

Item	Description
-f	Causes the **ln** command to replace any destination paths that already exist. If a destination path already exists and the **-f** flag is not specified, the **ln** command writes a diagnostic message to standard error without creating a new link and continues to link the remaining *SourceFiles*.
-L	Creates a hard link to the file that is referenced by the symbolic link for each *SourceFile* parameter that names a file of symbolic link type.
-n	Specifies that if the link is an existing file, do not overwrite the contents of the file. The **-f** flag overrides this flag. This is the default behavior.
-P	Creates a hard link to the symbolic link for each *SourceFile* parameter that names a file of symbolic link type.
-s	Causes the **ln** command to create symbolic links. A symbolic link contains the name of the file to which it is linked. The referenced file is used when an open operation is performed on the link. A **stat** call on a symbolic link returns the linked-to file; an **lstat** call must be done to obtain information about the link. The **readlink** call may be used to read the contents of a symbolic link. Symbolic links can span file systems and refer to directories. **Note:** Absolute path names must be used when specifying the *SourceFile* parameter for the **-s** flag. If the absolute path name is not designated, unexpected results may occur when the *SourceFile* and the *TargetFile* parameters are located in different directories. The source file does not need to exist before creating the symbolic link.

Exit Status

This command returns the following exit values:

Item	Description
0	All specified files were successfully linked.
>0	An error occurred.

Examples

1. To create another link (alias) to a file, enter:

   ```
   ln  -f chap1  intro
   ```

   This links `chap1` to the new name, `intro`. If `intro` does not already exist, the file name is created. If `intro` does exist, the file is replaced by a link to `chap1`. Then both the `chap1` and `intro` file names will refer to the same file. Any changes made to one also appear in the other. If one file name is deleted with the **rm** command, the file is not completely deleted since it remains under the other name.
2. To link a file to the same name in another directory, enter:

```
ln  index  manual
```

This links index to the new name, manual/index.

Note: intro in example 1 is the name of a file; manual in example 2 is a directory that already exists.

3. To link several files to names in another directory, enter:

```
ln  chap2  jim/chap3  /home/manual
```

This links chap2 to the new name /home/manual/chap2 and jim/chap3 to /home/manual/chap3.

4. To use the **ln** command with pattern-matching characters, enter:

```
ln  manual/* .
```

This links all files in the manual directory into the current directory, . (dot), giving them the same names they have in the manual directory.

Note: You must type a space between the asterisk and the period.

5. To create a symbolic link, enter:

```
ln -s /tmp/toc toc
```

This creates the symbolic link, toc, in the current directory. The toc file points to the /tmp/toc file. If the /tmp/toc file exists, the **cat** toc command lists its contents.

To achieve identical results without designating the *TargetFile* parameter, enter:

```
ln -s /tmp/toc
```

6. To create a hard link to the symbolic link of the toc file, enter:

```
ln -P toc target
```

7. To create a hard link to the /tmp/toc file that is referenced by the symbolic link, enter:

```
ln -L toc target
```

Files

Item	Description
/usr/bin/ln	Contains the **ln** command.

locale Command

Purpose

Writes information to standard output about either the current locale or all public locales.

Syntax

locale [**-O 64**][**-a** | **-m**] | [[**-c**] [**-k**] *Name* ...]

Description

The **locale** command writes information to standard output about either the current locale or all public locales. A public locale is a locale available to any application.

To write the name and value of each current locale category, do not specify any flags or variables. To write the names of all available public locales, specify the **-a** flag. To write a list of the names of all available character-mapping (charmap) files, specify the **-m** flag. These charmap filenames are suitable values for the **-f** flag specified with the **localedef** command.

To write information about specified locale categories and keywords in the current locale, specify the *Name* parameter. The *Name* parameter can be one of the following:

- A locale category, such as **LC_CTYPE** or **LC_MESSAGES**
- A keyword, such as **yesexpr** or **decimal_point**
- The **charmap** reserved word to determine the current character mapping

You can specify more than one *Name* parameter with the **locale** command.

If you specify the **locale** command with a locale category name and no flags, the **locale** command writes the values of all keywords in the locale category specified by the *Name* parameter. If you specify the **locale** command with a locale keyword and no flags, the **locale** command writes the value of the keyword specified by the *Name* parameter.

If the *Name* parameter is a locale category name or keyword, the **-c** and **-k** flags can determine the information displayed by the **locale** command.

Flags

Item	Description
-a	Writes the names of all available public locales.
-c	Writes the names of selected locale categories. If the *Name* parameter is a keyword, the **locale** command writes the name of the locale category that contains the specified keyword, and the value of the specified keyword. If the *Name* parameter is a locale category, the **locale** command writes the name of the specified locale category and the values of all keywords in the specified locale category.
-k	Writes the names and values of selected keywords. If the *Name* parameter is a keyword, the **locale** command writes the name and value of the specified keyword. If the *Name* parameter is a locale category, the **locale** command writes the names and values of all keywords in the specified locale category.
-m	Writes the names of all available character-mapping (charmap) files.
-ck	Writes the name of the locale category, followed by the names and values of selected keywords. If the *Name* parameter is a keyword, the **locale** command writes the name of the locale category that contains the specified keyword, and the name and value of the specified keyword. If the *Name* parameter is a locale category, the **locale** command writes the name of the specified locale category and the names and values of all keywords in the specified locale category.
-O 64	Displays locale information as seen by a 64 bit executable. This should be identical to information as seen by a 32 bit executable.

Exit Status

This command returns the following exit values:

Item	Description
0	All the requested information was found and output successfully.
>0	An error occurred.

Examples

1. To retrieve the names and values of all the current locale environment variables, enter:

```
locale
```

If `locale_x` and `locale_y` are valid locales on the system, as determined with `locale -a`, and if the locale environment variables are set as follows:

```
LANG=locale_x
LC_COLLATE=locale_y
```

The **locale** command produces the following output:

```
LANG=locale_x
LC_CTYPE="locale_x"
LC_COLLATE=locale_y
LC_TIME="locale_x"
LC_NUMERIC="locale_x"
LC_MONETARY="locale_x"
LC_MESSAGES="locale_x"
LC_ALL=
```

> **Note:** When setting the locale variables, some values imply values for other locale variables. For example, if the **LC_ALL** locale variable is set to the **En_US** locale, all locale environment variables are set to the **En_US** locale. In addition, implicit values are enclosed in double quotes ("). Explicitly set values are not enclosed in double quotes (").

2. To determine the current character mapping, enter:

```
locale charmap
```

If the **LC_ALL** locale variable is set to the C locale, the **locale** command produces the following output:

```
ISO8859-1
```

3. To retrieve the value of the `decimal_point` delimiter for the current locale, enter:

```
locale  -ck  decimal_point
```

If the **LC_ALL** locale variable is set to the C locale, the **locale** command produces the following output:

```
LC_NUMERIC
decimal_point="."
```

localedef Command

Purpose

Converts locale and character set description (charmap) source files to produce a locale database.

Syntax

localedef [**-c**] [**-f** *Charmap*] [**-i** *SourceFile*] [**-L** *LinkOptions*] [**-m** *MethodFile*] *LocaleName*

Description

The **localedef** command converts source files that contain definitions of locale-dependent information (such as collation, date and time formats, and character properties) into a locale object file used at run-time. The locale object file created by the **localedef** command is then used by commands and subroutines that set the locale with the **setlocale** subroutine.

The **-i** *SourceFile* flag and variable specify the file that contains the source category definitions. If the **-i** flag is not specified, the file is read from standard input.

The **-f** *CharMap* flag and variable specify a file that maps character symbols to actual character encodings. Using the **-f** flag allows one locale source definition to be applicable to more than one code set. If the **-f** flag is not specified, the default value for the *CharMap* variable is ISO8859-1.

The *LocaleName* parameter specifies the locale name for the locale database generated by the **localedef** command from the specified source files. The *LocaleName* parameter can be either an absolute path name for the file location or a relative path name.

If a locale category source definition contains a copy statement and the statement names an existing locale installed in the system, the **localedef** command proceeds as though the source definition contained the valid category source definition for the named locale.

Notes:

1. The **localedef** command uses the C compiler to generate the locale database. Therefore, to use this command you must have the C compiler installed.
2. When replacing systemwide databases, it is advisable to do a soft reboot to ensure that the new locale is used throughout the system.

If an error is detected, no permanent output is created.

If warnings occur, permanent output is created when the **-c** flag is specified. The following conditions cause warning messages to be issued:

- A symbolic name not found in the file pointed to by the *Charmap* variable is used for the descriptions of the **LC_TYPE** or **LC_COLLATE** categories. This is an error condition for other categories.
- The number of operands to the **order_start** keyword exceeds the **COLL_WEIGHTS_MAX** limit.
- Optional keywords not supported by the implementation are present in the source file.

Flags

Item	Description
-c	Forces the creation of locale tables even if warning messages have been issued.
-f *CharMap*	Specifies the name of a file containing a mapping of character symbols and collating element symbols to actual character encodings. A locale is associated with one and only one code set. If this flag is not specified, the ISO 8859-1 code set is assumed. **Note:** The use of certain system-provided *CharMap* files is fully supported. However, while correctly defined user-provided *CharMap* files may work properly, the result of such use is not guaranteed.
-i *SourceFile*	Specifies the path name of a file containing the locale category source definitions. If this flag is not present, source definitions are read from standard input.
-L *LinkOptions*	Passes the specified link options to the **ld** command used to build the locale.
-m *MethodFile*	Specifies the name of a method file that describes the methods to override when constructing a locale. The method file specifies user-supplied subroutines that override existing definitions, as well as a path name for the library containing the specified subroutines. The **localedef** command reads the method file and uses entry points when constructing the locale objects. The code set methods specified are also used in parsing the file pointed to by the *CharMap* variable. **Note:** To create a 64-bit locale, the method file must specify the path of the library as a single archive that has two shared objects, one 32-bit and the other 64-bit, that contain the specified subroutines. Specifying separate paths to the 32-bit and 64-bit shared objects causes the **localedef** command to fail due to incompatible XCOFF format.

Item	Description
LocaleName	Specifies the name of the locale to be created. This is the name that can subsequently be used to access this locale information.

Exit Status

The **localedef** command returns the following exit values:

Item	Description
0	No errors occurred and the locales were successfully created.
1	Warnings occurred and the locales were successfully created.
2	The locale specification exceeded limits or the code set or sets used were not supported by the implementation, and no locale was created.
3	The capability to create new locales is not supported.
>3	Warnings or errors occurred and no locales were created.

Examples

1. To create a locale called `Austin` from standard input and disregard warnings, enter:

```
localedef  -c  Austin
```

2. To create a locale called `Austin` with `Austin.src` as source input, enter:

```
localedef  -i  Austin.src  Austin
```

lock Command

Purpose

Reserves a terminal.

Syntax

lock [**-***Timeout*]

Description

The **lock** command requests a password from the user, reads it, and requests the password a second time to verify it. In the interim, the command locks the terminal and does not relinquish it until the password is received the second time or one of the following occurs:

- The timeout interval is exceeded.
- The command is killed by a user with appropriate permission.

The timeout default value is 15 minutes, but this can be changed with the **-***Timeout* flag.

Flags

Item	Description
-*Timeout*	Indicates the timeout interval in minutes, as specified by the *Timeout* parameter. The default value is 15 minutes.

Examples

1. To reserve a terminal under password control, enter:

   ```
   lock
   ```

   You are prompted for the password twice so the system can verify it. If the password is not repeated within 15 minutes, the command times out.
2. To reserve a terminal under password control, with a timeout interval of 10 minutes, enter:

   ```
   lock -10
   ```

Files

Item	Description
/usr/bin/lock	Contains the **lock** command.

lockd Daemon

Purpose

Processes lock requests.

Syntax

/usr/sbin/rpc.lockd [**-t** *TimeOut*] [**-g** *GracePeriod*] [**-d** *debug*] [**-x** *xnfs*] [**-T** *RetransmissionsTimeout*] [*number of server*]

Description

The **lockd** daemon processes lock requests that are either sent locally by the kernel or remotely by another lock daemon. The **lockd** daemon forwards lock requests for remote data to the server site lock daemon through the RPC package. The **lockd** daemon then asks the **statd** (status monitor) daemon for monitor service. The reply to the lock request is not sent to the kernel until both the **statd** daemon and the server site **lockd** daemon reply. The **statd** daemon should always be started before the **lockd** daemon.

If either the status monitor or the server site lock daemon is unavailable, the reply to a lock request for remote data is delayed until all daemons become available.

When a server recovers, it waits for a grace period for all client site **lockd** daemons to submit reclaim requests. The client site **lockd** daemons, on the other hand, are notified of the server recovery by the **statd** daemon. These daemons promptly resubmit previously granted lock requests. If a **lockd** daemon fails to secure a previously granted lock at the server site, the **lockd** daemon sends a SIGLOST signal to the process.

The **lockd** daemon is started and stopped with the following System Resource Controller (SRC) commands:

```
startsrc -s rpc.lockd
stopsrc -s rpc.lockd
```

To modify the arguments passed to the **lockd** daemon when it is started, use the following command:

```
chssys -s rpc.lockd Parameters...
```

The status monitor maintains information about the location of connections as well as the status in the **/var/statmon/sm** directory, the **/var/statmon/sm.bak** file, and the **/var/statmon/state** file. When restarted, the **statd** daemon queries these files and tries to reestablish the connection it had prior to termination. To restart the **statd** daemon, and subsequently the **lockd** daemon, without prior knowledge of existing locks or status, delete these files before restarting the **statd** daemon.

By default **rpc.lockd** establishes a dynamic socket port number for receiving requests. Entries may be added to the **/etc/services** file specifying the port that **rpc.lock** will listen for requests on. The service name is **lockd** and a unique port number should be specified. The following entries in **/etc/services** file would specify that port 16001 be used for both **tcp** and **udp**.

```
lockd 16001/tcp
lockd 16001/udp
```

Flags

Item	Description
-d *debug*	Specifies the debug level of the **rpc.statd** daemon. By default, the debug level is disabled.
-g *GracePeriod*	Uses the *GracePeriod* variable to specify the amount of time, in seconds, that the **lockd** daemon should wait for reclaim requests for previously granted locks. The default value of the *GracePeriod* variable is 45 seconds.
-T *RetransmissionsTimeout*	Specifies the timeout for one-way RPC connections. One-way RPC connections stay valid for *RetransmissionsTimeout* number of seconds. If this variable is set to 0, then there is no client cache for one-way RPC calls. The default value for the *RetransmissionsTimeout* variable is 300 seconds.
-t *TimeOut*	Uses the *TimeOut* variable to specify the interval between retransmitting lock requests to the remote server. The default value for the *TimeOut* variable is 15 seconds.
-x *xnfs*	Specifies if the **rpc.lockd** daemon needs to follow the xnfs specification. By default, this flag is turned off.

Parameters

Item	Description
number of server	Specifies the number of daemons to start.

Examples

1. To specify a grace period, enter:

   ```
   /usr/sbin/rpc.lockd  -g 60
   ```

   In this example, the grace period is set for 60 seconds.
2. To specify the amount of time the **lockd** daemon should wait before retransmitting a lock request, enter:

   ```
   /usr/sbin/rpc.lockd  -t 30
   ```

   In this example, the retransmissions occur after 30 seconds.

Files

Item	Description
/etc/services	Contains **lockd** parameter information entries.

locktrace Command

Purpose

Controls kernel lock tracing.

Syntax

locktrace [**-r** *ClassName* | **-s** *ClassName* | **-S** | **-R** | **-l**]

Description

The **locktrace** command controls which kernel locks are being traced by the **trace** subsystem. The default is to **trace** none. If the machine has been rebooted after running the `bosboot -L` command, kernel lock tracing can be turned on or off for one or more individual lock classes, or for all lock classes. If `bosboot -L` was not run, lock tracing can only be turned on for all locks or none. The **trace** events collected in this case when locks are taken or missed (hook id 112), and released (hook id 113) do not have the lock class name available.

Flags

Item	Description
-r *classname*	Turns off lock tracing for all kernel locks belonging to the specified class. The option always fails if you did not run the `bosboot -L` command.
-s *classname*	Turns on lock tracing for all kernel locks belonging to the specified class. The option always fails if you did not run the `bosboot -L` command. To trace several specific classes at the same time, run the **locktrace** command multiple times, with a specific lock class each time. You can enter up to 32 class names.
-R	Turns off all lock tracing.
-S	Turns on lock tracing for all locks regardless of their class membership.
-l	Lists kernel lock tracing current status.

Examples

1. To start tracing the SEM_LOCK_CLASS, enter the following command:

```
locktrace -s SEM_LOCK_CLASS
```

2. To stop all lock tracing, enter the following command:

```
locktrace -R
```

3. To reset previous lock trace entries and then trace the SEM_LOCK_CLASS and SHM_LOCK_CLASS lock classes, enter the following commands:

```
locktrace -R
locktrace -s SEM_LOCK_CLASS
locktrace -s SHM_LOCK_CLASS
```

You can view current lock classes using the **-l** flag:

```
locktrace -l
```

The following output will be displayed:

```
lock tracing enabled for classes:
  SHM_LOCK_CLASS
  SEM_LOCK_CLASS
```

File

Item	Description
/usr/bin/locktrace	Contains the **locktrace** command.
/usr/include/sys/lockname.h	Contains the **lock** class names.

logevent Command

Purpose

Logs event information generated by the event response resource manager (ERRM) to a specified log file.

Syntax

logevent [-h] *log_file*

Description

The `logevent` captures event information that is posted by the event response resource manager (ERRM) in environment variables the ERRM generates when an event occurs. This script can be used as an action that is run by an event response resource. It can also be used as a template to create other user-defined actions. The language in which the messages of the `logevent` script are returned depend on the locale settings.

Event information that is returned about the ERRM environment variables includes the following:

Local Time
Time when the event or rearm event is observed. The actual environment variable supplied by ERRM is ERRM_TIME. This value is localized and converted to readable form before being displayed.

This script uses the `alog` command to write event information to and read event information from the specified *log_file*.

Flags

-h
Writes the script's usage statement to standard output.

Parameters

log_file
Specifies the name of the file where event information is logged. An absolute path for the *log_file* parameter should be specified.

The *log_file* is treated as a circular log and has a fixed size of 64KB. When *log_file* is full, new entries are written over the oldest existing entries.

If *log_file* already exists, event information is appended to it. If *log_file* does not exist, it is created so that event information can be written to it.

Exit Status

0
The script has run successfully.

1
A required *log_file* is not specified.

2
The *log_file* path is not valid.

Restrictions

- This script must be run on the node where the ERRM is running.
- The user who runs this script must have write permission for the *log_file* where the event information is logged.

Standard Output

When the `-h` flag is specified, the script's usage statement is written to standard output.

Examples

1. To log information, specify `/tmp/event.log` as follows:

```
/opt/rsct/bin/logevent
/tmp/event.log
```

   The `/tmp/event.log` file does not need to exist when the command is run.

2. To see the contents of the `/tmp/event.log` file, run this command:

```
alog -f /tmp/event.log -o
```

   The following sample output shows a warning event for the `/var` file system (a file system resource):

```
================================================================================
Event reported at Mon Mar 27 16:38:03 2007

Condition Name:                         /var space used
Severity:                               Warning
Event Type:                             Event
Expression:                             PercentTotUsed>90

Resource Name:                          /var
Resource Class Name:                    IBM.FileSystem
Data Type:                              CT_UINT32
Data Value:                             91
```

Location

/opt/rsct/bin/logevent

logform Command

Purpose

Initializes a logical volume for use as a Journaled File System (JFS) log. Initializes an Enhanced Journaled File System (JFS2) outline log. Reformats an inline log for an existing JFS2 file system using an inline log.

Syntax

logform [**-V** *vfstype*] *LogName*

Description

The **logform** command initializes a logical volume for use as a JFS or JFS2 log device. Running the **logform** command on any JFS log device or JFS2 outline or inline log device will destroy all log records on the log device. This may cause the file system to lose its recovery capability and therefore to lose the file system data.

When you run the **logform** command on an outline log for a JFS2 file system that is already using an outline log, the device type for the outline log must be **jfs2log**. Otherwise, the **logform** command will exit with an error.

To reuse an existing logical volume as an outline log device for a JFS2 file system, you must delete the logical volume and then recreate it as device type **jfs2log**.

For the outline log device of a JFS file system, the same rules are applied. That is, for a new logical volume, the type should be `jfslog`. For a reuse logical volume, you should delete the logical volume and recreate it as lv type `jfslog`. However, logform does not do type check for the log device of a JFS file system. The `logform` command does not report any error when input log device has a wrong lv type. The user should pay attention to the lv type.

When you run the **logform** command on a device with logical volume type **jfs2**, if the device has a file system with an inline log, then the inline log will be reformatted. If the device has a file system with an outline log, then an error will be reported.

When the **logform** command is used to format an inline log for an existing JFS2 file system, the file system data will not be affected; only the log records are destroyed. The logical volume type for an inline log is the same as for the file system. For a JFS2 file system, the inline log logical volume type is **jfs2**.

For a JFS2 file system, **logform** formats a maximum of 2047 MBytes of log. If the log size is greater than 2047 MBytes, only 2047 MBytes will be formatted and the rest will left untouched and will not be used.

Flags

Item	Description
-V *vfstype* [`jfs` | `jfs2`]	If specified, *vfstype* indicates what type of file system the log should be formatted for. If this option is not specified, then the type is obtained from the logical volume's type. Note that for jfs2 log device this flag is always ignored. The `logform` command is unable to change the lv type according to the value of the `-V` flag. Therefore the user should create an lv with the correct lv type (`jfslog` or `jfs2log`) before calling the `logform` command. Use of this flag is strongly discouraged.

Parameters

Item	Description
LogName	The *LogName* parameter specifies the absolute path to the logical volume to be initialized (for example, `/dev/jfslog1`). When the **logform** command is run on an inline log, *LogName* is the device name of the file system.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To create a JFS logging device on a newly created volume group, first create a logical volume of type `jfslog`:

```
mklv -t jfslog -y jfslog1 newvg 1
```

This command creates a `jfslog` logical volume named `jfslog1` in the volume group `newvg`. The size of the logical volume is 1 logical partition.

2. To format the `jfslog1` logical volume once it has been created, enter:

```
logform /dev/jfslog1
```

The `jfslog1` logical volume is now ready to be used as a JFS log device.

3. To format the inline log for an existing file system called `/j2` which is on the file system device `/dev/fslv00`, type:

```
logform /dev/fslv00
```

This formats the inline log for file system `/j2`, but does not touch the data in the file system.

Files

Item	Description
/etc/filesystems	Lists the known file systems and defines their characteristics, including the log device.

logger Command

Purpose

Makes entries in the system log.

Syntax

logger [**-f** *File*] [**-i**] [**-r** *[Count]*] [**-p** *Priority*] [**-t** *Tag*] [*Message*]

Description

The **logger** command provides an interface to the **syslog** subroutine, which writes entries to the system log. A *Message* variable can be specified on the command line, which is logged immediately, or a *File* variable is read and each line of the *File* variable is logged. If you specify no flags or variables, the **logger** command will wait for you to enter a message from standard input. The messages returned by the **LOG_KERN** facility cannot be logged by this command.

Flags

Item	Description
-f *File*	Logs the specified *File* variable. If the *Message* variable is specified, this flag is ignored.
-i	Logs the process ID of the logger process with each line.
-p *Priority*	Enters the message with the specified priority. The *Priority* parameter may be a number or a *facility***.***level* priority specifier.
-t *Tag*	Marks every line in the log with the specified *Tag* parameter.
Message	Indicates the message to log. If this variable is not specified, the **logger** command logs either standard input or the file specified with the **-f** *File* flag.

Item	Description
-r *Count*	If the buffer resource is not available, retries logging the message for the specified number of times. If the number is not specified, retries logging the message until the message is logged. The number must be a positive integer ranging from 1 - 1000.

Examples

1. To log a message indicating a system reboot, enter:

```
logger System rebooted
```

2. To log a message contained in the `/tmp/msg1` file, enter:

```
logger -f /tmp/msg1
```

3. To log the daemon facility critical level messages, enter:

```
logger -pdaemon.crit
```

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Files

Item	Description
/usr/bin/logger	Contains the **logger** command.

login Command

Purpose

Initiates a user session.

Syntax

login [**-h** *HostName*] [**-p**] [**-f** *User* | **-k**] [**-e** *Label*] [**-t** *Label*] [*User* [*Environment*]]

Description

The **login** command (part of the **tsm** command) initiates sessions on the system for the user that is specified by the *User* parameter. You can also specify environment variables to be added to the user's environment. These commands are strings of the form *Variable*=*Value*. The **login** command is not normally entered on the command line.

You can configure the **login** command to create your home directory at your login if you do not have a home directory already. The **login** command calls the **mkuser.sys** command to create the home directory and customize the account. To enable this capability, set the **mkhomeatlogin** attribute of the **usw** stanza in the **/etc/security/login.cfg** file to `true`.

Note:

1. The **PATH**, **IFS**, **HOME**, and **SHELL** environment variables can not be initialized from the command line.
2. The **login** command supports multibyte user names. It is recommended that the system administrator must restrict the user names to characters within the portable character set, to avoid any ambiguity.
3. If the **/etc/nologin** file exists, the system prevents the user from logging in and displays the contents of the **/etc/nologin** file. The system does allow the root user to log in if this file exists. The **/etc/nologin** file is removed when you restart the system.
4. If the *domainlessgroups* attribute is set in the /etc/secvars.cfg file, all group IDs are fetched from the LDAP module, and from the files modules, if the user belongs to any one of these domains.

The **login** command can handle Distributed Computing Environment (DCE) user names of up to 1024 characters. DCE user names are stored in the **LOGIN** environment variable. Because DCE user names do not conform to standard operating system requirements, the first 8 characters of the DCE user name are stored in all standard operating system files and environments.

The **login** command performs the following functions:

Item	Description
Checks accounts	The **login** command validates the user's account, ensuring authentication, logins enabled properly, and correct capacity for the port that is used for the login.
Authenticates users	The **login** command verifies the user's identity by using the system defined authentication methods for each user. If a password has expired, the user must supply a new password. If secondary authentication methods are defined, these methods are invoked but need not be successful in logging in to the system.
Establishes credentials	The **login** command establishes the initial credentials for the user from the user database. These credentials define the user's access rights and accountability on the system.
Initiates a session	The **login** command initializes the user environment from the user database, from the command line, and from the **/etc/environment** configuration file; changes the current directory to the user's home directory (normally); and runs the user's initial program.

These functions are performed in the order given; if one fails, the functions that follow are not performed.

When a user logs in successfully, the **login** command makes entries in the **/etc/utmp** file that tracks current user logins and the **/var/adm/wtmp** file that is used for accounting purposes. The **login** command also sets the **LOGIN** and **LOGNAME** environment variables.

Information pertaining to each unsuccessful login is recorded in the **/etc/security/failedlogin** file. The information that is stored is the same as that in the **/etc/utmp** file, except that unrecognizable user names are logged as UNKNOWN_USER. This check ensures that a password accidentally entered as a user name, for example, is not allowed into the system unencrypted.

After a successful login, the **login** command displays the message of the day, the date and time of the last successful and unsuccessful login attempts for this account, and the total number of unsuccessful login attempts for this account since the last successful login. These messages are suppressed if there is a **.hushlogin** file in your home directory.

The **login** command also changes the ownership of the login port to the user. This includes any ports noted as synonyms in the **/etc/security/login.cfg** file.

To preserve the integrity of the system, only one session at a time is allowed to be logged in to a port. This check means that the **login** command entered from the shell prompt cannot succeed, as both the original session and the new login session would be on the same port. However, the **exec login** command

succeeds, because a new shell replaces the current one. The **login** command is typically a built-in shell command, causing the shell to replace itself.

On a Trusted AIX system, you can specify an effective sensitivity label (SL) at login time by specifying the label with the **-e** flag along with the user name. To specify an effective integrity label (TL) during login, specify the label by using the **-t** flag.

If the label has spaces, specify it within quotation marks. The default login SL and TL are defined in the **/etc/security/user** file as user attributes. If no label attribute is specified in the file, the label attributes that are defined in the default stanza are used.

The labels that you supply must be dominated by your clearance and contained in the system accreditation range. You can specify the SL with the **-e** flag and the TL with the **-t** flag at login time. In a labeled network, unless the login is done by using the console, the network's label is assigned to you, regardless of the labels that you specified with the **-e** or **-t** flag.

Your SL clearance must be within the range that is defined for the TTY device in the **/etc/security/login.cfg** file. The effective TL of the user must be the same as the TL of the TTY. After successfully logging in, the clearance is assigned to the login port.

Tip:

Unless your terminal displays only uppercase letters, do not use only uppercase characters for your user name.

To log in with multibyte user names, you must first open a Japanese window (aixterm) and initiate a new login from the Japanese window.

Flags

Item	Description
-e *Label*	Specifies the effective sensitivity label to be used to log in to a Trusted AIX system. **Restriction:** The **-e** flag applies only to systems that are running Trusted AIX.
-f *User*	Identifies a user who has already been authenticated. If the real ID of the login process is root (0), then the user is not authenticated.
-h *HostName*	Identifies the login as a remote login and specifies with the *HostName* variable the name of the system that is requesting the login. This form of the login is used only by the **telnetd** and **rlogind** daemons.
-k	Identifies the login as using Kerberos authentication and causes login to pass control to **/usr/bin/k5dcelogin** to handle authentication. This form of login is only used by the **krshd** daemon.
-p	Preserves the current terminal type by setting it the value of the **$TERM** environment variable instead of the type that is contained in the **CuAt/PdAt** object classes database.
-t *Label*	Specifies the effective integrity label to be used to log in to a Trusted AIX system. **Restriction:** The **-t** flag applies only to systems that are running Trusted AIX.

Security

The **login** command is a PAM-enabled application with a service name of `login`. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the `usw` stanza of **/etc/security/login.cfg**, to `PAM_AUTH` as the root user.

The authentication mechanisms that are used when PAM is enabled depend on the configuration for the login service in **/etc/pam.conf**. The **login** command requires **/etc/pam.conf** entries for the `auth`,

account, password, and session module types. The following is a recommended configuration in **/etc/pam.conf** for the login service:

```
#
# AIX login configuration
#
login auth required /usr/lib/security/pam_aix

login account required /usr/lib/security/pam_aix

login session required /usr/lib/security/pam_aix

login password required /usr/lib/security/pam_aix
```

Examples

1. To log in to the system as user jamesd, enter the following at the login prompt:

   ```
   login: jamesd
   ```

   If a password is defined, the password prompt appears. Enter your password at this prompt.
2. On a Trusted AIX system, to log in to the system as user james, with the effective SL of TOP SECRET, enter the following command:

   ```
   login: james –e “TOP SECRET”
   ```
3. To log in with the effective SL of SECRET, and the effective TL of TOP SECRET, enter the following command:

   ```
   login: james –e “TOP SECRET” –t “TOP SECRET”
   ```
4. On the command line the following can be used:

   ```
   $ login –e “TOP SECRET” james
   ```

Files

Item	Description
/usr/sbin/login	Contains the **login** command.
/etc/utmp	Contains accounting information.
/var/adm/wtmp	Contains accounting information.
/etc/motd	Contains the message of the day.
/etc/passwd	Contains passwords.
$HOME/.hushlogin	Suppresses login messages.
/etc/environment	Contains user environment configuration information.
/etc/security/login.cfg	Contains port synonyms.
/etc/security/lastlog	Contains information that pertains to the most recent successful and unsuccessful login attempts.
/etc/security/failedlogin	Contains information that pertains to each unsuccessful login.
/etc/security/enc/LabelEncodings	Contains label definitions for the Trusted AIX system.

logins Command

Note: Logins command displays system login information details only for the local users or groups which are defined in the **/etc/passwd** and **/etc/group** files.

Purpose

Displays user and system login information.

Syntax

logins [**-a**] [**-m**] [**-o**] [**-p**] [**-s**] [**-t**] [**-u**] [**-x**] [**-g** *Groups*] [**-l** *Logins*]

Description

The **logins** command displays information about user and system logins. By default, the **logins** command prints the following items:

- User ID
- Primary group name
- Primary group ID
- The `/etc/passwd` account field on user information.

The output is sorted by user ID, displaying system logins followed by user logins.

Depending on the options chosen, the following fields can also be displayed:

- user or system login
- user ID number
- multiple group names
- multiple group IDs
- home directory
- login shell
- four password aging parameters
- /etc/passwd account field value (user name or other information)
- primary group name
- primary group ID

Flags

Item	Description
-a	In addition to the default output, the **-a** flag adds two password expiration fields to the display. These fields show how many days a password can remain unused before it automatically becomes inactive and the date that the password will expire.
-g *Groups*	Displays all users belonging to group, sorted by user ID. Multiple groups can be specified as a comma separated list. *Groups* must specify valid group names on the system. Comma separate names when specifying more than one group.
-l *Logins*	Displays the requested login. Multiple logins can be specified as a comma-separated list. *Logins* must specify valid user names on the system.
-m	Displays multiple group membership information.
-o	Formats output into one line of colon separated fields.

Item	Description
-p	Displays users without passwords.
-s	Displays all system logins.
-t	Sorts output by user name instead of by user ID.
-u	Displays all user logins.
-x	Prints an extended set of information about each selected user. Information for each user is printed on a separate line containing the home directory, login shell, and password aging information. The extended information includes the following: • The password status • The date on which the password was last changed • The number of days required between changes • The number of days allowed before a change is needed • The number of days that the user will receive a password expiration warning message before the password expires The password status is displayed in an abbreviated form as PS for logins with password, NP for no password or LK for locked.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Examples

1. To list all the logins with no passwords, enter:

```
logins -p
```

The output looks similar to the following:

```
pwdless      204     staff           1
nopwd        208     staff           1
```

The **-p** option ensures that only logins with no passwords are listed.

2. To list all the system logins sorted by alphabetical order enter:

```
logins -st
```

The output looks similar to the following:

```
adm             4       adm             4
bin             2       bin             2
daemon          1       staff           1
lp              11      lp              11
lpd             9       nobody          -2
root            0       system          0
sys             3       sys             3
uucp            5       uucp            5
```

The **-t** option prints out the logins sorted alphabetically and not by uid.

3. To list the login details of users "root' and "admin", enter:

```
logins -l root,adm
```

The output looks similar to the following:

```
root            0       system          0
adm             4       adm             4
```

4. To list the password aging details of users "root" and "admin" enter:

```
logins -xl root,adm
```

The output looks similar to the following:

```
root            0       system          0
                        /
                        /usr/bin/ksh
                        PS 021102 0 0 0
adm             4       adm             4
                        /var/adm
                        /sbin/sh
                        PS 000000 0 0 0
```

The **-x** option ensures that extended password information for these logins are retrieved and printed in the output.

5. To display the multiple group information of a particular user in a colon separated format enter:

```
logins -mol root,adm
```

The output looks similar to the following:

```
root:0:system:0::bin:2:sys:3:security:7:cron:8:audit:10:lp:11
adm:4:adm:4:
```

The **-m** option is used here to retrieve the multiple group information of a particular login (user). The **-o** option ensures that the output is displayed in colon separated format.

6. To display the users of the "staff" and "sys" groups in a colon separated format, sorted by user name, enter:

```
logins -tsog staff,sys
```

The output looks similar to the following:

```
bin:2:bin:2:
daemon:1:staff:1:
invscout:200:staff:1:
root:0:system:0:
sys:3:sys:3:
```

Files

Item	Description
/usr/bin/logins	Contains the **logins** command.
/etc/passwd	Contains the password file.
/etc/group	Contains the group file.

logname Command

Purpose

Displays login name.

Syntax

logname

Description

The **logname** command displays the login name of the current process. This is the name that the user logged in with and corresponds to the **LOGNAME** variable in the system-state environment. This variable is only set when the user logs into the system.

The **logname** command invokes the **getlogin** subroutine to get the information about the login name.

Security

Access Control: This program is installed as a normal user program in the Trusted Computing Base.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

To display your login name to standard output, enter:

```
logname
```

Files

Item	Description
/usr/bin/logname	Contains the **logname** command.

logout Command

Purpose

Stops all processes on a port.

Syntax

logout

Description

The **logout** command terminates all processes either with the same controlling terminal as the present process or with all processes which have this terminal open. Processes that are not children of the present process are terminated upon access to the terminal. The present process is also terminated. If the **login** command user and the **logout** command user do not match, the **logout** command permission is denied, and the command stops.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

From a shell started by the **ksh** or **bsh** command, enter:

```
logout
```

Files

Item	Description
/usr/bin/logout	Contains the **logout** command.
/etc/utmp	Contains a record of logged-in users.

look Command

Purpose

Finds lines in a sorted file.

Syntax

look [**-d**] [**-f**] *String* [*File ...*]

Description

The **look** command searches sorted files specified by the *File* parameter and prints all lines that begin with the string specified by the *String* parameter. The **look** command uses a binary search, therefore files specified by the *File* parameter must be sorted in the C locale collating sequence.

The **-d** and **-f** flags affect comparisons as in the **sort** command. This means a file must be sorted using the **-f** flag in the **sort** command before using the look command with the **-f** flag.

If the *File* parameter is not specified, the **/usr/share/dict/words** file is assumed with the collating sequence specified by the **-df** flags. The sort is completed using the current collating sequence. This should match the collating sequence used to produce the dictionary file. The **look** command limits the length of a word search to 256 characters.

Flags

Item	Description
-d	Specifies dictionary order. Only letters, digits, tabs, and spaces are considered in comparisons.
-f	Compares uppercase and lowercase letters as equivalent values. Case is not considered in the sorting so that initial-capital and all-capital words are not grouped together at the beginning of the output. **Note:** To use the **look -f** command, the input file must be sorted with the **sort -f** command.

Example

To search for all lines in the `sortfile` file that begin with the letter a, enter:

```
look  a  sortfile
```

File

Item	Description
/usr/share/dict/words	Contains the default dictionary.

lookbib Command

Purpose

Finds references in a bibliography.

Syntax

lookbib [**-n**] [*Database* ...]

Description

The **lookbib** command uses an inverted index made by the **indxbib** command to find sets of bibliographic references. The **lookbib** command reads keywords typed after the > prompt on the terminal, and retrieves records containing all these keywords. If nothing matches, nothing is returned except another > prompt.

The **lookbib** command asks if you need instructions and prints some brief information if you type a user-defined affirmative answer.

The *Database* parameter specifies files that contain bibliographic references, indexes, or similar types of information. It is possible to search multiple databases as long as they have a common index made by the **indxbib** command. In that case, only the first database name given to the **indxbib** command is specified to the **lookbib** command.

If the **lookbib** command does not find the index files (the **.i[abc]** files), it looks for a reference file with the same name as the first database, but without the suffixes. It creates a file with a **.ig** suffix, suitable for use with the **fgrep** command. It then uses this **fgrep** command file to find references. Using the **.ig** file is simpler but slower than using the **.i[abc]** files, and does not allow the use of multiple reference files.

Flags

Item	Description
-n	Turns off the prompt for instructions.

Files

Item	Description
Database.**ia**	Contains the entry file.
Database.**ib**	Contains the posting file.
Database.**ic**	Contains the tag file.
Database.**ig**	Contains the output file.

loopmount Command

Purpose

Associate an image file to a loopback device. Optionally, make an image file available as a file system via the loopback device.

Syntax

loopmount { -i *imagefile* | -l *device* } [-o *mount options* -m *mountpoint*]

Description

This command is similar to **mount** except that it creates a loopback device if not specified, binds the specified file to it, and optionally mounts it. If the command implicitly creates a new loopback device, it sets the temporary attribute in CuAt to **yes** so that it will be deleted by subsequent loopumount or reboot. All the restrictions and features of the **mount** command apply to **loopmount** also.

Flags

Item	Description
-i	Specify an image file name such as an ISO image. This must be specified if –l is not specified.
-l	**ODM name** of a loopback device such as loop0, loop1, etc. This must be specified if –i is not specified.
-o	Options for the mount command.
-m	Mount point such as **/mnt**.

If -l and -i are both specified, the *imagefile* is associated with the device before mounting it.

Security

Privilege Control: Only the root user and members of the system group should have execute (x) access to this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. The following mount an ISO image on /mnt.

   ```
   $ loopmount -i cdrom.iso -o "-V cdrfs -o ro" -m /mnt
   ```

2. The following mounts a disk image on /mydisk with loop2. The image file was bound to loop2 earlier with **chdev** command.

   ```
   $ loopmount -l loop2 -o "-V jfs2 -o rw,log=NULL" -m /mydisk
   ```

   In case the filesystem was created with an INLINE log, this INLINE log can be used.

   ```
   $ loopmount -l loop2 -o "-V jfs2 -o rw,log=INLINE" -m /mydisk
   ```

3. The following mounts an image file bound to loop0 on /mnt.

   ```
   $ loopmount -i mycd.iso -l loop0 -o "-V cdrfs -o ro" -m /mnt
   ```

Files

Item	Description
/usr/sbin/ loopmount	Contains the loopmount command.

loopumount Command

Purpose

Unmounts a previously mounted image file on a loopback device and then removes the device.

Syntax

loopumount { -i *imagefile* | -l *device* } [-o *umount options* -m *mountpoint*]

Description

This command is similar to **umount** except that it unmount the file and deletes the loopback device associated with the mount point if the temporary attribute for the loopback device in CuAt is set to **yes**, and then unmounts it. All the restrictions and features of the **umount** command apply to **loopumount** also.

Flags

Item	Description
-i	Specify an image file name such as an ISO image. This must be specified if –l is not specified.
-l	**ODM name** of a loopback device such as loop0, loop1, etc. This must be specified if –i is not specified.
-o	Options for the **umount** command.
-m	Mount point such as /mnt.

If -i and –l both are specified, -i is ignored.

Security

Privilege Control: Only the root user and members of the system group should have execute (x) access to this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. The following unmount /mnt and deletes the underlying device that was created by a previous invocation of **loopmount**.

```
$ loopumount –i cdrom.iso –o "/mnt"
```

2. The following command unmount /dev/loop2 from /mydisk but does not delete the device as loop2 was created by the user with **mkdev**.

```
$ loopumount -l loop2 -o "/mydisk"
```

Files

Item	Description
/usr/sbin/loopumount	Contains the loopumount command.

lorder Command

Purpose

Finds the best order for member files in an object library.

Syntax

lorder [**-X** {**32**|**64**|**32_64**}] *File ...*

Description

The **lorder** command reads one or more object or library archive files, looking for external references and writing a list of paired file names to standard output. The first pair of files contains references to identifiers that are defined in the second file.

If object files do not end with **.o**, the **lorder** command overlooks them and attributes their global symbols and references to some other file.

Flags

Item	Description
-X *mode*	Specifies the type of object file **lorder** should examine. The *mode* must be one of the following: **32** Processes only 32-bit object files **64** Processes only 64-bit object files **32_64** Processes both 32-bit and 64-bit object files The default is to process 32-bit object files (ignore 64-bit objects). The *mode* can also be set with the **OBJECT_MODE** environment variable. For example, **OBJECT_MODE=64** causes **lorder** to process any 64-bit objects and ignore 32-bit objects. The **-X** flag overrides the **OBJECT_MODE** variable.

Files

Item	Description
/tmp/sym*	Contains temporary files.

lp Command

The **lp** command includes information for the AIX Print Subsystem **lp** and the System V Print Subsystem **lp**.

AIX Print Subsystem lp Command

Purpose

Sends requests to a line printer.

Syntax

lp [**-c**] [**-d***Queue*] [**-m**] [**-n***Number*] [**-o***Option*] [**-s**] [**-t***Title*] [**-w**] [*Files*]

Description

The **lp** command arranges for the files specified by the *Files* parameter and their associated information (called a request) to be printed by a line printer. If you do not specify a value for the *Files* parameter, the **lp** command accepts standard input. The file name **-** (dash) represents standard input and can be specified on the command line in addition to files. The **lp** command sends the requests in the order specified. If the job is submitted to a local print queue, the **lp** command displays the following to standard output:

```
Job number is: nnn
```

where nnn is the assigned job number. To suppress the job number use the **-s** flag.

Flags

Item	Description
-c	Copies the files to be printed immediately when the **lp** command is run. The **lp** command copies files only when requested. No links are created. If you specify the **-c** flag, be careful not to remove any of the files before they are printed. If you do not specify the **-c** flag, changes made to the files after the request is made appear in the printed output.
-d*Queue*	Specifies the print queue to which a job is sent.
-m	Sends mail (see the **mail** command) after the files are printed. By default, no mail is sent upon normal completion of the print request.
-n*Number*	Prints the number of copies of printed output. The default number of copies is 1.
-o*Options*	Specifies that flags specific to the backend be passed to the backend. Thus, for each queue, other flags not described in this article can be included with the **lp** command. See the **piobe** command for a list of these flags. Specifying this flag is the same as specifying the **-o** flag for the **enq** command.
-s	Suppresses the automatic return of job numbers. The **lp** command reports the job number as the default, the **-s** flag overrides the default.
-t*Title*	Specifies printing the title of the file on the banner page of the output.

Item	Description
-w	Writes a message on the print requesters terminal after the files are printed. If the requester is not logged in, the **mail** command sends the message. If the user is logged in on multiple windows or terminals, the message may not be sent to the LFT where the command was issued. The message is sent to the first terminal on which the **writesrv** daemon sees the user to be logged in. **Note:** If the **-w** flag is used in conjunction with the **-m** flag, the print requester will only receive mail and will not get a message on the terminal.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To print the **/etc/motd** file on printer `lp0` attached to device `dlp0`, enter:

   ```
   lp /etc/motd
   ```

2. To print 30 copies of the **/etc/motd** file using a copy of the file, and to notify the user that the job is completed using mail, enter:

   ```
   lp -c -m -n30 -dlp0:lpd0 /etc/motd
   ```

3. To print the **/etc/motd** file using backend flags **-f** and **-a**, with a job title of `blah`, enter:

   ```
   lp -t"blah" -o -f -o -a /etc/motd
   ```

4. To queue the `MyFile` file and return the job number, enter:

   ```
   lp myfile
   ```

5. To queue the `MyFile` file and suppress the job number, enter:

   ```
   lp -s myfile
   ```

Exit Status

This command returns the following exit values:

Item	Description
0	All input files processed successfully.
>0	No output device is available, or an error occurred.

Files

Item	Description
/usr/sbin/qdaemon	Contains the queuing daemon.
/var/spool/lpd/qdir/*	Contains the queue requests.
/var/spool/lpd/stat/*	Contains information on the status of the devices.
/var/spool/qdaemon/*	Contains temporary copies of enqueued files.

Item	Description
/etc/qconfig	Contains the queue configuration file.
/etc/qconfig.bin	Contains digested, binary version of the **/etc/qconfig** file.

System V Print Subsystem lp Command

Purpose (System V)

Sends print requests

Syntax (System V)

lp [*print-options*] [*files*]

lp -i *request-ID print-options*

Description (System V)

The first form of the **lp** command arranges for the named *files* and associated information (collectively called a request) to be printed. If filenames are not specified on the command line, the standard input is assumed. The standard input may be specified along with named *files* on the command line by listing the filenames and specifying **-** for the standard input. The *files* will be printed in the order in which they appear on the command line.

The LP print service associates a unique *request-ID* with each request and displays it on the standard output. This *request-ID* can be used later when canceling or changing a request, or when determining its status. See the **cancel** command for details about canceling a request, and **lpstat** for information about checking the status of a print request.

The second form of **lp** is used to change the options for a request submitted previously. The print request identified by the *request-ID* is changed according to the *print-options* specified with this command. The *print-options* available are the same as those with the first form of the **lp** command. If the request has finished printing, the change is rejected. If the request is already printing, it will be stopped and restarted from the beginning (unless the **-P** flag has been given).

If you enter `lp -?`, the system displays the command usage message and returns 0.

Sending a print request (System V)

The first form of the **lp** command is used to send a print request either to a particular printer or to any printer capable of meeting all requirements of the print request.

Flags must always precede filenames, but may be specified in any order.

Printers for which requests are not being accepted will not be considered when the destination is **any**. (Use the **lpstat -a** command to see which printers are accepting requests.) However, if a request is destined for a class of printers and the class itself is accepting requests, then all printers in the class will be considered, regardless of their acceptance status.

For printers that take mountable print wheels or font cartridges, if you do not specify a particular print wheel or font with the **-S** flag, whichever one happens to be mounted at the time your request is printed will be used. The **lpstat -p** *printer* **-l** command is used to see which print wheels are available on a particular printer. The **lpstat -S -l** command is used to see what print wheels are available and on which printers. Without the **-S** flag, the standard character set is used for printers that have selectable character sets.

If you experience problems with jobs that usually print but on occasion do not print, check the physical connections between the printer and your computer. If you are using an automatic data switch or an A/B switch, try removing it and see if the problem clears.

Flags (System V)

-c
Make copies of the *files* to be printed immediately when **lp** is invoked. Normally *files* will not be copied, but will be linked whenever possible. If the **-c** flag is not specified, the user should be careful not to remove any of the *files* before the request has been printed in its entirety. It should also be noted that if the **-c** flag is not specified, any changes made to the named *files* after the request is made but before it is printed will be reflected in the printed output.

-d *dest*
Choose *dest* as the printer or class of printers that is to do the printing. If *dest* is a printer, then the request will be printed only on that specific printer. If *dest* is a class of printers, then the request will be printed on the first available printer that is a member of the class. If *dest* is **any**, then the request will be printed on any printer that can handle it. Under certain conditions (unavailability of printers, file space limitations, and so on) requests for specific destinations may not be accepted (see **lpstat**). By default, *dest* is taken from the environment variable **LPDEST**. If **LPDEST** is not set, then *dest* is taken from the environment variable **PRINTER**. If **PRINTER** is not set, a default destination (if one exists) for the computer system is used. If no system default is set and **-T** is used, *dest* will be selected on the basis of *content-type* specified with the **-T** flag [see the description of **-T**]. Destination names vary between systems (see **lpstat**).

-f *form-name* [-d any]
Print the request on the form *form-name*. The LP print service ensures that the form is mounted on the printer. If *form-name* is requested with a printer destination that cannot support the form, the request is rejected. If *form-name* has not been defined for the system, or if the user is not allowed to use the form, the request is rejected. (see **lpforms**). When the **-d any** flag is given, the request is printed on any printer that has the requested form mounted and can handle all other needs of the print request.

-H *special-handling*
Print the request according to the value of *special-handling*. Acceptable values for *special-handling* are defined below:

hold
Do not print the request until notified. If printing has already begun, stop it. Other print requests will go ahead of a held request until it is resumed.

resume
Resume a held request. If it had been printing when held, it will be the next request printed, unless subsequently bumped by an **immediate** request. The **-i** flag (followed by a *request-ID*) must be used whenever this argument is specified.

immediate
(Available only to LP administrators) Print the request next. If more than one request is assigned **immediate**, the most recent request will be printed first. If another request is currently printing, it must be put on hold to allow this immediate request to print.

-L *locale-name*
Specify *locale-name* as the locale to use with this print request. By default, *locale-name* is set to the value of **LC_CTYPE**. If **LC_CTYPE** is not set, *locale-name* defaults to the C locale.

-m
Send mail after the files have been printed. By default, mail is not sent upon normal completion of the print request.

-n *number*
Print *number* copies of the output. The default is one copy.

-o *options*
Specify printer-dependent *options*. Several such *options* may be collected by specifying the **-o** keyletter more than once (that is, **-o option[1] -o option[2]** ... **-o option[n]**), or by specifying a list of options with one **-o** keyletter enclosed in double quotes and separated by spaces (that is, **-o "option[1]** *option[2]* **. . .** *option[n]***"**).

nobanner
Do not print a banner page with this request. The administrator can disallow this option at any time.

nofilebreak
Do not insert a form feed between the files given, if submitting a job to print more than one file. This option is not supported by printers configured to use the PS (PostScript) interface.

length=*scaled-decimal-number*
Print this request with pages *scaled-decimal-number* long. A *scaled-decimal-number* is an optionally scaled decimal number that gives a size in lines, characters, inches, or centimeters, as appropriate. The scale is indicated by appending the letter **i** for inches, or the letter **c** for centimeters. For length or width settings, an unscaled number indicates lines or characters; for line pitch or character pitch settings, an unscaled number indicates lines per inch or characters per inch (the same as a number scaled with **i**). For example, **length=66** indicates a page length of 66 lines, **length=11i** indicates a page length of 11 inches, and **length=27.94c** indicates a page length of 27.94 centimeters. This option may not be used with the **-f** option and is not supported by the PS (PostScript).

width=*scaled-decimal-number*
Print this request with pages *scaled-decimal-number* wide. (See the explanation of *scaled-decimal-numbers* in the discussion of **length**, above.) This option may not be used with the **-f** option and is not supported by the PS (PostScript).

lpi=*scaled-decimal-number*
Print this request with the line pitch set to *scaled-decimal-number*. (See the explanation of *scaled-decimal-numbers* in the discussion of **length**, above.) This option may not be used with the **-f** flag and is not supported by the PS (PostScript).

cpi=pica|elite|compressed
Print this request with the character pitch set to **pica** (representing 10 characters per inch), **elite** (representing 12 characters per inch), or **compressed** (representing as many characters per inch as a printer can handle). There is not a standard number of characters per inch for all printers; see the **terminfo** database for the default character pitch for your printer. This option may not be used with the **-f** flag and is not supported by the PS (PostScript).

stty=*stty-option-list*
A list of options valid for the **stty** command; enclose the list with single quotes if it contains blanks.

-P *page-list*
Print the pages specified in *page-list*. This flag can be used only if there is a filter available to handle it; otherwise, the print request will be rejected. The *page-list* may consist of ranges of numbers, single page numbers, or a combination of both. The pages will be printed in ascending order.

-q *priority-level*
Assign this request *priority-level* in the printing queue. The values of *priority-level* range from 0 (highest priority) to 39 (lowest priority). If a priority is not specified, the default for the print service is used, as assigned by the system administrator. A priority limit may be assigned to individual users by the system administrator.

-R
Remove file(s) after submitting the print request. Use this flag with caution.

-r
See **-T** *content-type* [**-r**] below.

-s
Suppress the ` ` request id is ...'' message.

-S *character-set* [-d any]

-S *print-wheel* [-d any]
Print this request using the specified *character-set* or *print-wheel*. If a form was requested and it requires a character set or print wheel other than the one specified with the **-S** flag, the request is rejected.

For printers that take print wheels: if the print wheel specified is not one listed by the administrator as acceptable for the printer specified in this request, the request is rejected unless the print wheel is already mounted on the printer.

For printers that use selectable or programmable character sets: if the *character-set* specified is not one defined in the Terminfo database for the printer (see **terminfo**), or is not an alias defined by the administrator, the request is rejected.

When the **-d any** flag is used, the request is printed on any printer that has the print wheel mounted or any printer that can select the character set, and that can handle all other needs of the request.

-t *title*
Print *title* on the banner page of the output. The default is no title. Enclose *title* in quotes if it contains blanks.

-T *content-type* [-r]
Print the request on a printer that can support the specified *content-type*. If no printer accepts this type directly, a filter will be used to convert the content into an acceptable type. If the **-r** flag is specified, a filter will not be used. If **-r** is specified but no printer accepts the *content-type* directly, the request is rejected. If the *content-type* is not acceptable to any printer, either directly or with a filter, the request is rejected.

In addition to ensuring that no filters will be used, the **-r** flag will force the equivalent of the **-o 'stty=-opost'** flag.

-w
Write a message on the user's terminal after the *files* have been printed. If the user is not logged in, or if the printer resides on a remote system, then mail will be sent instead. Be aware that messages may be sent to a window other than the one in which the command was originally entered.

-y *mode-list*
Print this request according to the printing modes listed in *mode-list*. The allowed values for *mode-list* are locally defined. This option may be used only if there is a filter available to handle it; otherwise, the print request will be rejected.

The following list describes the *mode-list* options:

-y reverse
Reverse the order in which pages are printed. This filter option is not supported by the LP Print Service.

-y landscape
Change the orientation of a physical page from portrait to landscape.

-y x=*number*,y=*number*
Change the default position of a logical page on a physical page by moving the origin.

-y group=*number*
Group multiple logical pages on a single physical page.

-y magnify=*number*
Change the logical size of each page in a document.

-o length=*number*
Select the number of lines in each page of the document.

-P *number*
Select, by page numbers, a subset of a document to be printed.

-n *number*
Print multiple copies of a document.

Security (System V)

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files (System V)

/var/spool/lp/*

lp.cat, lp.set, lp.tell Command

Note: This is a System V Print Subsystem command.

Purpose

Default printer filter used by interface programs.

Syntax

/usr/lib/lp/bin/lp.cat [**-r**] [*Rate*]

/usr/lib/lp/bin/lp.set *H_pitch V_pitch Width Length Charset*

/usr/lib/lp/bin/lp.tell *Printer*

Description

The **lp.cat** command is the default printer filter called by the interface programs. **lp.cat** reads the file to be printed on its standard input and writes it to the device to be printed on.

lp.cat handles the following signals:

- normal termination (**SIGTERM**)
- serial line hangup (**SIGHUP** due to DCD, Data Carrier Detect, dropping)
- interrupts (**SIGINT** and **SIGQUIT**)
- early pipe termination (**SIGPIPE**)

The **lp.cat** command aborts a printing job if it has to wait too long on output. The default maximum time to wait is calculated as twice the output buffer size (2 * 1024 bytes) divided by the smallest of the values of the transmission rate, print rate, or the specified *Rate* (all rates are in characters per second, CPS). A new maximum delay time may be defined by the *Rate* parameter provided that this increases the delay time. If *Rate* is 0 then the delay allowed is effectively infinite.

When the **-r** flag is specified, **lp.cat** reports the effective throughput in characters per second. This statistic is printed on the standard error after the transmission of every 1024 bytes.

Interface programs may call the **lp.set** command to set the character pitch (*H_pitch*), line pitch (*V_pitch*), page *Width*, page *Length*, and character set (*Charset*) on the printer.

The default units are characters per inch for the character and line pitch, number of columns for width, and number of lines for length. Units may be stated explicitly by appending the values with **c** for centimeters, and **i** for inches.

If it cannot set a particular characteristic, **lp.set** exits with a value of 1 and prints a one letter character code followed by a newline character to the standard error. The character codes are shown in the following table:

Character code	Printer characteristic not set
H	character pitch

Character code	Printer characteristic not set
L	page length
S	character set
V	line pitch
W	page width

Interface scripts use **lp.tell** to forward descriptions of printer faults to the print service. **lp.tell** sends everything that it reads on its standard input to the print service. The print service forwards the message as an alert to the print administrator.

Flags

Item	Description
-r	Specifies reports the effective throughput in characters per second for the **lp.cat** command. This statistic is printed on the standard error after the transmission of every 1024 bytes.

Exit Status

The following table shows the possible exit values for **lp.cat**:

Item	Description
0	Normal exit; no error
1	Standard input not defined
2	Standard output not defined
3	Printer type (TERM) not defined or not in terminfo database
4	Standard input and output are identical
5	Write failed; printer may be off-line
6	Excessive delay caused timeout; printer may be off-line
7	Hangup (SIGHUP) detected; loss of carrier
8	Termination (SIGINT or SIGQUIT), or pipe closed early (SIGPIPE)

lp.set returns 0 if successful; otherwise it returns 1 on error.

lp.tell returns:

Item	Description
0	Normal exit; no error
1	Cannot open message queue to the print service
90	Cannot get printer name or key from print service
91	Cannot send message to print service
92	Cannot receive acknowledgment from print service
93	Corrupted acknowledgment received from print service
94	Print service reports message corrupted in transmission

Files

Item	Description
/usr/lib/lp/bin/lp.cat	Full pathname of **lp.cat**
/usr/lib/lp/bin/lp.set	Full pathname of **lp.set**
/usr/lib/lp/bin/lp.tell	Full pathname of **lp.tell**
/etc/lp/model	Printer interface programs directory
/etc/lp/interfaces	Printer interface programs directory

lpacl Information

Purpose

Provides general information about protecting the least-privilege (LP) commands resource class and its resources by using access controls that are provided by the resource monitoring and control (RMC) subsystem.

Description

RMC controls access to all of its resources and resource classes through access control lists (ACLs), using two different ACL implementations. The implementation that RMC uses depends on which class is involved. The two major differences between the implementations are in: 1) the mechanisms with which ACLs are viewed and modified and 2) whether ACLs are associated with individual resources.

RMC implements access controls for its resources and resource classes in the following ways:

1. Through ACLs that are defined by resource class stanzas in the `ctrmc.acls` file.

   You can view these ACLs by examining the `ctrmc.acls` file. You can modify these ACLs using the `chrmcacl` command. Use a stanza to define an ACL that applies to a class or to all of the resources in a class.

   RMC uses this method for all of its resources and resource classes, except for the `IBM.LPCommands` resource class and its resources.
2. Through ACLs that are associated with resources and a resource class within the RMC subsystem.

   You can view and modify these ACLs using LP commands. You can define an ACL that applies to a class or an ACL that applies to an individual resource of a class.

   RMC uses this method for the `IBM.LPCommands` resource class and its resources.

   This section provides information about ACLs that are specific to the `IBM.LPCommands` resource class and its resources.

The LP resource manager uses the `IBM.LPCommands` resource class to define LP resources. These resources represent commands or scripts that require `root` authority to run, but typically the users who need to run these commands do not have `root` authority. By using the LP resource manager commands, users can run commands that require `root` authority. The LP resource manager commands are:

chlpcmd
: Changes the read or write attribute values of an LP resource

lphistory
: Lists or clears a certain number of LP commands that were previously issued during the current RMC session.

lslpcmd
: Lists information about the LP resources on one or more nodes in a domain.

mklpcmd
Defines a new LP resource to RMC and specifies user permissions.

rmlpcmd
Removes one or more LP resources from the RMC subsystem.

runlpcmd
Runs an LP resource.

For descriptions of these commands, see Least-privilege (LP) resource manager commands in *Technical Reference: RSCT for AIX* for AIX and Least-privilege (LP) resource manager commands in *Technical Reference: RSCT for Multiplatforms* for other operating systems. For information about how to use these commands, see the *Administering RSCT* guide.

Because each LP resource can define a unique command, RMC implements ACLs for the `IBM.LPCommands` class that allows access to be controlled at the individual resource level and at the class level. RSCT provides a set of commands that you can use to list and modify the ACLs for the `IBM.LPCommands` class and its resources. The LP ACL commands are:

chlpclacl
Changes the Class ACL

chlpracl
Changes the Resource ACL

chlpriacl
Changes the Resource Initial ACL

chlprsacl
Changes the Resource Shared ACL

lslpclacl
Lists the Class ACL

lslpracl
Lists the Resource ACL

lslpriacl
Lists the Resource Initial ACL

lslprsacl
Lists the Resource Shared ACL

mklpcmd
Defines a new LP resource to RMC and specifies user permissions

Security

- To use the LP commands that change the Class ACL, the Resource Initial ACL, and the Resource Shared ACL, you must have query and administrator permission for the `IBM.LPCommands` class.
- To use the LP command that changes a Resource ACL for an LP resource, you must have query and administrator permission for the LP resource.
- To use the LP commands that list the Class ACL, the Resource Initial ACL, and the Resource Shared ACL, you must have query permission for the `IBM.LPCommands` class.
- To use the LP command that lists a Resource ACL for an LP resource, you must have query permission for the LP resource.

The `Security` section of each LP command description indicates which permissions are required for the command to run properly.

Implementation specifics

This information is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

Location

/opt/rsct/man/lpacl.7

Examples

Some examples of how to modify the LP ACLs follow. In these examples, the commands are run on a management server for a group of nodes in a management domain. The management server is named ms_node and the managed nodes are called mc_node1, mc_node2, and so on. In a management domain, it is most likely that the LP resources are defined on the management server and the LP commands themselves are targeted to the managed nodes. In these examples, the Resource Shared ACL is not used because separate permissions are required for the individual LP resources. These examples assume that the LP resources are not yet defined by using the mklpcmd command.

1. You want to define the lpadmin ID to be the administrator for the LP commands. This ID has the authority to modify the LP ACLs. You also want to give this ID read and write permission to be able to create, delete, and modify the LP resources. To configure this setting, use the root mapped identity to run these commands on the management server:

   ```
   chlpclacl lpadmin@LOCALHOST rwa
   chlpriacl lpadmin@LOCALHOST rwa
   ```

   These commands define the lpadmin ID on the management server as having administrator, read, and write permission for the IBM.LPCommands class and for the Resource Initial ACL. The Resource Initial ACL is used to initialize a Resource ACL when an LP resource is created. Therefore, when an LP resource is created, the lpadmin ID has administrator, read, and write permission to it.

2. The lpadmin ID can now create LP resources that define the LP commands that are needed. Access to the LP resources can be defined using the mklpcmd command or the chlpracl command. When the resource is created, the Resource Initial ACL is copied to the Resource ACL. To modify the Resource ACL using the chlpracl command so that joe is able to use the runlpcmd command for the resource named SysCmd1, the lpadmin ID runs this command on the management server:

   ```
   chlpracl SysCmd1 joe@LOCALHOST x
   ```

   This command gives joe run permission on the management server to the SysCmd1 resource so he can use the runlpcmd command.

3. In this example, only the lpadmin ID has permission to create, delete, and modify LP resources. Use the chlpclacl command so that other users can create and delete LP resources. In this case, they need to have write access to the class. To be able to list the resources in the IBM.LPCommands class, read permission is required. Read permission on a Resource ACL allows a user to view that LP resource. Write permission on a Resource ACL allows a user to modify that LP resource. To allow joe to view the LP resource named SysCmd1, the lpadmin ID runs this command on the management server:

   ```
   chlpracl SysCmd1 joe@LOCALHOST r
   ```

4. There are several nodes in a peer domain. There is an LP resource called SysCmdB1 on nodeB for which joe needs run permission. In addition, joe needs to have run permission from nodes nodeA, nodeB, and nodeD. If you run the chlpracl command on nodeB, you can use joe@LOCALHOST for nodeB, but you need to determine the node IDs for nodeA and nodeD. To obtain the node IDs, enter:

   ```
   lsrpnode -i
   ```

   The following output is displayed:

   ```
   Name    OpState RSCTVersion NodeNum NodeID
   nodeA   Online  3.1.0.0     2       48ce221932ae0062
   nodeB   Online  3.1.0.0     1       7283cb8de374d123
   nodeC   Online  3.1.0.0     4       b3eda8374bc839de
   nodeD   Online  3.1.0.0     5       374bdcbe384ed38a
   nodeE   Online  3.1.0.0     2       ba74503cea374110
   ```

```
nodeF   Online  3.1.0.0     1       4859dfbd44023e13
nodeG   Online  3.1.0.0     4       68463748bcc7e773
```

Then, to give joe the permissions as stated earlier, run on nodeB:

```
chlpracl SysCmd1 -l joe@LOCALHOST joe@0x48ce221932ae0062 \
joe@0x374bdcbe384ed38a x
```

lpadmin Command

Note: This is a System V Print Subsystem command.

Purpose

Configures the LP print service.

Syntax

Adding or Changing the Configuration of a Local Printer

lpadmin -p *Printer* **-v** *Device* [**-D** *Comment*] [**-A** *AlertType*] [**-W** *Minutes*] [**-c** *Class*] [**-e** *Printer1*] [**-F** *FaultRecovery*] [**-f allow:***FormList* | **-f deny***FormList*] [**-h**] [**-I** *Content-Type-List*] [**-i** *Interface*] [**-l**] [**-M -f** *Form-Name* [**-o** *File-break*]] [**-M -S** *Print-Wheel*] [**-m** *Model*] [**-O** *Copy-Options*] [**-o** *Print-Options*] [**-o nobanner** | **-o banner**] [**-r** *Class*] [**-S** *List*] [**-s** *Server-Name* [**!***ServerPrinterName*]] [**-T** *Printer-Type-List*] [**-u allow:***Login-Id-List* | **-u deny:***Login-Id-List*]]

Adding or Changing the Configuration of a Remote Printer

lpadmin -p *Printer* **-s** *ServerName* [**!***ServerPrinterName*] **-v** *Device* [**-D** *Comment*] [**-A** *AlertType*] [**-W** *Minutes*] [**-c** *Class*] [**-e** *Printer1*] [**-F** *FaultRecovery*] [**-f allow:***FormList* | **-f deny***FormList*] [**-h**] [**-I** *Content-Type-List*] [**-i** *Interface*] [**-l**] [**-M -f** *Form-Name* [**-o** *Filebreak*]] [**-M -S** *Print-Wheel*] [**-m** *Model*] [**-O** *CopyOptions*] [**-o** *PrintOptions*] [**-o nobanner** | **-o banner**] [**-r** *Class*] [**-S** *List*] [**-T** *PrinterTypeList*] [**-u allow:***LoginIdList* | **-u deny:***LoginIdList*]] [**-v** *Device*]

Removing a Printer Destination

lpadmin -x *Destination*

Setting or Changing the System Default Destination

lpadmin -d [*Destination*]

Setting an Alert for a Print Wheel

lpadmin -S *Print-Wheel* **-A** *AlertType* [**-W** *Minutes*] [**-Q** *Requests*]

Setting or Changing the Printer's High Sensitivity Labels and Low Sensitivity Labels with Trusted AIX

lpadmin -p *Printer* **-J** *label* **-L** *label*

Description

The **lpadmin** command configures the LP print service by defining printers and devices. It is used to:

- Add and change printers
- Remove printers from the service
- Set or change the system default destination
- Define alerts for printer faults
- Mount print wheels
- Define printers for remote printing services

Printer and class names may be no longer than the maximum length filename allowed for the file system type you are using, and may consist of all printable characters except the space, slash, backslash, colon,

semicolon, comma, asterisk, question mark, and tilde. The dash can be used in any position except the first position in a printer name.

If you enter `lpadmin -?`, the system displays the command usage message and returns 0.

Adding or changing a printer

The **-p** *Printer* flag is used to configure a new printer or to change the configuration of an existing printer. When you use this form of the **lpadmin** command, you must select one of the following:

- **-v** *Device*, required to configure a local printer
- **-s** *ServerNname* [**!***ServerPrinterName*], required to configure a remote printer

Removing a printer destination

The **-x** *dest* flag removes the destination *dest* (a printer or a class), from the LP print service. If *dest* is a printer and is the only member of a class, then the class is deleted. If *dest* is **all**, all printers and classes are removed. No other parameters are allowed with **-x**.

Setting/changing the system default destination

The **-d** [*dest*] flag makes *dest*, an existing printer or class, the new system default destination. If *dest* is not supplied, then there is no system default destination. No other parameters are allowed with **-d**. To unset the system default printer, the user can enter the keyword **none**.

Setting an alert for a print wheel

The **-S** *Print-Wheel* flag is used with the **-A** *Alert-Type* flag to define an alert to mount the print wheel when there are jobs queued for it. If this command is not used to arrange alerting for a print wheel, no alert will be sent for the print wheel. See the other use of **-A** flag, with the **-p**.

The *Alert-Types* are the same as those available with the **-A** flag: **mail**, **write**, **quiet**, **none**, *shell-command*, and **list**. See the description of **-A**, for details about each.

The message sent appears as follows:

```
The print wheel Print-Wheel needs to be mounted
on the printer(s):
printer (integer1 requests)
integer2 print requests await this print wheel.
```

The printers listed are those that the administrator had earlier specified were candidates for this print wheel. The number *integer1* listed next to each printer is the number of requests eligible for the printer. The number *integer2* shown after the printer list is the total number of requests awaiting the print wheel. It will be less than the sum of the other numbers if some requests can be handled by more than one printer.

If the *Print-Wheel* is **all**, the alerting defined in this command applies to all print wheels already defined to have an alert.

If the **-W** flag is not given, the default procedure is that only one message will be sent per need to mount the print wheel. Not specifying the **-W** flag is equivalent to specifying **-W once** or **-W 0**. If *Minutes* is a number greater than zero, an alert is sent at intervals specified by *minutes*.

If the **-Q** flag is also given, the alert is sent when a certain number (specified by the argument *requests*) of print requests that need the print wheel are waiting. If the **-Q** flag is not given, or *requests* is 1 or the word **any** (which are both the default), a message is sent as soon as anyone submits a print request for the print wheel when it is not mounted.

Flags

Item	Description
-A *AlertType* **[-W** *minutes*]	The **-A** flag defines an alert to inform the administrator when a printer fault is detected, and periodically thereafter, until the printer fault is cleared by the administrator. If an alert is not defined for a particular printer, mail is sent to user **lp** by default. The *AlertTypes* are: **mail** Send the alert message using mail (see **mail**) to the administrator. **write** Write the message to the terminal on which the administrator is logged in. If the administrator is logged in on several terminals, one is chosen arbitrarily. **quiet** Do not send messages for the current condition. An administrator can use this option to temporarily stop receiving further messages about a known problem. Once the fault has been cleared and printing resumes, messages will again be sent when another fault occurs with the printer. **none** Do not send messages; any existing alert definition for the printer is removed. No alert is sent when the printer faults until a different alert-type (except **quiet**) is used. ***shell-command*** Run the *shell-command* each time the alert needs to be sent. The shell command should expect the message in standard input. If there are blanks embedded in the command, enclose the command in quotes. The **mail** and **write** values for this option are equivalent to the values **mail** *login-ID* and **write** *login-ID* respectively, where *login-ID* is the current name for the administrator. This will be the login ID of the person submitting this command unless he or she has used the **su** command to change to another login ID. If the **su** command has been used to change the login ID, then the *login-ID* for the new login is used. **list** Display the type of the alert for the printer fault. No change is made to the alert. The message sent appears as follows: `The printer Printer has stopped printing for the reason given` `below. Fix the problem and bring the printer back on line.` `Printing has stopped, but will be restarted in a few minutes;` `issue an enable command if you want to restart sooner.` `Unless someone issues a change request` `lp -i request-id -P . . .` `to change the page list to print, the current request will be` `reprinted from the beginning.` `The reason(s) it stopped (multiple reasons indicate reprinted` `attempts):` `reason` The LP print service can detect printer faults only through an adequate fast filter and only when the standard interface program or a suitable customized interface program is used. Furthermore, the level of recovery after a fault depends on the capabilities of the filter.

Item	Description
	If the *Printer* is **all**, the alerting defined in this command applies to all existing printers. If the **-W** flag is not used to arrange fault alerting for *Printer*, the default procedure is to mail one message to the administrator of *Printer* per fault. This is equivalent to specifying **-W once** or **-W 0**. If *minutes* is a number greater than zero, an alert is sent at intervals specified by *minutes*.
-c *Class*	Insert *Printer* into the specified *Class*. *Class* is created if it does not already exist.
-d [*Dest*]	Makes *dest*, an existing printer or class, the new system default destination.
-D *Comment*	Saves the *Comment* for display whenever a user asks for a full description of *Printer* (see **lpstat**). The LP print service does not interpret this comment.
-e *Printer1*	Copies the interface program of an existing *Printer1* to be the interface program for *Printer*. **Note:** Do not specify the **-i** and **-m** flags may not be specified with the **-e** flag.
-f allow: *FormList* | **-f deny:***FormList*	Allows or denies the forms in *FormLlist* to be printed on *Printer*. By default no forms are allowed on a new printer. For each printer, the LP print service keeps two lists of forms: an *allow-list* of forms that may be used with the printer, and a *deny-list* of forms that may not be used with the printer. With the **-f allow** flag, the forms listed are added to the allow-list and removed from the deny-list. With the **-f deny** flag, the forms listed are added to the deny-list and removed from the allow-list. If the allow-list is not empty, only the forms in the list may be used on the printer, regardless of the contents of the deny-list. If the allow-list is empty, but the deny-list is not, the forms in the deny-list may not be used with the printer. All forms can be excluded from a printer by specifying **-f deny:all**. All forms can be used on a printer (provided the printer can handle all the characteristics of each form) by specifying **-f allow:all**. The LP print service uses this information as a set of guidelines for determining where a form can be mounted. Administrators, however, are not restricted from mounting a form on any printer. If mounting a form on a particular printer is in disagreement with the information in the allow-list or deny-list, the administrator is warned but the mount is accepted. Nonetheless, if a user attempts to issue a print or change request for a form and printer combination that is in disagreement with the information, the request is accepted only if the form is currently mounted on the printer. If the form is later unmounted before the request can print, the request is canceled and the user is notified by mail. If the administrator tries to specify a form as acceptable for use on a printer that doesn't have the capabilities needed by the form, the command is rejected. The **lpadmin** command issues a warning when an invalid (nonexistent) form name is submitted with the **-f deny:** flag. See the other use of **-f**, with the **-M** flag.

Item	Description
-F *FaultRecovery*	Specifies the recovery to be used for any print request that is stopped because of a printer fault, according to the value of *FaultRecovery*: **continue** Continue printing on the top of the page where printing stopped. This requires a filter to wait for the fault to clear before automatically continuing. **beginning** Start printing the request again from the beginning. **wait** Disable printing on *Printer* and wait for the administrator or a user to enable printing again. During the wait the administrator or the user who submitted the stopped print request can issue a change request that specifies where printing should resume. (See the **-i** flag of the **lp** command.) If no change request is made before printing is enabled, printing will resume at the top of the page where stopped, if the filter allows; otherwise, the request will be printed from the beginning. The default value of *FaultRecovery* is **beginning**.
-h	Indicates that the device associated with the printer is hardwired. If neither of the mutually exclusive flags, **-h** and **-l**, is specified, this flag is assumed.
-i *Interface*	Establish a new interface program for *Printer*. The *Interface* is the path name of the new program. Do not specify the **-e** and **-m** flags with this flag.
-I *Content-Type-List*	Allow *Printer* to handle print requests with the content types listed in a *Content-Type-List*. If the list includes names of more than one type, the names must be separated by commas or blank spaces. If they are separated by blank spaces, the entire list must be enclosed in double quotes. The type **simple** is recognized as the default content type. A **simple** type of file is a data stream containing only printable ASCII characters and the following control characters. `Control Character Octal Value Meaning` `backspace 10 move back one character, except at beginning of line` `tab 11 move to next tab stop` `linefeed (newline) 12 move to beginning of next line` `form feed 14 move to beginning of next page` `carriage return 15 move to beginning of current line` To prevent the print service from considering **simple** a valid type for the printer, specify either an explicit value (such as the printer type) in the *content-type-list*, or an empty list. If you do want **simple** included along with other types, you must include **simple** in the *content-type-list*. Except for **simple**, each *content-type* name is freely determined by the administrator. If the printer type is specified by the **-T** option, then the printer type is implicitly considered to be also a valid content type.
-J *label*	Defines the high Sensitivity Label (SL) for a printer with Trusted AIX installed.
-l	Indicates that the device associated with *Printer* is a login terminal. The LP scheduler (**lpsched**) disables all login terminals automatically each time it is started. The **-h** flag may not be specified with this flag.
-L *label*	Defines the low Sensitivity Label (SL) for a printer with Trusted AIX installed.

Item	Description
-M -f *Form-Name* [**-a** [**-o filebreak**]]	Mounts the form *Form-Name* on *Printer*. Print requests that need the pre-printed form *Form-Name* are printed on *Printer*. If more than one printer has the form mounted and the user has specified **any** with the **-d** flag of the **lp** command as the printer destination, then the print request is printed on the one printer that also meets the other needs of the request. The page length and width, and character and line pitches needed by the form are compared with those allowed for the printer, by checking the capabilities in the *terminfo* database for the type of printer. If the form requires attributes that are not available with the printer, the administrator is warned but the mount is accepted. If the form lists a print wheel as mandatory, but the print wheel mounted on the printer is different, the administrator is also warned but the mount is accepted. If the **-a** flag is given, an alignment pattern is printed, preceded by the same initialization of the physical printer that precedes a normal print request. Printing is assumed to start at the top of the first page of the form. After the pattern is printed, the administrator can adjust the mounted form in the printer and press return for another alignment pattern (no initialization this time), and can continue printing as many alignment patterns as desired. The administrator can quit the printing of alignment patterns by typing q. If the **-o filebreak** flag is given, a form feed is inserted between each copy of the alignment pattern. By default, the alignment pattern is assumed to correctly fill a form, so no form feed is added. A form is unmounted either by mounting a new form in its place or by using the **-f none** flag. By default, a new printer has no form mounted. See the other use of **-f** without the **-M**.
-M -S *Print-Wheel*	Mount the *Print-Wheel* on *Printer*. Print requests that need the *Print-Wheel* will be printed on *Printer*. If more than one printer has *Print-Wheel* mounted and the user has specified **any** with the **-d** flag of the **lp** command as the printer destination, then the print request is printed on the one printer that also meets the other needs of the request. If the *Print-Wheel* is not listed as acceptable for the printer, the administrator is warned but the mount is accepted. If the printer does not take print wheels, the command is rejected. A print wheel is unmounted either by mounting a new print wheel in its place or by using the **-S none** flag. By default, a new printer has no print wheel mounted. See the other uses of the **-S** flag without the **-M**.

Item	Description
-m *Model*	Select *Model* interface program, provided with the LP print service, for the printer. DO not use the **-e** and **-i** flags with this flag. The following interface programs are available: **standard** generic printer interface **PS** interface for PostScript printers only By default, the **standard** interface is used.
	-O *Copy-Option* The **-O** controls whether or not **lp** makes a copy of the user's file(s) when a print job is submitted. The *copy-option* can be either **copy** or **nocopy**. If **-O copy** is specified, the LP system always copies the user's source files to the spool area when a print job is submitted. If **-O nocopy** is specified, the files are copied only if the user specifies the **-c** flag of **lp** when submitting the job. This flag sets the value of the **copy-files** parameter in the **/etc/default/lp** file. The value, which can be either **on** or **off**, is checked every time a print job is submitted.
-o *Printing-Option*	Specifies the, in the list below the default given to an interface program if the option is not taken from a preprinted form description or is not explicitly given by the user submitting a request (see **lp**). The only **-o** options that can have defaults defined are listed below. **length**=*scaled-decimal-number* **width**=*scaled-decimal-number* **cpi**=*scaled-decimal-number* **lpi**=*scaled-decimal-number* **stty**='*stty-option-list*' *scaled-decimal-number* refers to a non-negative number used to indicate a unit of size. The type of unit is shown by a trailing letter attached to the number. Three types of scaled decimal numbers can be used with the LP print service: • numbers that show sizes in centimeters, marked with a trailing **c** • numbers that show sizes in inches, marked with a trailing **i** • numbers that show sizes in units appropriate to use, without a trailing letter that is, lines, characters, lines per inch, or characters per inch. The first four default option values must agree with the capabilities of the type of physical printer, as defined in the *terminfo* database for the printer type. If they do not, the command is rejected. The *stty-option-list* is not checked for allowed values, but is passed directly to the **stty** program by the standard interface program. Any error messages produced by **stty** when a request is processed (by the standard interface program) are mailed to the user submitting the request. For each printing option not specified, the defaults for the following attributes are defined in the *terminfo* entry for the specified printer type.

Item	Description
	`` length width cpi lpi ``` The default for **stty** is ``` stty='9600 cs8 -cstopb -parenb ixon -ixany opost -olcuc onlcr -ocrnl -onocr -onlret -ofill nl0 cr0 tab0 bs0 vt0 ff0' ``` You can set any of the **-o** flags to the default values (which vary for different types of printers), by typing them without assigned values, as follows: ``` length= width= cpi= lpi= stty= ```
-o nobanner	Allows a user to submit a print request specifying that no banner page be printed.
-o banner	Forces a banner page to be printed with every print request, even when a user asks for no banner page. This is the default; you must specify **-o nobanner** if you want to allow users to be able to specify **-o nobanner** with the **lp** command.
-p *Printer*	Configures a new printer changes the configuration of an existing printer.
-Q *Requests*	Specifies that an alert be sent when a certain number of print *Requests* that need the print wheel are waiting.
-r *Class*	Remove *Printer* from the specified *Class*. If *Printer* is the last member of *Class*, then *Class* is removed.
-s *Server-Name* [**!***Server-Printer-Name*]	Specifies that you are configuring a remote printer. It makes a server printer accessible to users on your system. *Server-Name* is the name of the system on which the printer is located. It must be listed in the LP systems table. *Server-Printer-Name* is the name used on the server system for that printer. For example, if you want to access *Printer1* on *Server1* and you want it called *Printer2* on your system, enter **-p** *Printer2* **-s** *Server1***!***Printer1*. If *Server-Name* is a Netware server, defined as **-t nuc** using the **lpsystem** command, then *Server-Printer-Name* can be the name of a Netware queue or Netware printer.
-S *List*	Allows either the print wheels or aliases for character sets named in *List* to be used on the printer. The **-S** flag does not let you add items to a *List* specified with an earlier invocation of **-S**; instead, it replaces an existing *List* with a new one. Thus **-S** differs from the **-f**, **-u**, **allow**, and **deny** options, which allow you to modify existing lists of available forms and authorized users. Once you've run the **-S** flag, the print wheels and character sets specified, in *List*, on the current command line are the only ones available. If the printer is a type that takes print wheels, then *List* is a comma or space separated list of print wheel names. Enclose the list with quotes if it contains blanks. These are the only print wheels considered mountable on the printer. You can always force a different print wheel to be mounted, however. Until the flag is used to specify a list, no print wheels is considered mountable on the printer, and print requests that ask for a particular print wheel with this printer is rejected.

Item	Description
	If the printer is a type that has selectable character sets, then *List* is a comma or blank separated list of character set name mappings or aliases. Enclose the list with quotes if it contains blanks. Each mapping is of the form: ``` known-name=alias ``` The *known-name* is a character set number preceded by **cs**, such as cs3 for character set three, or a character set name from the *Terminfo* database entry **csnm**. See **terminfo**. If this flag is not used to specify a list, only the names already known from the Terminfo database or numbers with a prefix of **cs** are acceptable for the printer. If *List* is the word none, any existing print wheel lists or character set aliases is removed. See the other uses of the **-S** with the **-M** flag.
-T *Printer-Type-List*	Identify the printer as being of one or more *Printer-Type*s. Each *Printer-Type* is used to extract data from the **terminfo** database; this information is used to initialize the printer before printing each user's request. Some filters may also use a *Printer-Type* to convert content for the printer. If this flag is not used, the default *Printer-Type* is unknown; no information will be extracted from **terminfo** so each user request is printed without first initializing the printer. Also, this flag must be used if the following are to work: **-o cpi**, **-o lpi**, **-o width**, and **-o length** flags of the **lpadmin** and **lp** commands, and the **-S** and **-f** flags of the **lpadmin** command. If the *Printer-Type-List* contains more than one type, then the *content-type-list* of the **-I** option must either be specified as **simple**, as empty (**-I ""**), or not specified at all.
-u allow:*Login-ID-List*	

Item	Description
-u deny:*Login-ID-List*	Allows or denies the users in *Login-ID-List* access to the printer. By default all users on the local system are allowed on a new printer. The *Login-ID-List* parameter may include any or all of the following constructs: ***login-ID*** a user on the local system ***system-name*!*login-ID*** a user on system *system-name* ***system-name*!all** all users on system *system-name* **all!*login-ID*** a user on all systems **all** all users on the local system **all!all** all users on all systems For each printer the LP print service keeps two lists of users: • An *allow-list* of people allowed to use the printer • A *deny-list* of people denied access to the printer. With the **-u allow** flag, the users listed are added to the allow-list and removed from the deny-list. With the **-u deny** flag, the users listed are added to the deny-list and removed from the allow-list. If the allow-list is not empty, only the users in the list may use the printer, regardless of the contents of the deny-list. If the allow-list is empty, but the deny-list is not, the users in the deny-list may not use the printer. All users can be denied access to the printer by specifying **-u deny:all**. All users may use the printer by specifying **-u allow:all**.
-v *Device*	Specifies you are configuring a local printer. It associates a *Device* with *Printer*. *Device* is the path name of a file that is writable by **lp**. The same *Device* can be associated with more than one printer.
-x *Dest*	Removes the destination *dest* (a printer or a class), from the LP print service.

Notes:

- When creating a new printer, you must specify the **-v**, or **-s** flag. In addition, only one of the following can be supplied: **-e**, **-i**, or **-m**; if none of these three flags is supplied, the model standard is used.
- If you specify the **-s** or **-R** flags, the following flags are not valid: **-A**, **-e**, **-F**, **-h**, **-i**, **-l**, **-M**, **-m**, **-o**, **-v**, and **-W**.
- If you specify the **-J** or **-L** flag, you must specify both flags. The **-p** flag is the only other flag that you can specify with these two flags.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Files

/var/spool/lp/*

/etc/lp

lpar_netboot Command

Purpose

Retrieves the media access control (MAC) address and physical location code from network adapters for a partition, or instructs a partition to start the network.

Syntax

To retrieve the MAC address and physical location code:

`lpar_netboot -M -n [ -v ] [ -x ] [ -f ] [ -i ] [ -E` *environment* `[ -E ... ]] [ -A ] -t ent [ -T` *{ on | off }*`] [ -D -s` *Speed* `-d` *Duplex* `-S` *Server* `-G` *Gateway* `-C` *Client* `[ -K` *subnetmask* `] [ -V` *vlan_tag* `] [ -Y` *vlan_priority* `]] partition_name partition_profile managed_system`

To perform a network boot:

`lpar_netboot [ -v ] [ -x ] [ -f ] [ -i ] [ -E` *environment* `[ -E ... ]] [ -g` *args* `] [{ -A -D | [ -D ] -l` *physical_location_code* `| [ -D ] -m` *MAC-address* `}] -t ent [ -T` *{ on | off }*`] -s` *Speed* `-d` *Duplex* `-S` *Server* `-G` *Gateway* `-C` *Client* `[ -K` *subnetmask* `] [ -V` *vlan_tag* `] [ -Y` *vlan_priority* `] [[ -a -B` *tftp_image_filename* `] -B` *bootp_image_filename* `] partition_name partition_profile managed_system`

To retrieve the MAC address and physical location code on a system supporting a full system partition:

`lpar_netboot -M -n [ -v ] [ -x ] [ -f ] [ -i ] [ -E` *environment* `[ -E ... ]] [ -A ] -t ent [ -T` *{ on | off }*`] [ -D -s` *Speed* `-d` *Duplex* `-S` *Server* `-G` *Gateway* `-C` *Client* `[ -K` *subnetmask* `] [ -V` *vlan_tag* `] [ -Y` *vlan_priority* `]] managed_system managed_system`

To perform network boot on a system supporting a full system partition:

`lpar_netboot [ -v ] [ -x ] [ -f ] [ -i ] [ -E` *environment* `[ -E ... ]] [ -g` *args* `] [{ -A -D | [ -D ] -l` *physical_location_code* `| [ -D ] -m` *MAC-address* `}] -t ent [ -T` *{ on | off }*`] -s` *Speed* `-d` *Duplex* `-S` *Server* `-G` *Gateway* `-C` *Client* `[ -K` *subnetmask* `] [ -V` *vlan_tag* `] [ -Y` *vlan_priority* `] [[ -a -B` *tftp_image_filename* `] -B` *bootp_image_filename* `] managed_system managed_system`

Description

The `lpar_netboot` command instructs a logical partition to network boot, by having the partition send out a `bootp` request to a server specified with the `-S` flag. The server can be a NIM server that serves SPOT resources, or it can be another server that serves network boot images.

If the `-M` and `-n` flags are specified, the `lpar_netboot` command returns the Media Access Control (MAC) address and the physical location code for a particular type of network adapter that is specified with the `-t` flag. When the `-m` flag is specified, `lpar_netboot` boots a partition, by using a specific network adapter that matches the specified MAC address. When the `-l` flag is specified, `lpar_netboot` boots a partition, by using a specific physical location code for the network adapter that matches the specified physical location code. The matching MAC address or physical location code is dependent upon the hardware resource allocation in the profile in which the partition was booted. The `lpar_netboot` command also requires arguments for partition name, partition profile (which contains the allocated hardware resources), and the name of the managed system in which the partition was defined.

Flags

Item	Description
-A	Returns all adapters of the particular type that are specified with the **-t** flag.
-a	Specifies the network IP addresses when the server, client, and gateway are IPv6 addresses.
-B *Image_filename*	Specifies the file name of the network boot image. The **-B** flag is a required flag for IPv6 addresses.
-C *Client*	Specifies the IP address of the partition to start the network.
-D	Performs a ping test to identify and use the adapter that can successfully ping the server that is specified with the -S flag.
-d *Duplex*	Specifies the duplex setting of the partition that is specified with the -C flag. The valid values for the **-d** flag are *full*, *half*, and *auto*.
-E	Specifies the setting for the environment variable. The following commands return the same output: **-E** *LPAR_NETBOOT_DEBUG=1* **export** *LPAR_NETBOOT_DEBUG=1*
-f	Force closes a virtual terminal session for the partition.
-G *Gateway*	Specifies the gateway IP address of the partition that is specified with the -C flag.
-g *args*	Specifies generic arguments for starting the partition. You can specify additional arguments with the firmware boot command, by using the **-g** flag. The **-g** flag is added for starting the preboot execution environment (PXE). An example for the **-g** argument follows: **-g** *autoyast= nfs://9.184.115.219// csminstall/csm/SLES10/09B873DC dhcptimeout=150 install=nfs://9.184.115.219// csminstall/Linux/SLES/10/ppc64/GA/CD1*
-i	Forces an immediate shutdown of the partition. If this option is not specified, a delayed shutdown is performed.
-K *subnetmask*	Specifies the mask that the gateway uses in determining the appropriate subnetwork for routing. The subnet mask is a set of 4 bytes, as in the IP address. The subnet mask consists of high bits (1s) corresponding to the bit positions of the network and subnetwork address, and low bits (0s) corresponding to the bit positions of the host address.

Item	Description
`-l` *phys_loc*	Specifies the physical location code of the network adapter to use for network boot.
`-M`	Displays the network adapter MAC address and physical location code.
`-m` *maddress*	Specifies the MAC address of the network adapter to use for network boot.
`-n`	Instructs the partition to not network boot.
`-S` *Server*	Specifies the IP address of the partition, from which to retrieve the network boot image during network boot.
`-s` *Speed*	Specifies the speed setting of the partition that is specified with the `-C` flag.
`-T`	Enables or disables the display of the firmware-spanning tree. The valid values for the **-d** flag are *on* and *off*.
`-t` *ent*	Specifies the type of adapter for displaying the MAC address or physical location code discovery, or for network boot. The only valid value for the **-t** flag is *ent* for Ethernet.
`-V` *vlan_tag*	Specifies the VLAN tag identifier for tagging Ethernet frames during network installation for virtual network communication. The valid values for the **-V** flag are *0* - *4094*.
`-v`	Displays additional information while the command is running.
`-x`	Displays debug output while the command is running.
`-Y` *vlan_priority*	Specifies the VLAN tag priority for tagging Ethernet frames during network installation for virtual network communication. The valid values for the **-Y** flag are *0* - *7*.

Parameters

Item	Description
partition_name	Specifies the name of the partition.
partition_profile	Specifies the name of the partition profile to use.
managed_system	Specifies the name of the managed system on which the partition is defined.

Environment variables

Item	Description
INSTALLIOS_DEBUG	Prints the lpar_netboot debug output, when specified with the `installios` command.
LPAR_NETBOOT_DEBUG	Prints the lpar_netboot debug output. Hence, it is similar to the **-x** flag.

Item	Description
LPAR_NETBOOT_DEBUG_BOOT	Initiates the firmware boot command, when specified with the **-s** flag.
LPAR_NETBOOT_ADD_TIMEOUT	Extends the timeout value by 5 seconds, as shown in the following example: `LPAR_NETBOOT_ADD_TIMEOUT=5`
LPAR_NETBOOT_SUB_TIMEOUT	Lowers the timeout value by 8 seconds, as shown in the following example: `LPAR_NETBOOT_SUB_TIMEOUT=8`
LPAR_NETBOOT_SPANNING_TREE	Enables or disables the display of the firmware-spanning tree. The valid values for the **-d** flag are *on* and *off*. Hence, it is similar to the **-T** flag
OPEN_DEV_DEBUG	Displays the firmware open_dev debug output, when the value of the *OPEN_DEV_DEBUG* variable is set to *yes*.
FIRMWARE_DUMP	Displays the firmware dump for firmware debugging, when the value of the *FIRMWARE_DUMP* variable is set to *yes*.

Exit Status

Item	Description
`0`	Successful completion

Security

Access Control: You must have root authority to run the `lpar_netboot` command.

Examples

1. To retrieve MAC address and physical location code for partition `machA` with a partition profile `machA_prof` on a managed system `test_sys`, enter the following command:

   ```
   lpar_netboot -M -n -t ent "machA" "machA_prof" "test_sys"
   ```

2. To network boot partition `machA` with a partition profile `machA_prof` on a managed system `test_sys`, enter the following command:

   ```
   lpar_netboot -t ent -s auto -d auto -S 9.3.6.49 -G 9.3.6.1 -C 9.3.6.234
                "machA" "machA_prof" "test_sys"
   ```

3. To network boot partition `machA` with a specific MAC address of `00:09:6b:dd:02:e8` and a partition profile `machA_prof` on a managed system `test_sys`, enter the following command:

   ```
   lpar_netboot -t ent -m 00096bdd02e8 -s auto -d auto -S 9.3.6.49 -G 9.3.6.1
                -C 9.3.6.234 "machA" "machA_prof" "test_sys"
   ```

4. To network boot partition `machA` with a specific physical location code of `U1234.121.A123456-P1-T6` and a partition profile `machA_prof` on a managed system `test_sys`, enter the following command:

   ```
   lpar_netboot -t ent -l U1234.121.A123456-P1-T6 -s auto -d auto -S 9.3.6.49
                -G 9.3.6.1 -C 9.3.6.234 "machA" "machA_prof" "test_sys"
   ```

5. To perform a ping test and a network boot of partition machA with a partition profile machA_prof on a managed system test_sys, enter the following command:

```
lpar_netboot -t ent -D -s auto -d auto -S 9.3.6.49 -G 9.3.6.1 -C 9.3.6.234
             "machA" "machA_prof" "test_sys"
```

6. To perform a ping test and a network boot of partition machA with a partition profile machA_prof on a managed system test_sys and disable firmware-spanning tree discovery, enter the following command:

```
lpar_netboot -t ent -T off -D -s auto -d auto -S 9.3.6.49 -G 9.3.6.1
             -C 9.3.6.234 "machA" "machA_prof" "test_sys"
```

Location

/opt/ibm/sysmgt/dsm/dsmbin/lpar_netboot

lparstat Command

Purpose

Reports logical partition (LPAR) related information and statistics.

Syntax

lparstat {-i [-W | -x | **-s** | **-P** | >|**-N**|<] | -W | **-s** | **-P** | >|**-u**|< | >|**-N**|< | -d | **-m** [**-e** [**r** | **R**] [**-p**[**w**]]] | [-H | -h] | [**-X** [**-o** *filename*]] [-c] | [**-E** [**w**]] = [*Interval* [*Count*]] [**-L**]}

Description

The lparstat command provides a report of LPAR related information and utilization statistics. This command provides a display of current LPAR related parameters and Hypervisor information, as well as utilization statistics for the LPAR. An interval mechanism retrieves numbers of reports at a certain interval.

The various options of lparstat command are exclusive of each other. The lparstat command with no options will generate a single report containing utilization statistics related to the LPAR since boot time. If the -h option is specified, the report will include summary statistics related to the Hypervisor. If an *interval* and *count* are specified, the above report display repeats for every *interval* seconds and for *count* iterations. *interval* and *count* cannot be used with the -i option. Only root users can run the -h and -H flags.

The *interval* parameter specifies the amount of time in seconds between each report. If you do not specify the *interval* parameter, the **lparstat** command generates a single report that contains statistics for the time since system startup and then exits. You can specify the *count* parameter only with the *interval* parameter. If you specify the *count* parameter, its value determines the number of reports that are generated and the number of seconds apart. If you specify the *interval* parameter without the *count* parameter, reports are continuously generated. Do not specify a value of zero to the *count* parameter.

When the lparstat command is invoked without the -i flag, two rows of statistics are displayed. The first row displays the System Configuration, which is displayed once when the command starts and again whenever there is a change in the system configuration. The second row contains the Utilization Statistics which will be displayed in intervals and again any time the values of these statistics are deltas from the previous interval.

If you specify the **-X** option, the lparstat command creates an XML file.

The following information is displayed in the system configuration row:

type
Indicates the partition type. The value can be either dedicated or shared.

mode
Indicates whether the partition processor capacity is capped or uncapped allowing it to consume idle cycles from the shared pool. Dedicated LPAR is capped or donating.

smt
Indicates whether simultaneous multithreading is enabled or disabled in the partition. If there are two SMT threads, the row is displayed as "on." However, if there are more than two SMT threads, the number of SMT threads is displayed.

lcpu
Indicates the number of online logical processors.

mem
Indicates online memory capacity.

Note: If Active Memory Expansion is enabled, **mem** specifies the expanded memory size configured for this LPAR. However, if the environment variable *AME_MEMVIEW* is set to **TRUE**, the **mem** value specifies the true memory size.

psize
Indicates the number of online physical processors in the pool.

ent
Indicates the entitled processing capacity in processor units. This information is displayed only if the partition type is shared.

If you specify the **-m** flag, the following information is displayed in the system configuration row:

lcpu
Indicates the number of online logical processors.

ent
Indicates the entitled processing capacity in processor units.

mem
Indicates online memory capacity.

Note: If Active Memory Expansion is enabled, **mem** specifies the expanded memory size configured for this LPAR. However, if the environment variable *AME_MEMVIEW* is set to **TRUE**, the **mem** value specifies the true memory size.

mpsz
Indicates the memory pool size of the pool that the partition belongs to (in GB).

iome
Indicates the I/O memory entitlement of the partition (in MB).

iomp
Indicates the number of I/O memory entitlement pools in the LPAR.

If you specify the -c flag, the following additional information is displayed in the system configuration row:

mmode
Indicates the system's memory mode. The values for **mmode** are:

Item	Description
Ded	Neither Active Memory Sharing nor Active Memory Expansion is enabled
Shar	Active Memory Sharing is enabled
Ded-E	Active Memory Expansion is enabled
Shar-E	Both Active Memory Sharing and Active Memory Expansion are enabled

mem
Indicates the expanded memory size of the LPAR.

tmem
Indicates the true memory size of the LPAR.

The following information is displayed in the utilization row:

%user
Indicates the percentage of the entitled processing capacity used while executing at the user level (application).

For dedicated partitions, the entitled processing capacity is the number of physical processors.

For uncapped partitions with a current physical processor consumption above their entitled capacity, the percentage becomes relative to the number of physical processor consumed (physc).

%sys
Indicates the percentage of the entitled processing capacity used while executing at the system level (kernel).

For dedicated partitions, the entitled processing capacity is the number of physical processors.

For uncapped partitions with a current physical processor consumption above their entitled capacity, the percentage becomes relative to the number of physical processor consumed (physc).

%idle
Indicates the percentage of the entitled processing capacity unused while the partition was idle and did not have any outstanding disk I/O request.

For dedicated partitions, the entitled processing capacity is the number of physical processors.

For uncapped partitions with a current physical processor consumption above their entitled capacity, the percentage becomes relative to the number of physical processor consumed (physc).

%wait
Indicates the percentage of the entitled processing capacity unused while the partition was idle and had outstanding disk I/O request(s).

For dedicated partitions, the entitled processing capacity is the number of physical processors.

For uncapped partitions with a current physical processor consumption above their entitled capacity, the percentage becomes relative to the number of physical processor consumed (physc).

The following statistics are displayed when the partition type is shared or dedicated-donating:

physc
Indicates the number of physical processors consumed.

vcsw
Indicates the number of virtual context switches that are virtual-processor hardware preemptions.

The following statistics are displayed only when the partition type is shared:

%entc
Indicates the percentage of the entitled capacity consumed. Because the time base over which this data is computed can vary, the entitled capacity percentage can sometimes exceed 100%. This excess is noticeable only with small sampling intervals.

lbusy
Indicates the percentage of logical processor(s) utilization that occurred while executing at the user and system level.

app
Indicates the available physical processors in the shared pool.

phint
Indicates the number of phantom (targeted to another shared partition in this pool) interruptions received.

The following statistics are displayed only when the -h flag is specified:

%hypv
Indicates the percentage of physical processor consumption spent making hypervisor calls.

hcalls
Indicates the average number of hypervisor calls that were started.

The following statistic is displayed only if the hardware can use the SPURR, and the processor is not running at nominal speed:

%nsp
Indicates the current average processor speed as a percentage of nominal speed.

The following statistic is displayed only if the turbo-mode accounting is disabled:

%utcyc
Indicates the total percentage of unaccounted turbo cycles.

The following statistics are displayed only when the **-d** flag is specified.

%utuser
Indicates the percentage of unaccounted turbo cycles in the user mode execution (application).

%utsys
Indicates the percentage of unaccounted turbo cycles in the kernel mode execution (kernel).

%utidle
Indicates the percentage of the unaccounted turbo cycles when the partition is idle and does not have any outstanding disk I/O requests.

%utwait
Indicates the percentage of the unaccounted turbo cycles when the partition is idle and has outstanding disk I/O requests.

If you specify the **-m** flag, the following metrics are displayed:

physb
Indicates that the physical processor is busy.

%entc
Indicates the percentage of the entitled capacity consumed. Because the time base over which this data is computed might vary, the entitled capacity percentage can sometimes exceed 100%. This excess is noticeable only with small sampling intervals.

vcsw
Indicates the number of virtual context switches that are virtual-processor hardware preemptions.

hpi
Indicates the number of hypervisor page-ins occurred.

hpit
Indicates the time that is spent waiting for hypervisor page-ins in milliseconds.

pmem
Indicates the physical memory that is allocated to the LPAR by hypervisor in GB.

iomin
Indicates the minimum entitlement of the I/O memory pool in MB.

iomu
Indicates the I/O memory entitlement of the LPAR in use in MB.

iomf
Indicates the free I/O memory entitlement in MB.

iohwm
Indicates the high water mark of I/O memory entitlement usage in GB.

iomaf
Indicates the total number of times that allocation requests for I/O memory entitlement pools have failed since system startup.

If you specify the **-e** flag with the **-m** flag, the following information about I/O memory entitlement pools is displayed:

iompn
Indicates the name of the I/O memory entitlement pool in MB.

iomin
Indicates the minimum entitlement of the I/O memory pool in MB.

iodes
Indicates the desired entitlement of the I/O memory pool in MB.

ioinu
Indicates the entitlement of the I/O memory pool in use in MB.

iores
Indicates the reserved entitlement of the I/O memory pool in MB.

iohwm
Indicates the high water mark of entitlement usage of the I/O memory pool in MB.

ioafl
Indicates the total number of times that allocation requests for this I/O memory entitlement pool have failed since system startup.

The following statistics are displayed only when the -c flag is specified:

%xcpu
Indicates the percentage of utilization (relative to the overall CPU consumption by the logical partition) for the Active Memory Expansion (AME) activity.

xphysc
Indicates the number of physical processors used for the Active Memory Expansion activity.

dxm
Indicates the size of the expanded memory deficit for the LPAR in MB.

pgcol
Indicates the logical real memory pages of the calling partition in megabytes that are coalesced during the active memory sharing activity.

mpgcol
Indicates the number of megabytes of the memory pages that are called by the memory pool of the coalesced partition during the Active Memory sharing activity. If the partition is not authorized to access the poolwide statistics, the metric shows zero.

ccol
Indicates the fraction of the CPU consumed in coalescing pages during the Active Memory sharing activity. If the partition is not authorized to access the poolwide statistics, the metric shows zero.

Note: Memory page coalescing is a transparent operation wherein the hypervisor detects duplicate pages, directs all the user read pages to a single copy, and reclaims the other duplicate physical memory pages.

Flags

Item	Description
-c	Adds the memory compression statistics of the LPAR to the default **lparstat** output. **Note:** This option is available only when Active Memory Expansion is enabled.
-d	Shows the detailed CPU utilization statistics. When the turbo-mode accounting is disabled, the **lparstat** command shows the breakdown by category of the unaccounted turbo cycles along with the dedicated, donating or shared utilization columns: %user, %sys, %idle, %wait, %entc, %idon, %bdon, %istol and %bstol.
-e	Displays information about the I/O memory entitlement pools of the LPAR. You can specify the **-e** flag only with the **–m** flag. See the metrics that are displayed when you specify the **-m** flag.
-E	Reports Scaled Processor Utilization Resource Register (SPURR) based utilization metrics if run on a SPURR-capable processor.
-h	Adds summary hypervisor statistics to the default `lparstat` output.

Item	Description
`-H`	Provides detailed Hypervisor information. This option basically displays the statistics for each of the Hypervisor calls. The various Hypervisor statistics displayed by this option, for each of the Hypervisor calls, are as below: **Statistic** **Description** **Number of calls** Number of Hypervisor calls made. **Total Time Spent** Percentage of total time spent in this type of call. **Hypervisor Time Spent** Percentage of Hypervisor time spent in this type of call. **Average Call Time** Average call time for this type of call in nano-seconds. **Maximum Call Time** Maximum call time for this type of call in nano-seconds.
`-i`	Lists details on the LPAR configuration. The various details displayed by the `-i` option are listed below: **Name** **Description** **Partition Name** Logical partition name as assigned at the HMC. **Partition Number** Number of this logical partition. **Power Save Mode** Power saving mode of this logical partition. **Online Virtual CPUs** Number of CPUs (virtual engines) currently online. **Maximum Virtual CPUs** Maximum possible number of CPUs (virtual engines). **Online Memory** Amount of memory currently online. **Maximum Memory** Maximum possible amount of Memory. **Type** Indicates whether the LPAR is using dedicated or shared CPU resource and if the SMT is turned ON. The Type is displayed in the format [Shared | Dedicated] [-SMT] [-#] The following list explains the different Type formats: • Shared - Indicates that the LPAR is running in the Shared processor mode. • Dedicated - Indicates that the LPAR is running in the dedicated processor mode. • SMT[-#] - Indicates that the LPAR has SMT mode turned ON and the number of SMT threads is 2. If the number of threads is greater than 2, then the number of threads is also displayed. **Mode** Indicates whether the LPAR processor capacity is capped or uncapped allowing it to consume idle cycles from the shared pool. Dedicated LPAR is capped or donating. **Entitled Capacity** The number of processing units this LPAR is entitled to receive. **Variable Capacity Weight** The priority weight assigned to this LPAR which controls how extra (idle) capacity is allocated to it. A weight of `-1` indicates a soft cap is in place. **Minimum Capacity** The minimum number of processing units this LPAR was defined to ever have. Entitled capacity can be reduced down to this value. **Maximum Capacity** The maximum number of processing units this LPAR was defined to ever have. Entitled capacity can be increased up to this value. **Capacity Increment** The granule at which changes to Entitled Capacity can be made. A value in whole multiples indicates a Dedicated LPAR. **Maximum Physical CPUs in System** The maximum possible number of physical CPUs in the system containing this LPAR.

Item	Description
	(*Details displayed by the* **-i** *flag, are as follows*):
	Active Physical CPUs in System The current number of active physical CPUs in the system containing this LPAR.
	Active CPUs in Pool The maximum number of CPUs available to this LPAR's shared processor pool.
	Shared Physical CPUs in system The number of physical CPUs available for use by shared processor LPARs.
	Maximum Capacity of Pool The maximum number of processing units available to this LPAR's shared processor pool.
	Entitled Capacity of Pool The number of processing units that this LPAR's shared processor pool is entitled to receive.
	Unallocated Capacity The sum of the number of processor units unallocated from shared LPARs in an LPAR group. This sum does not include the processor units unallocated from a dedicated LPAR, which can also belong to the group. The unallocated processor units can be allocated to any dedicated LPAR (if it is greater than or equal to 1.0) or shared LPAR of the group.
	Physical CPU Percentage Fractional representation relative to whole physical CPUs that these LPARs virtual CPUs equate to. This is a function of Entitled Capacity / Online CPUs. Dedicated LPARs would have 100% Physical CPU Percentage. A 4-way virtual with Entitled Capacity of 2 processor units would have a 50% physical CPU Percentage.
	Minimum Memory Minimum memory this LPAR was defined to ever have.
	Minimum Virtual CPUs Minimum number of virtual CPUs this LPAR was defined to ever have.
	Unallocated Weight Number of variable processor capacity weight units currently unallocated within the LPAR group.
	Partition Group ID LPAR group that this LPAR is a member of.
	Shared Pool ID Identifier of Shared Pool of Physical processors that this LPAR is a member.
	(*Details displayed by the* **-i** *flag, are as follows*):
	Memory Mode Indicates whether the memory mode is shared or dedicated. If Active Memory Expansion is enabled, the memory mode also includes a new mode called **Expanded**.
	Total I/O memory entitlement The I/O memory entitlement of the LPAR.
	Variable memory capacity weight The variable memory capacity weight of the LPAR.
	Memory Pool ID The memory pool ID of the pool that the LPAR belongs to.
	Physical Memory in the Pool The physical memory present in the pool that the LPAR belongs to.
	Hypervisor Page Size The page size that hypervisor uses for the page-in and page-out of LPAR logical-memory pages.
	Unallocated Variable Memory Capacity Weight The unallocated variable memory-capacity weight of the LPAR.
	Unallocated I/O memory entitlement The unallocated I/O memory entitlement of the LPAR.
	Memory Group ID of LPAR The memory group ID of the Workload Manager group that the LPAR belongs to.
	Target Memory Expansion Factor The target memory expansion factor configured for the LPAR. **Note:** The target memory expansion factor is displayed when Active Memory Expansion is enabled.
	Target Memory Expansion Size The target expanded memory size for the LPAR. The target expanded memory size is the true memory size multiplied by the target memory expansion factor. **Note:** The target memory expansion size is displayed when Active Memory Expansion is enabled.
	Power Save Mode The power saving mode for the LPAR.
	Subprocessor Mode The subprocessor mode for the LPAR.
	You can specify the **-i** flag alone or with the **-P**, **-W**, **-s**, and **-N** flags.
-L	Displays whether the Live Partition Mobility (LPM) operation can be performed on an LPAR.

Item	Description
-m	Displays the statistics that are related to the following aspects: • The logical memory • The physical memory backing the logical memory of the LPAR • The I/O memory entitlement of the LPAR • The memory pool information on the pool that the LPAR belongs to For more information about the metrics that are displayed when you specify the **-m** flag, see the metrics section.
-N	Displays information about the EnergyScale modes of the system. You can specify only the **-N** flag or you can specify this flag along with the **-i**, **-P**, **-W**, and **-s** flags. **Note:** The details that are listed when you run the **lparstat -N** command might change based on the hardware configuration of the system and new firmware level of the system.
-o	Specifies the file name for the XML output.
-p	Displays the information about the page coalescing statistics of the LPAR. You can specify the **-p** flag only with the **-m** flag. When you run the **lparstat** command with the **-w** and **-p** flags, the result displays all the metrics that are displayed by the **-e** flag in a single line.
-P	Displays information about the energy management tuning parameters. You can specify the **-P** flag alone or with the **-i**, **-W**, and **-s** flags.
-r	Resets the high water mark of I/O memory entitlement once at the beginning of the command. You can use this flag only with the **-m** and **-e** flags.
-R	Resets the high water mark at the beginning of each monitoring interval. If you specify both the **-r** and **-R** flags, the **-R** flag takes effect.
-s	Displays LPAR information. The **-s** flag displays the following details: **Service partition ID** The service partition ID as assigned by the Hardware Management Console (HMC). **Number of configured LPARs** The number of LPARs that are configured on the HMC. You can specify the **-s** flag alone or with the **-P**, **-i**, **-W** and **-N** flags.
-t	Displays the time in the *HH:MM:SS* format when the command is run with intervals.
-W	Lists details of the workload partition (WPAR) configuration. If the command is run from the global environment, the WPAR Key value is 0. The **-W** flag displays the following details: **WPAR Key** WPAR static identifier. **WPAR Configured ID** WPAR dynamic identifier. **WPAR Maximum CPUs** Number of processors in a resource set. It displays the value of 0 if it is not restricted. **WPAR Effective CPUs** Number of processors in an effective resource set. It displays the value of 0 if it is not restricted. **WPAR CPU Percentage** WPAR processor-limit percentage. You can specify the **-W** flag alone or with **-P**, **-i**, **-s**, and **-N** flags.
-x	Lists the security mode settings for the LPAR.
-X	Generates the XML output. The default file name is **lparstat_DDMMYYHHMM.xml**, unless the user specifies a different file name with the **–o** option.
-u	Displays the expiration date of the AIX Update Access Key (UAK) of the server, the expiration date of the firmware UAK of the server, and the image date of the AIX operating system.

Note: If Pool Utilization Authority (PUA) is not available, the app column is not displayed.

Examples

1. To get the default LPAR statistics, enter the following command:

```
lparstat 1 1
```

2. To get default LPAR statistics with summary statistics on Hypervisor, enter the following command:

```
lparstat -h 1 1
```

3. To get the information about the partition, enter the following command:

```
lparstat -i
```

4. To get detailed Hypervisor statistics, enter the following command:

```
lparstat -H 1 1
```

5. To get statistics about the shared memory pool and the I/O memory entitlement of the partition, enter the following command:

```
lparstat -m
```

6. To get statistics about I/O memory pools inside the LPAR, enter the following command:

```
lparstat -me
```

7. If the LPAR is running in shared mode and with 4 SMT threads the type would be in the following format:

```
Type - Shared-SMT-4
```

8. If the LPAR is running in dedicated mode and with 2 SMT threads the type would be in the following format:

```
Type - Dedicated-SMT
```

9. To calculate the memory compression statistics in an LPAR when Active Memory Expansion is enabled, enter the following command:

```
lparstat -c 1 1
```

10. To get statistics about page coalescing inside an LPAR, enter the following command:

```
lparstat -mp
```

11. To check whether you can perform the Live Partition Mobility operations on an LPAR, enter the following command:

```
lparstat -L
```

An output similar to the following example is displayed based on the LPM capability of the LPAR:

```
Live Partition Mobility: Enabled
```

Files

Item	Description
`/usr/bin/lparstat`	Contains the `lparstat` command.

lpc Command

Note: This is a System V Print Subsystem command.

Purpose

Provides (BSD) line printer control.

Syntax

/usr/ucb/lpc [*Command* [*Parameter* . . .]]

Description

The **lpc** command controls the operation of the printer or of multiple printers. The **lpc** command can be used to start or stop a printer, disable or enable a printer's spooling queue, rearrange the order of jobs in a queue, or display the status of each printer, along with its spooling queue and printer daemon.

If you enter `lpc -?`, the system displays the command usage message and returns 0.

With no parameters, the **lpc** command runs interactively, prompting with `lpc>`. If parameters are supplied, the **lpc** command interprets the first as a *Command* to execute; each subsequent parameter is taken as a *Parameter* for that command. The standard input can be redirected so that the **lpc** command reads *Command*s from a file.

*Command*s may be abbreviated to an unambiguous substring.

> **Note:** The *printer* parameter is specified just by the name of the printer (as **lw**), not as you would specify it to **lpr** or **lpq** (not as **-Plw**).

Item	Description
? [*Command*. . .]	
help [*Command* . . .]	Displays a short description of each command specified in the parameter list or, if no parameters are given, a list of the recognized commands.
abort [**all** | [*Printer* . . .]]	Terminates an active spooling daemon on the local host immediately and then disables printing (preventing new daemons from being started by **lpr**) for the specified printers. The **abort** command can only be used by a privileged user.
clean [**all** | [*Printer* . . .]]	Removes all files created in the spool directory by the daemon from the specified printer queues on the local machine. The **clean** command can only be used by a privileged user.
disable [**all** | [*Printer* . . .]]	Turns the specified printer queues off. This prevents new printer jobs from being entered into the queue by **lpr**. The **disable** command can only be used by a privileged user.
down [**all** | [*Printer*. . .]] [*Message*]	Turns the specified printer queue off, disables printing, and puts *Message* in the printer status file. The message does not need to be quoted. The remaining parameters are treated like **echo**. This is normally used to take a printer down and let others know why (**lpq** indicates that the printer is down, as does the **status** command).
enable [**all** | [*Printer*. . .]]	Enables spooling on the local queue for the listed printers so that **lpr** can put new jobs in the spool queue. The **enable** command can only be used by a privileged user.
exit	Exits from **lpc**.
quit	Quits from **lpc**.

Item	Description
restart [**all** | [*Printer*....]]	Attempts to start a new printer daemon. This is useful when some abnormal condition causes the daemon to die unexpectedly leaving jobs in the queue. This command can be run by any user.
start [**all** | [*Printer*...]]	Enables printing and starts a spooling daemon for the listed printers. The **start** command can only be used by a privileged user.
status [**all** | [*Printer*...].]	Displays the status of daemons and queues on the local machine. This command can be run by any user.
stop [**all** | [*Printer*...]]	Stops a spooling daemon after the current job completes and disable printing. The **stop** command can only be used by a privileged user.
topq *Printer* [*Job#*...] [*User*...]	Moves the print jobs specified by *Job#* or those jobs belonging to *User* to the top (head) of the printer queue. The **topq** command can only be used by a privileged user.
up [**all** | [*Printer*...]]	Enables everything and starts a new printer daemon. Undoes the effects of **down**.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/var/spool/lp/*	
/var/spool/lp/system/pstatus	

Error Codes

Item	Description
?Ambiguous command	The abbreviation matches more than one command.
?Invalid command	A command or abbreviation is not recognized.
?Privileged command	The command can be executed only by the privileged user.
lpc: *printer***: unknown printer to the print service**	The *printer* was not found in the System V LP database. Usually this is a typing mistake; however, it may indicate that the printer does not exist on the system. Use **lptstat -p** to find the reason.

Item	Description
lpc: error on opening queue to spooler	The connection to **lpsched** on the local machine failed. This usually means the printer server started at boot time has died or is hung. Check if the printer spooler daemon **/usr/lib/lp/lpsched** is running.
lpc: Can't send message to LP print service	
lpc: Can't receive message from LP print service	Indicates that the LP print service has been stopped. Get help from the system administrator.
lpc: Received unexpected message from LP print service	It is likely there is an error in this software. Get help from system administrator.

lpd Command

Purpose

Provides the remote print server on a network.

Syntax

lpd [**-d**] [**-l**] [**-D** *DebugOutputFile*]

Description

The **lpd** daemon is the remote print server. It monitors port 515 for print requests. Each request is placed in a directory named **/var/spool/lpd**.

A computer on a network (host) that can create a Transmission Control Protocol/Internet Protocol (TCP/IP) data stream and use the **lpd** protocol can print remotely or act as a print server. As a security feature, the **lpd** daemon accepts print requests only from remote hosts that are listed in the local **/etc/hosts.equiv** or **/etc/hosts.lpd** file.

The **lpd** daemon can run on any host in the network; its function is to accept print requests from foreign hosts (on port 515). The **lpd** daemon handles each request by forking a child process. Remote requests are first checked against the **/etc/hosts.equiv** and **/etc/hosts.lpd** files for permission to print on the local host.

Changes can be made to the **/etc/hosts.equiv** and **/etc/hosts.lpd** files without restarting the system. To put changes to these files into effect without restarting the system, use the System Resource Controller (SRC) **refresh** command. This command causes the **/etc/hosts.equiv** and **/etc/hosts.lpd** database files to be reloaded and the changes implemented.

> **Note:** The queuing system does not support multibyte host names.

The **/etc/locks/lpd** file contains the process ID of the currently running instance of the **lpd** daemon. If the current machine becomes inoperable, you may need to remove the ID for the **lpd** daemon when the system starts up again. The error message displayed is `lpd: lock file or duplicate daemon`.

Manipulating the lpd Daemon with the System Resource Controller

The **lpd** daemon is a subsystem controlled by the System Resource Controller (SRC). The **lpd** daemon is a member of the TCP/IP system group.

Use the following SRC commands to manipulate the **lpd** daemon:

Item	Description
startsrc	Starts a subsystem, group of subsystems, or a subserver.

Item	Description
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
refresh	Causes the subsystem or group of subsystems to reread the appropriate configuration file.
traceson	Enables tracing of a subsystem, group of subsystems, or a subserver.
tracesoff	Disables tracing of a subsystem, group of subsystems, or a subserver.
lssrc	Gets the status of a subsystem, group of subsystems, or a subserver.

Flags

Item	Description
-d	Sends a status of Inactive to be logged with the SRC controller and sends error messages during socket communication setup failures to the user display.
-l	Sends a status of Active to be logged with the SRC controller and sends valid or invalid job request messages to the user display.
-D *DebugOutputFile*	Sends extensive debugging output used for problem determination to the file specified by *DebugOuputFile*. This should only be used during problem determination as the *DebugOuputFile* can grow large rapidly. If the output file specified already exists, new debugging output is appended to the end of it. If there are any problems creating or writing to the output file, the debugging option is ignored.

Examples

1. To start the **lpd** server daemon, enter:

   ```
   startsrc -s lpd
   ```

2. To start the **lpd** server daemon while enabling the display of certain error messages, enter:

   ```
   startsrc -s lpd -a " -d"
   ```

3. To send logging information to the **stderr** daemon, enter:

   ```
   startsrc -s lpd -a " -l"
   ```

4. To start the **lpd** server daemon in debugging mode with output going to **/tmp/dbglpd.out**, enter:

   ```
   startsrc -s lpd -a " -D /tmp/dbglpd.out"
   ```

Files

Item	Description
/usr/sbin/lpd	Specifies the path to the **lpd** daemon.
/dev/lp*	Contains the names of print devices.
/etc/hosts.equiv	Contains the names of hosts allowed to execute commands and print.
/etc/hosts.lpd	Contains the names of hosts allowed to print only.
/var/spool/lpd	Contains the spool directory for control, status, and data files.

Item	Description
/etc/locks/lpd	Contains the PID of the currently running **lpd** daemon. After a system crash, this PID may need to be deleted. The following error message indicates the problem: `lpd: lock file or duplicate daemon`

lpfilter Command

Note: This is a System V Print Subsystem command.

Purpose

Administers filters used with the LP print service.

Syntax

lpfilter -f *FilterName* **-F** *PathName*

lpfilter -f *FilterName* **-**

lpfilter -f *FilterName* **-i**

lpfilter -f *FilterName* **-x**

lpfilter -f *FilterName* **-l**

Description

The **lpfilter** command is used to add, change, delete, and list a filter used with the LP print service. These filters are used to convert the content type of a file to a content type acceptable to a printer.

If you enter `lpfilter -?`, the system displays the command usage message and returns 0.

Flags

Item	Description
- (hyphen)	Adds or changes a filter as specified from standard input.
-f *FilterName*	Specifies the name of the filter to be added, changed, deleted, or listed.
-F *PathName*	Add or changes a filter as specified by the contents of the file pathname.
-i	Resets an original filter to its original settings.
-l	Lists a filter description.
-x	Deletes a filter.

The parameter **all** can be used instead of a *FilterName* with any of these flags. When **all** is specified with the **-F** or **-** flag, the requested change is made to all filters. Using **all** with the **-i** flag has the effect of restoring to their original settings all filters for which predefined settings were initially available. Using the **all** parameter with the **-x** flag results in all filters being deleted, and using it with the **-l** flag produces a list of all filters.

Adding or changing a filter

The filter named in the **-f** flag is added to the filter table. If the filter already exists, its description is changed to reflect the new information in the input.

The filter description is taken from the *PathName* if the **-F** flag is given or from the standard input if the **-** flag is specified. One of the two must be given to define or change a filter. If the filter named is one originally delivered with the LPprint service, the **-i** flag restores the original filter description.

When an existing filter is changed with the **-F** flag or the **-** flag, items that are not specified in the new information are left as they were. When a new filter is added with this command, unspecified items are given default values.

Filters are used to convert the content of a request into a data stream acceptable to a printer. For a given print request, the LP print service knows the following:

- Content in the request
- Name of the printer
- Type of the printer
- Types of content acceptable to the printer
- Modes of printing asked for by the originator of the request

It uses this information to find a filter or a pipeline of filters that converts the content into a type acceptable to the printer.

A list of items that provide input to this command and a description of each item follows. All lists are comma or space separated.

- Input types: *content-type-list*
- Output types: *content-type-list*
- Printer types: *printer-type-list*
- Printers: *printer-list*
- Filter type: *filter-type*
- Command: *shell-command*
- Flags: *template-list*

Item	Description
Input types	Gives the types of content that can be accepted by the filter. (The default is any.)
Output types	Gives the types of content that the filter can produce from any of the input content types. (The default is any.)
Printer types	Gives the type of printers for which the filter can be used. The LP print service restricts the use of the filter to these types of printers. (The default is any.)
Printers	Gives the names of the printers for which the filter can be used. The LP print service restricts the use of the filter to just the printers named. (The default is **any**.)
Filter type	Marks the filter as a **slow** filter or a **fast** filter. Slow filters are generally those that take a long time to convert their input. They are run unconnected to a printer to keep the printers from being tied up while the filter is running. If a listed printer is on a remote system, the filter type for it must have the value **slow**. Fast filters are generally those that convert their input quickly or those that must be connected to the printer when run. These are given to the interface program IP to run connected to the physical printer.
Command	Specifies the program to run to invoke the filter. The full program pathname as well as fixed flags must be included in the *shell-command*; additional flags are constructed, based on the characteristics of each print request and on the "flags'' field. A command must be given for each filter. The command must accept a data stream as standard input and produce the converted data stream on its standard output. This allows filter pipelines to be constructed to convert data not handled by a single filter.

Item	Description
Flags	Specifies the comma-separated list of templates used by the LP print service to construct flags to the filter from the characteristics of each print request listed in the table later. In general, each template is of the following form: *keyword-pattern=replacement* The *keyword* names the characteristic that the template attempts to map into a filter-specific flag; each valid *keyword* is listed in the table below. A *pattern* is one of the following: a literal pattern of one of the forms listed in the table, a single asterisk (*), or a regular expression. If *pattern* matches the value of the characteristic, the template fits and is used to generate a filter-specific flag. The *replacement* is what is used as the flag.

Regular expressions are the same as those found in the **ed** or **vi** commands. This includes the \(. . . \) and *n* constructions, which can be used to extract portions of the *pattern* for copying into the *replacement*, and the ` `&'', which can be used to copy the entire *pattern* into the *replacement*.

The *replacement* can also contain a ` `*''. It too, is replaced with the entire *pattern*, just like the ` `&'' of the **ed** command.

lp flag	Properties
-T	**Characteristic** Content type (input) **Keyword** INPUT **Possible patterns** content-type
N/A	**Characteristic** Content type (output) **Keyword** OUTPUT **Possible patterns** content-type
N/A	**Characteristic** Printer type **Keyword** TERM **Possible patterns** printer-type
-d	**Characteristic** Printer name **Keyword** PRINTER **Possible patterns** printer-name

lp flag	Properties
-f, -o cpi=	**Characteristic** Character pitch **Keyword** CPI **Possible patterns** integer
-f, -o lpi=	**Characteristic** Line pitch **Keyword** LPI **Possible patterns** integer
-f, -o length=	**Characteristic** Page length **Keyword** LENGTH **Possible patterns** integer
-f, -o width=	**Characteristic** Page width **Keyword** WIDTH **Possible patterns** integer
-P	**Characteristic** Pages to print **Keyword** PAGES **Possible patterns** page-list
-S	**Characteristic** Character set Print wheel **Keyword** CHARSET CHARSET **Possible patterns** character-set-name print-wheel-name
-f	**Characteristic** Form name **Keyword** FORM **Possible patterns** form-name

lp flag	Properties
-y	**Characteristic** Modes **Keyword** MODES **Possible patterns** mode
-n	**Characteristic** Number of copies **Keyword** COPIES **Possible patterns** integer

For example, the template `MODES landscape = -l` shows that if a print request is submitted with the **-y landscape** flag, the filter is given the flag **-l**. As another example, the template `TERM * = -T *` shows that the filter is given the flag **-T** *printer-type* for whichever *printer-type* is associated with a print request using the filter.

As a last example, consider the template `MODES prwidth\=\(.*\) = -w\1`. Suppose a user gives the command **lp -y prwidth=10**

From the table above, the LP print service determines that the **-y** flag is handled by a **MODES** template. The **MODES** template here works because the pattern ``prwidth\=\(.*\)'' matches the **prwidth=10** given by the user. The *replacement* **-w\1** causes the LP print service to generate the filter flag **-w10**.

If necessary, the LP print service constructs a filter pipeline by concatenating several filters to handle the user's file and all the print flags. If the print service constructs a filter pipeline, the **INPUT** and **OUTPUT** values used for each filter in the pipeline are the types of the input and output for that filter, not for the entire pipeline.

Deleting a filter

The **-x** flag is used to delete the filter specified in *FilterName* from the LP filter table.

Listing a filter description

The **-l** flag is used to list the description of the filter named in *FilterName*. If the command is successful, the following message is sent to standard output:

```
Input types: content-type-list
   Output types: content-type-list
   Printer types: printer-type-list
   Printers: printer-list
   Filter type: filter-type
   Command: shell-command
   flags: template-list
```

If the command fails, an error message is sent to standard error.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

lpforms Command

Note: This is a System V Print Subsystem command.

Purpose

Administer forms used with the LP print service.

Syntax

lpforms -f *FormName Options*

lpforms -f *FormName* **-A** *AlertType* [**-Q** *minutes*] [**-W** *requests*]

Description

The **lpforms** command is used to administer the use of preprinted forms, such as company letterhead paper, with the LP print service. A form is specified by its *FormName*. Users may specify a form when submitting a print request. The parameter **all** can be used instead of *FormName* with either of the command lines shown above. The first command line allows the administrator to add, change, and delete forms, to list the attributes of an existing form, and to allow and deny users access to particular forms. The second command line is used to establish the method by which the administrator is alerted that the form *FormName* must be mounted on a printer.

If you enter `lpforms -?`, the system displays the command usage message and returns 0.

With the first **lpforms** command line, one of the following flags must be used:

Flags

Item	Description
- (hyphen)	Adds or changes form *FormName*, as specified by the information from standard input.
-F *pathname*	Adds or changes form *FormName*, as specified by the information in *pathname*.
-l	Lists the attributes of form *FormName*.
-x	Deletes form *FormName* (this flag must be used separately; it may not be used with any other flag).

Adding or changing a form

The **-F** *pathname* flag is used to add a new form, *FormName*, to the LP print service, or to change the attributes of an existing form. The form description is taken from *pathname* if the **-F** flag is given, or from the standard input if the **-** flag is used. One of these two flags must be used to define or change a form. *pathname* is the pathname of a file that contains all or any subset of the following information about the form:

```
Page length: scaled-decimal-number1
Page width: scaled-decimal-number2
Number of pages: integer
Line pitch: scaled-decimal-number3
Character pitch: scaled-decimal-number4
Character set choice: character-set/print-wheel [mandatory]
Ribbon color: ribbon-color
Comment:
comment
Alignment pattern: [content-type]
content
```

The term "scaled-decimal-number" refers to a non-negative number used to indicate a unit of size. The type of unit is shown by a "trailing" letter attached to the number. Three types of scaled decimal numbers can be used with the LP print service: numbers that show sizes in centimeters (marked with a trailing "c"); numbers that show sizes in inches (marked with a trailing "i"); and numbers that show sizes in units appropriate to use (without a trailing letter), that is, lines, characters, lines per inch, or characters per inch.

Except for the last two lines, the above lines may appear in any order. The Comment and *comment* items must appear in consecutive order but may appear before the other items, and the "Alignment pattern" and the *content* items must appear in consecutive order at the end of the file. Also, the *comment* item may not contain a line that begins with any of the key phrases above, unless the key phrase is preceded with a ">". Any leading ">" sign found in the *comment* are removed when the comment is displayed. Case distinctions in the key phrases are ignored.

When this command is issued, the form specified by *FormName* is added to the list of forms. If the form already exists, its description is changed to reflect the new information. Once added, a form is available for use in a print request, except where access to the form has been restricted, as described under the **-u** flag. A form may also be allowed to be used on certain printers only.

A description of each form attribute is below:

Item	Description
Page length and Page width	Before printing the content of a print request needing this form, the generic interface program provided with the LP print service initializes the physical printer to handle pages *scaled-decimal-number1* long, and *scaled-decimal-number2* wide using the printer type as a key into the *terminfo* database. The page length and page width are also passed, if possible, to each filter used in a request needing this form.
Number of pages	Each time the alignment pattern is printed, the LP print service attempts to truncate the *content* to a single form by, if possible, passing to each filter the page subset of 1-*integer*.
Line pitch and Character pitch	Before printing the content of a print request needing this form, the interface programs provided with the LP print service initializes the physical printer to handle these pitches, using the printer type as a key into the *terminfo* database. Also, the pitches are passed, if possible, to each filter used in a request needing this form. *scaled-decimal-number3* is in lines per centimeter if a "c" is appended, and lines per inch otherwise; similarly, *scaled-decimal-number4* is in characters per centimeter if a "c" is appended, and characters per inch otherwise. The character pitch can also be given as **elite** (12 characters per inch), **pica** (10 characters per inch), or **compressed** (as many characters per inch as possible).

Item	Description
Character set choice	When the LP print service alerts an administrator to mount this form, it also mentions that the print wheel *print-wheel* should be used on those printers that take print wheels. If printing with this form is to be done on a printer that has selectable or loadable character sets instead of print wheels, the interface programs provided with the LP print service automatically selects or loads the correct character set. If **mandatory** is appended, a user is not allowed to select a different character set for use with the form; otherwise, the character set or print wheel named is a suggestion and a default only.
Ribbon color	When the LP print service alerts an administrator to mount this form, it also mentions that the color of the ribbon should be *ribbon-color*.
Comment	The LP print service displays the *comment* unaltered when a user asks about this form.
Alignment pattern	When mounting this form, an administrator can ask for the *content* to be printed repeatedly, as an aid in correctly positioning the preprinted form. The optional *content-type* defines the type of printer for which *content* had been generated. If *content-type* is not given, **simple** is assumed. **Note:** The content is stored as given and is readable only by the user *lp*.

When an existing form is changed with this command, items missing in the new information are left as they were. When a new form is added with this command, missing items gets the following defaults:

```
Page Length: 66
Page Width: 80
Number of Pages: 1
Line Pitch: 6
Character Pitch: 10
Character Set Choice: any
Ribbon Color: any
```

Deleting a form

The **-x** flag is used to delete the form *FormName* from the LP print service.

Listing form attributes

The **-l** flag is used to list the attributes of the existing form *FormName*. Because of the potentially sensitive nature of the alignment pattern, only the administrator can examine the form with this command. Other people may use the **lpstat** command to examine the non-sensitive part of the form description.

Allowing and denying access to a form

The **-u** flag, followed by the parameter **allow:***login-ID-list* or **-u deny:***login-ID-list* lets you determine which users are allowed to specify a particular form with a print request. This flag can be used with the **-F** or **-** flag.

The *login-ID-list* parameter may include any or all of the following constructs:

Item	Description
login-ID	A user on the local system

Item	Description
*system-name***!***login-ID*	A user on system *system-name*
*system-name***!all**	All users on system *system-name*
all!*login-ID*	A user on all systems
all	All users on the local system
all!all	All users on all systems

The default value of *login-ID-list* is **all**.

The LP print service keeps two lists of users for each form: an "allow-list" of people allowed to use the form, and a "deny-list" of people that may not use the form.

- If allow-list is present and *login-ID* is in it, access is allowed.
- If only deny-list is present and *login-ID* is not in it, access is allowed.
- If *login-ID* is in deny-list, access is denied.
- If neither allow-list or deny-list are present, access is denied.
- If both lists are present, and *login-ID* is in neither, access is denied.
- If only allow-list is present and *login-ID* is not in it, access is denied.

If the allow-list is not empty, only the users in the list are allowed access to the form, regardless of the contents of the deny-list. If the allow-list is empty but the deny-list is not, the users in the deny-list may not use the form (but all others may use it).

All users can be denied access to a form by specifying **-f deny:all**. All users can be allowed access to a form by specifying **-f allow:all**. (This is the default.)

Setting an alert to mount a form

The **-f** *FormName* flag is used with the **-A** *AlertType* flag to define an alert to mount the form when there are queued jobs which need it. If this flag is not used to arrange alerting for a form, no alert is sent for that form.

The method by which the alert is sent depends on the value of the *AlertType* parameter specified with the **-A** flag. The alert types are the same as those available with the **-A** flag to **lpadmin**: **mail**, **write**, **quiet**, **none**, *shell-command*, and **list**.

The message sent appears as follows:

```
   The form FormName needs to be mounted
   on the printer(s):
   printer (integer1 requests).
   integer2 print requests await this form.
   Use the ribbon-color ribbon.
   Use the print-wheel print wheel, if appropriate.
```

The printers listed are those that the administrator had earlier specified were candidates for this form. The number *integer1* listed next to each printer is the number of requests eligible for the printer. The number *integer2* shown after the list of printers is the total number of requests awaiting the form. It is less than the sum of the other numbers if some requests can be handled by more than one printer. The *ribbon-color* and *print-wheel* are those specified in the form description. The last line in the message is always sent, even if none of the printers listed use print wheels, because the administrator may choose to mount the form on a printer that does use a print wheel.

Where any color ribbon or any print wheel can be used, the statements above read:

```
Use any ribbon.
   Use any print-wheel.
```

If *FormName* is **any**, the alerting defined in this command applies to any form for which an alert has not yet been defined. If *FormName* is **all**, the alerting defined in this command applies to all forms.

If the **-W** flag is not given, the default procedure is that only one message is sent per need to mount the form. Not specifying the **-W** flag is equivalent to specifying **-W once** or **-W 0**. If *minutes* is a number greater than 0, an alert is sent at intervals specified by *minutes*.

If the **-Q** flag is also given, the alert is sent when a certain number (specified by the parameter *requests*) of print requests that need the form are waiting. If the **-Q** flag is not given, or the value of *requests* is 1 or **any** (which are both the default), a message is sent as soon as anyone submits a print request for the form when it is not mounted.

Listing the current alert

The **-f** flag, followed by the **-A** flag and the parameter **list** is used to list the type of alert that has been defined for the specified form *FormName*. No change is made to the alert. If *FormName* is recognized by the LP print service, one of the following lines is sent to the standard output, depending on the type of alert for the form.

```
When requests requests are queued:
alert with shell-command every minutes minutes

When requests requests are queued:
write to user-name every minutes minutes

When requests requests are queued:
mail to user-name every minutes minutes

No alert
```

The phrase "every *minutes* minutes" is replaced with "once" if *minutes* (**-W** *minutes*) is 0.

Terminating an active alert

The **-A quiet** flag is used to stop messages for the current condition. An administrator can use this flag to temporarily stop receiving further messages about a known problem. Once the form has been mounted and then unmounted, messages are again sent when the number of print requests reaches the threshold *requests*.

Removing an alert definition

No messages are sent after the **-A none** flag is used until the **-A**flag is given again with a different *AlertType*. This can be used to permanently stop further messages from being sent as any existing alert definition for the form is removed.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

lphistory Command

Purpose

Displays or clears the history list of least-privilege (LP) commands that have been run during the current resource monitoring and control (RMC) session.

Syntax

- To list a particular number of previously-issued commands:
  - On the local node:

    lphistory [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-L a | c | e | m | n | t | u | x**] [**-h**] [**-TV**] [*num_records*]

– On all nodes in a domain:

lphistory -a [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-L a | c | e | m | n | t | u | x**] [**-h**] [**-TV**] [*num_records*]

– On a subset of nodes in a domain:

lphistory -n *host1*[**,***host2*...] [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-L a | c | e | m | n | t | u | x**] [**-h**] [**-TV**] [*num_records*]

- To clear the history list:

– On the local node:

lphistory -c [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-h**] [**-TV**]

– On all nodes in a domain:

lphistory -c -a [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-h**] [**-TV**]

– On a subset of nodes in a domain:

lphistory -c -n *host1*[**,***host2*...] [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-h**] [**-TV**]

Description

The **lphistory** command lists the history of LP commands that have been run by the least-privilege resource manager. The command history is maintained as records in the RSCT audit log. By default, only the command string (the path name plus arguments) from each audit log record is listed. The **-L** flag controls the output format of **lphistory**; use it to display specific fields as needed. The selection flags (**-B**, **-C**, **-E**, **-m**, **-S**, or **-u**) control the selection string that is passed to **lsaudrec**.

The **lphistory** command takes one optional parameter: the number of records to list. The default value of *num_records* is 10. If none of the selection flags is used, the latest number of records in the audit log (specified by *num_records*) are listed. Otherwise, the latest number of records (specified by *num_records*) from those selected by one or more of the selection flags are listed. This selection process applies to the audit records on each node specified by the **-a** flag or the **-n** flag. If neither **-a** nor **-n** is specified, the selection process applies to the audit records on the local node.

The **-B** and **-E** flags take time stamps as arguments. Time stamps are in the form *MMddhhmmyyyy*, where *MM* is the two-digit month (01-12), *dd* is the two-digit day of the month (01-31), *hh* is the two-digit hour (00-23), *mm* is the two-digit minute (00-59), and *yyyy* is the four-digit year.

You can use the wild card character (%) with identity-related arguments (*user_ID*, *mapped_ID*) and command names. The % can be placed at the beginning or end of the string, or anywhere within it. You cannot use any wild card characters when specifying *command_path*.

You can remove audit log records using the **-c** flag. If none of the selection flags is specified, all audit log records for the least-privilege resource manager are removed. Otherwise, the records selected by one or more of the selection flags are removed. The **-c** flag cannot be used with the **-L** flag or the *num_records* parameter.

Flags

-a

Displays previously-issued LP commands for all nodes in the domain.

The **CT_MANAGEMENT_SCOPE** environment variable determines the scope of the cluster. If **CT_MANAGEMENT_SCOPE** is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and

CT_MANAGEMENT_SCOPE is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set **CT_MANAGEMENT_SCOPE** to 2.

You cannot specify this flag with the **-n** flag.

-B *MMddhhmmyyyy*
Specifies a beginning time stamp in the form *MMddhhmmyyyy*, where *MM* is the two-digit month (01-12), *dd* is the two-digit day (01-31), *hh* is the two-digit hour (00-23), *mm* is the two-digit minute (00-59), and *yyyy* is the four-digit year. The time can be truncated from right to left, except for *MM*. If not all digits are specified, the year defaults to the current year, minutes to 0, hour to 0, and day to 01. At a minimum, the month must be specified. The command lists or removes only those records that were created at or after this time.

–c
Clears the history of LP commands. You cannot specify this flag with the *number_of_commands* parameter or the **-n** flag.

-C *command_name*
Specifies a command name. **lphistory -C** lists or removes only those records that contain *command_name*, which is the name of a command without a fully-qualified path (**mkrsrc**, for example). You can use wild card characters in *command_name*.

-E *MMddhhmmyyyy*
Specifies an ending time stamp in the form *MMddhhmmyyyy*, where *MM* is the two-digit month (01-12), *dd* is the two-digit day (01-31), *hh* is the two-digit hour (00-23), mm is the two-digit minute (00-59), and *yyyy* is the four-digit year. The time can be truncated from right to left, except for *MM*. If not all digits are specified, the year defaults to the current year, minutes to 0, hour to 0, and day to 01. At a minimum, the month must be specified. The command lists or removes only those records that were created at or before this time.

-L a | c | e | m | n | t | u | x
By default, only the command string (path name plus arguments) from each audit log record is listed. If this flag is specified, the argument is one or more of the following letters; the fields are displayed in the same order as the letters in the flag argument.

a
Displays all fields from the audit log in the following order: **t**, **u**, **m**, **n**, **x**, **c** (specifying **-L** a is the same as specifying **-L tumnxc**)

c
Displays the command string (the default)

e
Displays the standard error output

m
Displays the mapped identity

n
Displays the name of the node where the command ran

t
Displays the time field

u
Displays the authenticated user identity

x
Displays the LP command exit status

You cannot specify this flag with the **-c** flag.

-m *mapped_ID*
Specifies a mapped identity. **lphistory -m** lists or removes only those records that contain *mapped_ID*. You can use wild card characters in *mapped_ID*.

-n *host1*[,*host2*,...]
Specifies one or more nodes in the cluster on which the LP command history list is to be retrieved or cleared. (By default, the history list for the local node is retrieved or cleared.)

This flag is valid only in a management domain or a peer domain. If the **CT_MANAGEMENT_SCOPE** environment variable is not set, management domain scope is chosen first (if a management domain exists) and then peer domain scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds.

You cannot specify this flag with the **-a** flag.

-S *command_path*
Specifies a command path name. **lphistory -S** lists or removes only those records that contain *command_path*, which is identical to the value of the **CommandPath** in the LPCommands class (**/opt/rsct/bin/mkrsrc**, for example). You cannot use wild card characters in *command_path*.

-u *user_ID*
Specifies an authenticated user identity. **lphistory -u** lists or removes only those records that contain *user_ID*. You can use wild card characters in *user_ID*.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error.

-V
Writes the command's verbose messages to standard output.

Parameters

num_records
Specifies the number of commands to be displayed from the history list. You can list a minimum of one command and a maximum of 100 commands. The default value is 10. You cannot specify this parameter with the **-c** flag.

Security

To run the `lphistory` command, you need write permission in the Class ACL of the `IBM.LPCommands` resource class. Permissions are specified in the LP ACLs on the contacted system. See the `lpacl` file for general information about LP ACLs and the *RSCT Administration Guide* for information about modifying them.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
The resource was not found.

Environment Variables

CT_CONTACT
Determines the system that is used for the session with the RMC daemon. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the LP resources that are processed.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to process the LP resources. The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. When the -V flag is specified, this command's verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To list 20 LP commands that were previously issued on the local node, enter:

```
lphistory 20
```

2. Suppose nodeA is in a management domain and CT_MANAGEMENT_SCOPE is set to 3. To list the LP command history on nodeA, enter:

```
lphistory -c -n nodeA
```

3. To display the last 15 LP commands invoked with time, user ID, mapped ID, mechanism, return code, standard error, command name, and command string, enter:

```
lphistory -L a 15
```

4. To display the LP command names that end with rsrc, enter:

```
lphistory -C %rsrc
```

5. To display the LP commands that were invoked after 11:30 PM on April 18, 2006, enter:

```
lphistory -B 041823302006
```

Location

/opt/rsct/bin/lphistory
Contains the `lphistory` command.

lpmove Command

Note: This is a System V Print Subsystem command.

Purpose

Moves print requests.

Syntax

lpmove *Requests Destination*

lpmove *Destination1 Destination2*

Description

The **lpmove** command moves requests that were queued by **lp** between LP destinations. This command moves a specific *Request* to the specified *Destination*. *Requests* are request-IDs returned by **lp**. You can also attempt to move all requests for *Destination1* to *Destination2*. This form of the **lpmove** command causes **lp** to reject any new requests for *Destination1*.

Note: When moving requests, **lpmove** never checks the acceptance status of the new destination. Also, the request-IDs of the moved requests are not changed, so you can still find their requests. The **lpmove** command does not move requests that have options (such as content type and form required) that cannot be handled by the new destination.

If a request was originally queued for a class or the special destination **any** and the first form of **lpmove** was used, the destination of the request is changed to *New-Ddestination*. A request thus affected is printable only on *New-Destination* and not on other members of the class or other acceptable printers if the original destination was **any**.

If you enter `lpmove -?`, the system displays the command usage message and returns 0.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Files

Item	Description
/var/spool/lp/*	

lppchk Command

Purpose

Verifies files of an installable software product.

Syntax

lppchk [**-R** { **Path** | **ALL** }] { **-c**[**u**] | **-f** | **-l** [**u**] | **-v** } [**-m** [**1** | **2** | **3**]] [**-O** { [**r**] [**s**] [**u**] }] [*ProductName* [*FileList* ...]]

Description

The **lppchk** command verifies that files for an installable software product (fileset) match the Software Vital Product Data (SWVPD) database information for file sizes, checksum values, or symbolic links. A fileset is a separately installable option of a software package.

Flags

Item	Description
-c	Performs a checksum operation on the *FileList* items and verifies that the checksum and the file size are consistent with the SWVPD database.
-f	Checks that the *FileList* items are present and the file size matches the SWVPD database.
-l	Verifies symbolic links for files as specified in the SWVPD database.
-m [**1**|**2**|**3**]	Displays three levels of information. The levels are as follows: **1** Error messages only (default). **2** Error messages and warnings. **3** Error messages, warnings and informational messages.
-O {[**r**][**s**][**u**]}	Verifies the specified parts of the program. This flag is not needed with standalone systems because without this option all parts are verified by default. The flags specify the following parts: **r** Indicates the **/** (root) part is to be verified. **s** Indicates the **/usr/share** part is to be verified. **u** Indicates the **/usr** part is to be verified.
-R { **Path**|**ALL** }	Indicates a user-specified installation location.
-u	Updates the SWVPD with new checksum or size information from the system when the system information does not match the SWVPD database. This flag sets symbolic links that are found to be missing. This flag is valid with only the **-c** or **-l** flag.
-v	Verifies that the **/** (root), **/usr** and **/usr/share** parts of the system are valid with each other. In other words, this flag verifies that all software products installed on the **/** (root) file system are also installed on the **/usr** file system and, conversely, all the software products installed in the **/usr** file system are also installed on the **/** (root) file system. You cannot specify *FileList* items with this flag. This flag also verifies requisites. **Note:** Only one of the **-c**, **-f**, **-l**, and **-v** flags can be specified with each use of the **lppchk** command.

Parameters

Item	Description
FileList	Specifies the file or files to check. This parameter is a list of file names separated by spaces. The file names can be a single name or a pair of names separated by a colon. The first form specifies a simple file and the second form specifies a member of an archive file, where the first name specifies the member and the second name specifies the archive file that contains the member. The full path name of the file or files must be specified. To specify multiple files you can use the pattern-matching characters ***** (asterisk) and **?** (question mark), but they should be enclosed in a pair of **'**s (single quotes). Single quotes are recommended to prevent the korn shell wildcard expansion. If this parameter is omitted, all files of a software product are checked. If this parameter is specified, it must be preceded by a software product name.
ProductName	Specifies the name of the software product whose files are to be checked. If this parameter is omitted, all software products in the SWVPD are checked. To specify multiple software products you can use the pattern-matching characters ***** (asterisk) and **?** (question mark), but they must be enclosed in a pair of **'**s (single quotes) to prevent the shell from expanding them.

Exit Status

Item	Description
0 (zero)	The command completed successfully.
nonzero	An error was found.

The **lppchk** command returns zero if no errors were found. Any other return value indicates an error was found.

Note: If **lppchk -f** (size) or **lppchk -c** (checksum) detects a mismatch in the respective size or checksum for a file, it does not report an error for the file if the file has been changed by an interim fix within **/usr/emgrdata/DBS/files.db**.

Examples

1. To verify all files that comprise the **X11.fnt** package, type:

```
lppchk -c X11.fnt
```

2. To verify the symbolic links of all software products whose names begin with **X11**, type:

```
lppchk -l 'X11*'
```

3. To verify that all filesets have all required requisites and are completely installed, type:

```
lppchk -v
```

Files

Item	Description
/etc/objrepos/lpp	Specifies installation information of all software products on the root.
/usr/lib/objrepos/lpp	Specifies installation information of all software products on the **/usr** file system.

Item	Description
/usr/share/lib/objrepos/lpp	Specifies installation information of all software products on the **/usr/share** file system.
/etc/objrepos/product	Specifies installation and update information of all software products on the root.
/usr/lib/objrepos/product	Specifies installation and update information of all software products on the **/usr** file system.
/usr/share/lib/objrepos/product	Specifies installation and update information of all the software products on the **/usr/share** file system.
/etc/objrepos/inventory	Specifies names and locations of files in a software product on the root.
/usr/lib/objrepos/inventory	Specifies names and locations of files in a software product on the **/usr** file system.
/usr/share/lib/objrepos/ inventory	Specifies names and locations of files in a software product on the **/usr/share** file system.

lppmgr Command

Purpose

Manages an existing installp image source.

Syntax

lppmgr **-d** *DirectoryOrDevice* [**-r** | **-m** *MoveDirectory* | **-a** *APAR number*] { [**-x**] [**-X**] [**-l**] [**-u**] [**-b**] [**-k** *LANG*] } [**-p**] [**-t**] [**-s**] [**-V**] [**-D**]

Description

lppmgr is designed to perform the following functions on an existing installp image source (also known as an **lpp_source** in the NIM environment):

1. Remove duplicate updates (**-u** Flag).
2. Remove duplicate base levels (**-b** Flag).
3. Eliminating updates that are the same level as bases of the same file set. Such updates can create conflicts that lead to installation failure (**-u** Flag).
4. Remove message and locale file sets other than the language you specify (**-k** Flag).
5. Remove superseded file sets (**-x** Flag).
6. Remove non-system images from a NIM lpp_source resource (**-X** Flag).

By default, **lppmgr** lists all images that are filtered by the preceeding routines. The "**-r**" flag can be used to remove the filtered images and the "**-m**" flag can be used to move the images to another location.

Note: lppmgr is not intended to replace **bffcreate**, install anything, or work with installed file sets. It is also not intended to address any issues other than those mentioned earlier. Before you use the "**-X**" flag, you must have a good understanding of NIM, system images (known as SIMAGES in NIM), and the workings of a NIM **lpp_source** resource.

Flags

Item	Description
-a *APAR number*	Displays file sets associated with an APAR number or keyword. If there are more than one APAR numbers, they must be contained in quotation marks and separated by spaces.
-b	Causes **lppmgr** to filter for base level duplicates.
-D	Specifies debug mode. This flag is for debugging the **lppmgr** script. **Note:** Debug. This produces a large quantity of output and greatly reduces lppmgr performance. It is not useful for normal operations.
-d *DeviceOrDirectory*	Specifies the device or directory where the installp images reside. Currently it can be any directory, NFS mount point, or cdrom device. If the directory is not writable, you must use the **-t** flag. If the target of your operation is a NIM **lpp_source** resource, you must specify the **lpp_source** location (see the **lsnim** command). This flag is required for all operations.
-k *LANG*	Keeps only the message and locale images for the language specified by *LANG*. All other languages are filtered.
-l	Lists filtered images only. By default, **lppmgr** will only list all filtered image files unless the "**-r**" or "**-m**" flag is specified. The "**-l**" flag will override the "**-r**" or "**-m**" flag.
-m *Directory*	Moves filtered files to *Directory*. The location that is specified by *Directory* can be any writable directory path. This flag cannot be used with the "**-r**" flag.
-p	Specifies prompt mode. Prompt when moving or removing files.
-r	Removes files that have been filtered by **lppmgr**. **Note:** If the prompt flag is not specified (**-p**), **lppmgr** removes all filtered files without further user interaction. This flag cannot be used with the "**-m**" flag.
-s	Prints space usage information. This flag prints the amount of space a particular file set is using and the total amount of space in question. Some buffer space is added for file metadata.
-t	Specifies that **lppmgr** does *not* rebuild the **.toc** file. This flag can be useful for having a quick look without having to rebuild the entire **.toc** file, which can take some time. Also, this flag is required for read-only devices.
-u	Causes **lppmgr** to filter for duplicate updates and conflicting updates that are the same level as bases of the same file set.
-V	Specifies verbose mode. **lppmgr** gives more output in certain situations.
-x	Causes **lppmgr** to filter for superseded updates.
-X	Filters non-system images from a NIM **lpp_source** resource.

Exit Status

0
All **lppmgr** related operations that are completed successfully.

>0
An error occurred.

Security

Only the root user can execute **lppmgr**.

Examples

1. To list all duplicate and conflicting updates in image source directory **/myimages**, enter the following command:

```
lppmgr -d /myimages -u
```

2. To remove all duplicate and conflicting updates in image source directory **/myimages**, enter the following command:

```
lppmgr -d /myimages -u -r
```

3. To remove all duplicate and conflicting updates, duplicate base levels, and all message/locale file sets other than "en_US" in prompted mode, enter the following command:

```
lppmgr -d /myimages -purb -k en_US
```

4. To move all superseded update images and non SIMAGES from NIM **lpp_source** location **/lpps/433** to directory **/backups**, enter the following command:

```
lppmgr -d /lpps/433 -x -X -m  /backups
```

5. To list all the file sets associated with APAR numbers IX38794 and IX48523 in image source directory **/myimages**, enter the following command:

```
lppmgr -d /myimages -a "IX38794 IX48523"
```

Files

Item	Description
/usr/lib/instl/lppmgr	Contains the **lppmgr** command.

lpq Command

The **lpq** command includes information for the AIX Print Subsystem **lpq** and the System V Print Subsystem **lpq**.

AIX Print Subsystem lpq Command

Purpose

Examines the spool queue.

Syntax

lpq [**+** [*Number*]] [**-l** | **-W**] [**-P** *Printer*] [*JobNumber*] [*UserName*]

Description

The **lpq** command reports the status of the specified job or all jobs associated with the specified *UserName* and *JobNumber* variables. *JobNumber* variable specifies the number of the job in the spool queue that you want to view. A *UserName* variable specifies viewing the jobs for the name of the person who submitted the job to that queue.

The **lpq** command reports on any jobs currently in the default queue when invoked without any options. Parameters supplied that are not recognized as parameters are interpreted as user names or job numbers to filter out only those jobs of interest.

For each job submitted (each job called by the **lpr** command), the **lpq** command reports the user's name, current rank in the queue, the name of the job, the job identifier (a number that can be supplied to the **lprm** command for removing a specific job), and the total size in blocks. Normally, only as much information as will fit on one line is displayed. Job ordering depends on the algorithm used to scan the spooling directory and is supposed to be FIFO (first-in-first-out). File names making up a job may be unavailable (when the **lpr** command is used as a sink in a pipeline). In this case, the file is indicated as **-** (standard input).

The display generated by the **lpq** command contains two entries for remote queues. The first entry contains the client's local queue and local device name and its status information. The second entry follows immediately; it contains the client's local queue name (again), followed by the remote queue name. Any jobs submitted to a remote queue are displayed first on the local side and are moved to the remote device as the job is processed on the remote machine.

Since the status commands communicate with remote machines, the status display may occasionally appear to hang while waiting for a response from the remote machine. The command will eventually time out if a connection cannot be established between the two machines.

Flags

Item	Description
-l	Generates the long output format.
+ [*Number*]	Displays the spool queue until it empties. A *Number* variable is the time in seconds before the display regenerates.
-P *Printer*	Displays the spool queue for the printer specified by the *Printer* variable. **Note:** Any destination command line options override both the **LPDEST** and the **PRINTER** environment variables.
-W	Displays a wide version of status information with longer queue names, device names, and job numbers. This flag cannot be used with the -l flag. If the -l flag and the -W flag are used simultaneously, the first one specified takes precedence.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display a job number in the print queue `lp0`, enter:

   ```
   lpq  -P lp0
   ```

   This command displays a list similar to the following:

   ```
   Queue    Dev      Status    Job    Files    User      PP    %      Blks    CP    Rnk
   lp0       dlp0    running   39      motd     guest    10    83      12      1     1
   ```

2. To display the status of the default queue in wide format, enter:

```
lpq  -W
```

Files

Item	Description
/usr/bin/lpq	Contains the **lpq** command.
/usr/sbin/qdaemon	Contains the queuing daemon.
/etc/qconfig	Contains the queue configuration file.
/etc/qconfig.bin	Contains the digested, binary version of the **/etc/qconfig** file.
/var/spool/lpd/qdir/*	Contains queue requests.
/var/spool/lpd/stat/*	Contains information on the status of the devices.
/var/spool/qdaemon/*	Contains temporary copies of enqueued files.

System V Print Subsystem lpq Command

Purpose (System V)

(BSD) Displays the queue of printer jobs

Syntax (System V)

/usr/bin/lpq [**-P***printer*] [-l] [+ [*interval*]] [*job#* ...] [*username* ...]

Description (System V)

The **lpq** command displays the contents of a printer queue. It reports the status of jobs specified by *job#*, or all jobs owned by the user specified by *username*. **lpq** reports on all jobs in the default printer queue when invoked with no arguments.

For each print job in the queue, **lpq** reports the user's name, current position, the names of input files comprising the job, the job number (by which it is referred to when using **lprm**) and the total size in bytes. Normally, only as much information as will fit on one line is displayed. Jobs are normally queued on a first-in-first-out basis. Filenames comprising a job may be unavailable, such as when **lpr** is used at the end of a pipeline; in such cases the filename field indicates the standard input.

If **lpq** warns that there is no daemon present (that is, due to some malfunction), the **lpc** command can be used to restart a printer daemon.

Output formatting is sensitive to the line length of the terminal; this can result in widely-spaced columns.

Flags (System V)

-P *printer*
Display information about the queue for the specified *printer*. In the absence of the **-P** flag, the queue to the printer specified by the **PRINTER** variable in the environment is used. If the **PRINTER** variable is not set, the queue for the default printer is used.

-l
Display queue information in long format; includes the name of the host from which the job originated.

+[*interval*]
Display the spool queue periodically until it empties. This option clears the terminal screen before reporting on the queue. If an *interval* is supplied, **lpq** sleeps that number of seconds in between reports.

Security (System V)

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files (System V)

/var/spool/lp
spooling directory.

/var/spool/lp/tmp/*system_name*/*-0
request files specifying jobs

Error Codes (System V)

lpq: *printer* is printing
The **lpq** program queries the spooler **LPSCHED** about the status of the printer. If the printer is disabled, the system administrator can restart the spooler using **lpc**.

lpq: *printer* waiting for auto-retry (offline ?)
The daemon could not open the printer device. The printer may be turned off-line. This message can also occur if a printer is out of paper, the paper is jammed, and so on. Another possible cause is that a process, such as an output filter, has exclusive use of the device. The only recourse in this case is to kill the offending process and restart the printer with **lpc**.

lpq: waiting for *host* to come up
A daemon is trying to connect to the remote machine named *host*, in order to send the files in the local queue. If the remote machine is up, **lpd** on the remote machine is probably dead or hung and should be restarted using **lpc**.

lpq: sending to *host*
The files are being transferred to the remote *host*, or else the local daemon has hung while trying to transfer the files.

lpq: printer disabled reason:
The printer has been marked as being unavailable with **lpc**.

lpq: The LP print service isn't running or can't be reached.
The **lpsched** process overseeing the spooling queue does not exist. You can restart the printer daemon with **lpc**.

lpq: *printer*: unknown printer
The *printer* was not found in the System V LP database. Usually this is a typing mistake; however, it may indicate that the printer does not exist on the system. Use **lpstat -p** to find the reason.

lpq: error on opening queue to spooler
The connection to **lpsched** on the local machine failed. This usually means the printer server started at boot time has died or is hung. Check if the printer spooler daemon */usr/lib/lp/lpsched* is running.

lpq: Can't send message to LP print service

lpq: Can't establish contact with LP print service
These indicate that the LP print service has been stopped. Get help from the system administrator.

lpq: Received unexpected message from LP print service
It is likely there is an error in this software. Get help from system administrator.

lpr Command

The **lpr** command includes information for the AIX Print Subsystem **lpr** and the System V Print Subsystem **lpr**.

AIX Print Subsystem lpr Command

Purpose

Enqueues print jobs.

Syntax

lpr [**-f**] [**-g**] [**-h**] [**-j**] [**-l**] [**-m**] [**-n**] [**-p**] [**-r**] [**-s**] [**-P** *Printer*] [**-#** *NumberCopies*] [**-C** *Class*] [**-J** *Job*] [**-T** *Title*] [**-i** [*NumberColumns*]] [**-w** *Width*] [*File ...*]

Description

The **lpr** command uses a spooling daemon to print the named *File* parameter when facilities become available. If no files are specified, the **lpr** command reads from standard input.

Flags

Item	Description
-# *Number*	Produces multiple copies of output, using the *Number* variable as the number of copies for each file named.
-C *Class*	Specifies the print *Class* as the job classification on the burst page.
-f	Uses a filter that interprets the first character of each line as a standard FORTRAN carriage control character.
-g	The files are assumed to contain standard plot data.
-h	Suppresses printing of the burst page. **Note:** The default is to print a header page and not a trailer page.
-i [*Number*]	Indents output *Number* spaces. If the *Number* variable is not given, eight spaces are used as the default.
-j	Specifies that the message `Job number is: nnn`, where nnn is the assigned job number, be displayed to standard output. This occurs only if the job is submitted to a local print queue.
-J *Job*	Prints the *Job* variable as the job name on the burst page. Usually, the **lpr** command uses the name of the first file.
-l	(Lowercase L) Uses a filter that allows control characters to be printed.
-m	Sends mail upon completion of spooling.
-n	Uses a filter that formats files containing *ditroff* (device-independent *troff*) data.
-P *Printer*	Forces output to the *Printer* variable. If this flag is not specified, the following conditions occur: • If a default exists, the **lpr** command uses the default printer. • If the **LPDEST** environment variable is set, then **lpr** uses the value specified by the **LPDEST** variable. If set, this value is always used, even if the **PRINTER** variable is also set. • If the **PRINTER** variable is set and no **LPDEST** variable is set, then **lpr** uses the value specified by the **PRINTER** environment variable. **Note:** Any destination command line options override both the **LPDEST** and the **PRINTER** environment variables.
-p	Uses the **pr** command to format the file (`lpr -p` is very much like `pr | lpr`).
-r	Removes the file upon completion of spooling.

Item	Description
-s	Prints from the files specified on the command line rather than trying to copy them (so large files can be printed). This means the data files should not be modified or removed until they have been printed. Note that this flag only works on the local host (files sent to remote printer hosts are copied anyway), and only with named data files. It does not work if the **lpr** command is at the end of a pipeline.
-T *Title*	Uses the *Title* variable instead of the file name for the title used by the **pr** command.
-w *Number*	Uses the *Number* variable as the page width for the **pr** command.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To print three copies of the files `new.index.c, print.index.c,` and `more.c,` enter:

   ```
   lpr  -#3 new.index.c print.index.c more.c
   ```

   Prints three copies of the `new.index.c` file, three copies of the `print.index.c file,` and three copies of the `more.c file`.
2. To print three copies of the concatenation of three files `new.index.c, print.index.c,` and `more.c`, enter:

   ```
   cat new.index.c print.index.c more.c | lpr  -#3
   ```
3. To print `Operations` on the burst page, followed by file `new.index.c`, enter:

   ```
   lpr  -C Operations new.index.c
   ```

   This replaces the system name (the name returned by host name) with `Operations` on the burst page.
4. To queue the MyFile file and return the job number, enter:

   ```
   lpr -j MyFile
   ```

Files

Item	Description
/usr/sbin/qdaemon	Queuing daemon.
/etc/qconfig	Queue configuration file.
/etc/qconfig.bin	Digested, binary version of the **/etc/qconfig** file.
/var/spool/lpd/qdir/*	Queue requests.
/var/spool/lpd/stat/*	Information on the status of the queues.
/var/spool/qdaemon	Temporary copies of enqueued files.

System V Print Subsystem lpr Command

Purpose (System V)

(BSD) Sends a job to the printer.

Syntax (System V)

/usr/bin/lpr [**-P** *printer*] [**-#** *copies*] [**-C** *class*] [**-J** *job*] [**-T** *title*] [**-i** [*indent*]] [**-w** *cols*] [**-r**] [**-m**] [**-h**] [**-s**] [**-***filter_option*] [*file* ...]

Description (System V)

The **lpr** command forwards printer jobs to a spooling area for subsequent printing as facilities become available. Each printer job consists of copies of each *file* you specify. The spool area is managed by the line printer spooler, **lpsched**. **lpr** reads from the standard input if no files are specified.

lp is the preferred interface.

Command-line options cannot be combined into a single argument as with some other commands. The command:

```
lpr -fs
```

is not equivalent to

```
lpr -f -s
```

Placing the **-s** flag first, or writing each option as a separate argument, makes a link as expected.

lpr -p is not precisely equivalent to **pr | lpr**. **lpr -p** puts the current date at the top of each page, rather than the date last modified.

Fonts for **troff** and T[E]X reside on the printer host. It is not possible to use local font libraries.

lpr objects to printing binary files.

If userA uses **su** to become userB and uses */usr/bin/lpr*, then the printer request will be entered as userB, not userA

Flags (System V)

-P *printer*
Send output to the named *printer*. Otherwise send output to the printer named in the **PRINTER** environment variable, or to the default printer, **lp**.

-# *copies*
Produce the number of *copies* indicated for each named file. For example:

```
lpr -#3 index.c lookup.c
```

produces three copies of *index.c*, followed by three copies of *lookup.c*. On the other hand,

```
cat index.c lookup.c | lpr -#3
```

generates three copies of the concatenation of the files.

-C *class*
Print *class* as the job classification on the burst page. For example,

```
lpr -C Operations new.index.c
```

replaces the system name (the name returned by ``hostname'') with **Operations** on the burst page, and prints the file *new.index.c*.

-J *job*
Print *job* as the job name on the burst page. Usually, **lpr** uses the first file's name.

-T *title*
Use *title* instead of the file name for the title used by **pr**.

-i[*indent*]
Indent output *indent* <Space> characters. Eight <Space> characters is the default.

-w *cols*
Use *cols* as the page width for **pr**.

-r
Remove the file upon completion of spooling, or upon completion of printing with the **-s** flag.

-m
Send mail upon completion.

-h
Suppress printing the burst page.

-s
Use the full pathnames (not symbolic links) of the files to be printed rather than trying to copy them. This means the data files should not be modified or removed until they have been printed. This flag only prevents copies of local files from being made. Jobs from remote hosts are copied anyway. The **-s** flag only works with named data files; if the **lpr** command is at the end of a pipeline, the data is copied to the spool.

filter_option
The following single letter options notify the line printer spooler that the files are not standard text files. The spooling daemon will use the appropriate filters to print the data accordingly.

-p
Use **pr** to format the files (**lpr -p** is very much like **pr | lpr**).

-l
Print control characters and suppress page breaks.

-t
The files contain **troff** (cat phototypesetter) binary data.

-n
The files contain data from *ditroff* (device independent **troff**).

-d
The files contain data from *tex* (DVI format from Stanford).

-g
The files contain standard plot data as produced by the routine **plot** for the filters used by the printer spooler.

-v
The files contain a raster image. The printer must support an appropriate imaging model such as PostScript in order to print the image.

-c
The files contain data produced by *cifplot*.

-f
Interpret the first character of each line as a standard FORTRAN carriage control character.

If no *filter_option* is given (and the printer can interpret PostScript), the string `**%!**' as the first two characters of a file indicates that it contains PostScript commands.

These filter options offer a standard user interface, and all options may not be available for, nor applicable to, all printers.

Security (System V)

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files (System V)

/usr/lib/lp/lpsched
System V line printer spooler

/var/spool/lp/tmp/*
directories used for spooling

/var/spool/lp/tmp/*system/*-0*
spooler control files

/var/spool/lp/tmp/*system/*-N*
(*N* is an integer and > 0) data files specified in ` *-0' files

Error Codes (System V)

lpr: *printer*: unknown printer
The *printer* was not found in the LP database. Usually this is a typing mistake; however, it may indicate that the printer does not exist on the system. Use **lpstat -p** to find the reason.

lpr: error on opening queue to spooler
The connection to **lpsched** on the local machine failed. This usually means the printer server started at boot time has died or is hung. Check if the printer spooler daemon **/usr/lib/lpsched** is running.

lpr: *printer*: printer queue is disabled
This means the queue was turned off with

```
/usr/etc/lpc disable printer
```

to prevent **lpr** from putting files in the queue. This is usually done when a printer is going to be down for a long time. The printer can be turned back on by a privileged user with **lpc**.

lpr: Can't send message to the LP print service

lpr: Can't establish contact with the LP print service
These indicate that the LP print service has been stopped. Get help from the system administrator.

lpr: Received unexpected message from LP print service
It is likely there is an error in this software. Get help from system administrator.

lpr: There is no filter to convert the file content
Use the **lpstat -p -l** command to find a printer that can handle the file type directly, or consult with your system administrator.

lpr: cannot access the file
Make sure file names are valid.

lprm Command

The **lprm** command includes information for the AIX Print Subsystem **lprm** and the System V Print Subsystem **lprm**.

AIX Print Subsystem lprm Command

Purpose

Removes jobs from the line printer spooling queue.

Syntax

lprm [**-P** *Printer*] [*JobNumber*] [*UserName ...*] [**-**]

Description

The **lprm** command removes one or more jobs from the spool queue of a printer.

You cannot run the **lprm** command without specifying a job number, the **-** (minus sign) flag, or at least one user name.

Specifying a *UserName* parameter, or list of names, causes the **lprm** command to attempt to remove any jobs queued belonging to that user (or users).

You can remove an individual job from a queue by specifying its *JobNumber*. This job number is obtained by using the **lpq** command.

Flags

Item	Description
-	Removes all jobs a user owns. Someone with root user authority can use this flag to remove all jobs from a queue. This flag is not valid for remote print.
-P *Printer*	Specifies the queue associated with a specific *Printer* variable. If this flag is not specified, the following conditions occur: • If the **LPDEST** environment variable is set, then **lprm** uses the value specified by the **LPDEST** variable. If set, this value is always used, even if the **PRINTER** variable is also set. • If the **PRINTER** variable is set and no **LPDEST** variable is set, then **lprm** uses the value specified by the **PRINTER** environment variable. If neither the **LPDEST** nor the **PRINTER** variable is set, the **lprm** command removes jobs from the default queue. **Note:** Any destination command line options override both the **LPDEST** and the **PRINTER** environment variables.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To remove job number 13 from the default printer queue, enter:

```
lprm 13
```

2. To remove job number 13 from printer queue `lp0`, enter:

```
lprm  -P lp0 13
```

3. To remove a job from the printer queue for a certain user, enter:

```
lprm guest
```

Files

Item	Description
/usr/bin/lprm	Contains the **lprm** command.
/etc/qconfig	Contains the configuration file.

System V Print Subsystem lprm Command

Purpose (System V)

(BSD) Remove jobs from the printer queue

Syntax (System V)

/usr/bin/lprm [**-P***printer*] [-] [*job #* ...] [*username*...]

Description (System V)

The **lprm** command removes a job or jobs from a printer's spooling queue. Since the spool directory is protected from users, using **lprm** is normally the only method by which a user can remove a job.

Without any arguments, **lprm** deletes the job that is currently active, provided that the user who invoked **lprm** owns that job.

When the privileged user specifies a *username*, **lprm** removes all jobs belonging to that user.

You can remove a specific job by supplying its job number as an argument, which you can obtain using **lpq**. For example:

```
lpq  -Phost
host is ready and printing
Rank         Owner   Job     Files           Total Size
active       wendy   385     standard input  35501 bytes
lprm -Phost 385
```

lprm reports the names of any files it removes, and is silent if there are no applicable jobs to remove.

lprm Sends the request to cancel a job to the print spooler, **LPSCHED**.

An active job may be incorrectly identified for removal by an **lprm** command issued with no arguments. During the interval between an **lpq** command and the execution of **lprm**, the next job in queue may have become active; that job may be removed unintentionally if it is owned by you. To avoid this, supply **lprm** with the job number to remove when a critical job that you own is next in line.

Only the privileged user can remove print jobs submitted from another host.

Flags (System V)

-P*printer*
: Specify the queue associated with a specific printer. Otherwise the value of the **PRINTER** variable in the environment is used. If this variable is unset, the queue for the default printer is used.

-
: Remove all jobs owned by you. If invoked by the privileged user, all jobs in the spool are removed. Job ownership is determined by the user's login name and host name on the machine where the **lpr** command was executed.

Security (System V)

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files (System V)

/var/spool/lp/*
spooling directories

Error Codes (System V)

lprm: *printer*: unknown printer
The *printer* was not found in the System V LP database. Usually this is a typing mistake; however, it may indicate that the printer does not exist on the system. Use **lpstat -p** to get the status of printers.

lprm: error on opening queue to spooler
The connection to **lpsched** on the local machine failed. This usually means the printer server started at boot time has died or is hung. Check if the printer spooler daemon **/usr/lib/lp/lpsched** is running.

lprm: Can't send message to the LP print service

lprm: Can't receive message from the LP print service
These indicate that the LP print service has been stopped. Get help from the system administrator.

lprm: Received unexpected message from the LP print service
It is likely there is an error in this software. Get help from system administrator.

lprm: Can't cancel request
You are not allowed to remove another user's print request.

lpsched Command

Note: This is a System V Print Subsystem command.

Purpose

Starts/stops the print service.

Syntax

/usr/lib/lp/lpsched

lpshut

Description

The **lpsched** command starts the LP print service.

The **lpshut** command shuts down the print service. All printers that are printing at the time the **lpshut** command is invoked stop printing. When **lpsched** is started again, requests that were printing at the time a printer was shut down are reprinted from the beginning.

You must have the appropriate privilege to run these commands.

If the scheduler fails to run, check the **lpsched** log file, which contains all failed attempts to load print requests, printer descriptions, forms, filters, classes, alerts, and systems. The log files are located in **/var/lp/logs**. Useful information on the networked print service can also be found in the **/var/lp/logs/lpNet** log file.

If you enter `lpsched -?`, the system displays the command usage message and returns 0.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/var/spool/lp/*	

lpstat Command

The **lpstat** command includes information for the AIX Print Subsystem **lpstat** and the System V Print Subsystem **lpstat**.

AIX Print Subsystem lpstat Command

Purpose

Displays line printer status information.

Syntax

lpstat [**-a***List*] [**-c***List*] [**-d**] [**-o***List*] [**-p***List*] [**-r**] [**-s**] [**-t**] [**-u***List*] [**-v***List*] [**-W**]

Description

The **lpstat** command displays information about the current status of the line printer.

If no flags are given, **lpstat** prints the status of all requests made by the **lp** command.

Flags can appear in any order and can be repeated. Some flags take an optional list as a parameter. Enter the list as either a list of items separated by commas, as in `lpstat -aQueue1,Queue2`, or as a list of items enclosed in single or double quotes and separated either by commas or one or more spaces, as in, for example, `lpstat -a"Queue1 Queue2"` or `lpstat -a"Queue1,Queue2"` or `lpstat -a'Queue1 Queue2'` or `lpstat -a'Queue1,Queue2'`.

If you specify a flag with no parameters, all information pertaining to that flag is printed.

The display generated by the **lpstat** command contains two entries for remote queues. The first entry contains the client's local queue and local device name and its status information. The second entry contains the client's local queue name followed by the remote queue name. The spooling subsystem first displays remote print requests on the local queue. When the remote machine begins to process the remote print job, the status display for the print job moves to the remote queue.

When a status command communicates with a remote host, the display occasionally appears to hang while the command waits for a response from the remote machine. The command eventually times out if no connection is established between the two machines.

Flags

Item	Description
-a*List*	Provides status and job information on queues. Specifying the **lpstat** command with this flag is the same as specifying the **enq -q -P***Queue1* **-P***Queue2* ... command (where *Queue1*, *Queue2*, etc., are items in *List*).
-c*List*	Provides status and job information on queues. Specifying the **lpstat** command with this flag is the same as specifying the **enq -q -P***Queue1* **-P***Queue2* ... command (where *Queue1*, *Queue2*, etc., are items in *List*).
-d	Prints the status information for the system default destination for the **lp** command. Specifying the **lpstat** command with this flag is the same as specifying the **enq -q** command.

Item	Description
-o*List*	Prints the status of print requests or print queues. *List* is a list of intermixed printer names and job numbers.
-p*List*	Prints the status of printers. **Note:** You cannot use both the **-p** flag and the **-t** flag at the same time.
-r	Provides status and job information on queues. Specifying the **lpstat** command with this flag is the same as specifying the **enq -A** command.
-s	Displays a status summary, including a list of printers and their associated devices. Specifying the **lpstat** command with this flag is the same as specifying the **enq -A** command.
-t	Displays all status information, including a list of printers and their associated devices. Specifying the **lpstat** command with this flag is the same as specifying the **enq -AL** command.
-u*List*	Prints the status of all print requests for users specified in *List*. *List* is a list of login names. Specifying the **lpstat** command with this flag is the same as specifying the **enq -u** *UserName* command.
-v*List*	Prints the status of printers. The *List* variable is a list of printer names.
-W	Displays a wide version of the status information with longer queue names, device names, and job numbers. This flag cannot be used with the **-t** flag. If the **-t** flag and the **-W** flag are used simultaneously, the first specified flag takes precedence. If the **-W** flag and **-l** flag are used simultaneously, the result displays the long status of the print job in the semicolon-separated format.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the status for all print queues, enter:

   ```
   lpstat
   ```

2. To display the long status for all printers, enter:

   ```
   lpstat -t
   ```

3. To display a job number in the print queue `lp0`, enter:

   ```
   lpstat -plp0
   ```

   This command displays a list similar to the following:

```
Queue  Dev      Status      Job    Files      User      PP      %      Blks      CP      Rnk
lp0    dlp0     running     39     motd       guest     10      83      12        1       1
```

4. To display the status for users `root`, `ghandi`, and `king`, enter:

   ```
   lpstat  -u"root,ghandi,king"
   ```

5. To display the status of all print queues in wide format, enter:

   ```
   lpstat  -W
   ```

Files

Item	Description
/var/spool/lpd/*	Contains temporary copies of remote enqueued files.

System V Print Subsystem lpstat Command

Purpose (System V)

Prints information about the status of the LP print service.

Syntax (System V)

lpstat [*flags*] [*request-ID-list*]

Description (System V)

The **lpstat** command displays information about the current status of the LP print service. If no *flags* are given, **lpstat** displays the status of all print requests made by you.

The command **lpstat -o** *printername* is used to list all the requests queued on the specified printer. If *printername* points to a remote printer, then **lpstat -o** *printername* lists all the requests on the remote printer, not just those submitted locally.

Any arguments that are not *flags* are assumed to be *request-IDs* as returned by **lp**. The **lpstat** command displays the status of such requests. The *flags* may appear in any order and may be repeated and intermixed with other arguments. Some of the keyletters below may be followed by an optional *list* that can be in one of two forms:

- a list of items separated by commas, for example, **-p** *printer1*,*printer2*
- a list of items separated by spaces and enclosed in quotes, for example, **-u** "*user1 user2 user3*"

Specifying **all** after any keyletter that takes *list* as an argument causes all information relevant to the keyletter to be displayed. For example, the command **lpstat -a all** lists the accepting status of all print destinations.

The omission of a *list* following such keyletters causes all information relevant to the keyletter to be displayed. For example, the command **lpstat -a** is equivalent to **lpstat -a all**.

There are two exceptions to the behavior of the **all** keyword. The first is when it is used in conjunction with the **-o** flag: **lpstat -o all** only lists requests submitted locally to remote printers. The second is when it is used with directory-enabled print queues. Use of the **all** keyword will only return non-directory-enabled print queues. **lpstat -a** *list* will report whether the both directory-enabled and non-directory-enabled print queues in *list* are accepting requests. For the **-a** and **-b** flags, **lpsched** will remember the directory-enabled print queues specified until it is restarted. Subsequent calls to **lpstat -a** and **lpstat -p** will report the status of all non-directory-enabled print queues as well as the directory-enabled print queues previously specified. Once **lpsched** has been restarted, the use of the **all** keyword with the **lpstat** command will once again only display non-directory-enabled print queues. The **dslpsearch** command should be used to search for defined directory-enabled print queues.

If you enter `lpstat -?`, the system displays the command usage message and returns 0.

Flags (System V)

-a [*list*]
Report whether print destinations are accepting requests. *list* is a list of intermixed printer names and class names.

-c [*list*]
Report names of all classes and their members. *list* is a list of class names.

-d
Report what the system default destination is (if any).

-f [*list*] [-l]
Verify that the forms in *list* are recognized by the LP print service. *list* is a list of forms; the default is **all**. The **-l** option will list the form parameters.

-o [*list*] [-l]
Report the status of print requests. *list* is a list of intermixed printer names, class names, and *request-IDs*. The keyletter **-o** may be omitted. The **-l** option lists for each request whether it is queued for, assigned to, or being printed on a local printer, the form required (if any), and the character set or print wheel required (if any). Note that required forms (if any) are not listed for remote printers.

-p [*list*] [-D] [-l]
If the **-D** flag is given, a brief description is printed for each printer in *list*. If the **-l** flag is given, a full description of each printer's configuration is given, including the form mounted, the acceptable content and printer types, a printer description, the interface used, and so on.

In order to maintain system security access information, the information needed to produce the printer status given by **lpstat -p** is available only if the LP scheduler is running.

-r
Report the status of the LP request scheduler (whether it is running).

-R
Report a number showing the rank order of jobs in the print queue for each printer.

-s [-l]
Display a status summary, including the status of the LP scheduler, the system default destination, a list of class names and their members, a list of printers and their associated devices, a list of the systems sharing print services, a list of all forms and their availability, and a list of all recognized character sets and print wheels. The **-l** flag displays all parameters for each form and the printer name where each character set or print wheel is available.

-S [*list*] [-l]
Verify that the character sets or the print wheels specified in *list* are recognized by the LP print service. Items in *list* can be character sets or print wheels; the default for *list* is **all**. If the **-l** flag is given, each line is appended by a list of printers that can handle the print wheel or character set. The list also shows whether the print wheel or character set is mounted or specifies the built-in character set into which it maps.

-t [-l]
Display all status information: all the information obtained with the **-s** flag, plus the acceptance and idle/busy status of all printers and status of all requests. The **-l** flag displays more detail as described for the **-f**, **-o**, **-p**, and **-s** flag.

-u [*login-ID-list*]
Display the status of output requests for users. The *login-ID-list* argument may include any or all of the following constructs:

login-ID
a user on the local system

system-name*!*login-ID
a user on system *system-name*

***system-name*!all**
all users on system *system-name*

all!*login-ID*
a user on all systems

all
all users on the local system

all!all
all users on all systems

The default value of *login-ID-list* is **all**.

Security (System V)

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

lpsystem Command

Purpose

Registers remote systems with the print service.

Note: This is a System V Print Subsystem command.

Syntax

lpsystem [*-t Type*] [**-T** *Timeout*] [**-R** *Retry*] [**-y** *Comment*] *SystemName* [*SystemName* . . .]

lpsystem -l [*SystemName* . . .]

lpsystem -r *SystemName* [*SystemName* . . .]

lpsystem -A

Description

The **lpsystem** command defines parameters for the LP print service, with respect to communication (via a high-speed network such as TCP/IP) with remote systems.

Specifically, the **lpsystem** command defines remote systems with which the local LP print service can exchange print requests. These remote systems are described to the local LP print service in terms of several parameters that control communication: type, retry, and timeout. These parameters are defined in **/etc/lp/Systems**. You can edit this file with a text editor (such as **vi**), but editing is not recommended. By using **lpsystem**, you can ensure that **lpsched** is notified of any changes to the *Systems* file.

The *Timeout* parameter specifies the length of time (in minutes) that the print service should allow a network connection to be idle. If the connection to the remote system is idle (that is, there is no network traffic) for *N* minutes, then drop the connection. (When there is more work, the connection is re-established.) Legal values are **n**, **0**, and *N*, where *N* is an integer greater than 0. If a decimal number is used for *N*, it is truncated to the whole number. The value **n** means never time out; **0** means as soon as the connection is idle, drop it. The default is **n**.

The *Retry* parameter specifies the length of time (in minutes) to wait before trying to re-establish a connection to the remote system, when the connection was dropped abnormally (that is, a network error). Legal values are **n**, **0**, and *N*, where *N* is an integer greater than 0. It means wait *N* minutes before trying to reconnect. If a decimal number is used for *N*, it is truncated to the whole number. (The default is 10

minutes.) The value **n** means do not retry dropped connections until there is more work; **0** means try to reconnect immediately.

The *Comment* parameter allows you to associate a free form comment with the system entry. This is visible when **lpsystem -l** is used.

The *SystemName* is the name of the remote system from which you want to be able to receive jobs and to which you want to be able to send jobs. A special entry is provided with the **/etc/lp/Systems** file by default, which allows all connections to **bsd** systems. That entry uses the asterisk (*) as the *SystemName*.

The command **lpsystem -l** [*SystemName*] prints out a description of the parameters associated with *SystemName* (if a system has been specified) or with all the systems in its database (if *SystemName* has not been specified).

The command **lpsystem -r** *SystemName* removes the entry associated with *SystemName*. The print service no longer accepts jobs from that system or send jobs to it, even if the remote printer is still defined on the local system. The scheduler must be running when the removal of a systems file entry occurs, because the scheduler checks whether the system entry is currently used by a printer destination. If currently used, the system entry cannot be removed.

If you use **lpsystem -r** *SystemName* to remove a system and you have active printers for that system, you will not be allowed to remove the system from the system file. **lpsystem -r** *SystemName* only works if no printers for that system exist.

With respect to the semantics of the *Timeout* and *Retry* values, the print service uses one process for each remote system with which it communicates, and it communicates with a remote system only when there is work to be done on that system or work is being sent from that system.

The system initiating the connection is the master process, and the system accepting the connection is the secondary process. This designation serves only to determine which process dies (the secondary) when a connection is dropped. This helps prevent more than one process communicating with a remote system. All connections are bi-directional, regardless of the master/secondary designation. You cannot control a system's master/secondary designation. Typically, a client machine has the master child, and the server machine has the secondary child. If a master process times out, then both the secondary and master exit. If a secondary process times out, then it is possible that the master may still live and retry the connection after the retry interval. Therefore, one system's resource management strategy can affect another system's strategy.

All forms of the **lpsystem** command accept ***** (asterisk enclosed in double quotes) for *SystemName*.

Depending upon the configuration of the name server, you may need to change the entry in the *SystemName* field in **/etc/lp/Systems** to a full domain name.

If you enter `lpsystem -?`, the system displays the command usage message and returns 0.

Flags

Item	Description
-A	Prints out the TCP/IP address in a format.
-l [*SystemName*]	Prints out a description of the parameters associated with *SystemName* or with all the systems in its database.
-r *SystemName*	Removes the entry associated with *SystemName*.
-R *Retry*	Specifies time to wait before trying to reestablish a connection for a remote system.
-T *Timeout*	Specifies the time allowed for a network connection to be idle. *Timeout* is in minutes. Default is to never time out.

Item	Description
-y *Comment*	Allows you to associate a free-form comment with the system entry.
-t *Type*	Specifies the type of remote system. The only supported value for the -t flag is bsd.

Security

Only a user with appropriate privileges may execute the **lpsystem** command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Table 15. Files

Item	Description
/etc/lp/*	**/var/spool/lp/***

lptest Command

Purpose

Generates the line printer ripple pattern.

Syntax

lptest [;*Length Count*]

Description

The **lptest** command writes the traditional "ripple" test pattern on a standard output device such as a terminal or a printer. In 96 lines, this pattern will print all 96 printable ASCII characters in each position. While originally created to test printers, the ripple pattern is quite useful for testing terminals, driving terminal ports for debug purposes, or any other task where a quick supply of random data is needed.

Using the **lptest** command, you can specify the output line length if the default length of 79 is not appropriate. You can also specify the number of output lines to be generated if the default *Count* parameter of 200 is not appropriate. Note that if *Count* parameter is specified, *Length* must also be specified.

Examples

To display or print 100 lines of 80-column test output to standard output, enter:

```
lptest 80 100
```

lpusers Command

Note: This is a System V Print Subsystem command.

Purpose

Set printing queue priorities.

Syntax

lpusers -d *PriorityLevel*

lpusers -q *PriorityLimit* **-u** *LoginIDList*

lpusers -u *LoginIDList*

lpusers -q *PriorityLimit*

lpusers -l

Description

The **lpusers** command sets limits to the queue priority level that can be assigned to jobs submitted by users of the LP print service.

The first form of the command (with **-d**) sets the system-wide priority default to *PriorityLevel*, where *PriorityLevel* is a value of 0 to 39, with 0 being the highest priority. If a user does not specify a priority level with a print request, the default priority is used. Initially, the default priority level is 20.

The second form of the command (with **-q** and **-u**) sets the default *PriorityLimit* (from 0 to 39) that the users in the *LoginIDList* can request when submitting a print request. The *LoginIDList* parameter may include any or all of the following constructs:

Users that have been given a limit cannot submit a print request with a higher priority level than the one assigned, nor can they change a request already submitted to have a higher priority. Any print requests submitted with priority levels higher than allowed will be given the highest priority allowed.

The third form of the command (with **-u**) removes any explicit priority limit for the specified users.

The fourth form of the command (with **-q**) sets the default priority limit for all users not explicitly covered by the use of the second form of this command.

The last form of the command (with **-l**) lists the default priority level and the priority limits assigned to users.

If you enter `lpusers -?`, the system displays the command usage message and returns 0.

Parameters

Item	Description
LoginID	Specifies a a user on the local system.
*system_name***!***login-ID*	User on the system *system_name*
*system_name***!all**	Users on system *system_name*
all!*login-ID*	User on all systems
all	Users on the local system

Flags

Item	Description
-d *PriorityLevel*	Sets the system-wide priority default to *PriorityLevel*.
-l	Lists the default priority level and the priority limits assigned to users.

Item	Description
-q *PriorityLimit*	Sets the default highest priority level for all users not explicitly covered.
-q *PriorityLimit* **-u** *LoginIDList*	Sets the default highest priority level users in *LoginIDList* can request when submitting a print request.
-u *LoginIDList*	Removes any explicit priority level for the specified users.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

ls Command

Purpose

Displays the contents of a directory.

Syntax

To Display Contents of Directory or Name of File

ls [**-1**] [**-A**] [**-C**] [**-E**] [**-F**] [**-H** | **-L**] [**-N**] [**-R**] [**-S**] [**-X**] [**-a**] [**-b**] [**-c**] [**-d**] [**-e**] [**-f**] [**-g**] [**-i**] [**-k**] [**-l**] [**-m**] [**-n**] [**-o**] [**-p**] [**-q**] [**-r**] [**-s**] [**-t**] [**-u**] [-U] [**-x**] [*File* ...]

To Display Contents of Directory

ls -f [**-C**] [**-d**] [**-i**] [**-m**] [**-s**] [**-X**] [**-x**] [**-1**] [-U] [*Directory* ...]

Description

The **ls** command writes to standard output the contents of each specified *Directory* parameter or the name of each specified *File* parameter, along with any other information you ask for with the flags. If you do not specify a *File* or *Directory* parameter, the **ls** command displays the contents of the current directory.

Specifying more than one of the options in the mutually exclusive pairs is not considered an error. The last option specified in each pair determines the output format.

By default, the **ls** command displays all information in alphabetic order by file name. The collating sequence is determined by the **LANG** or **LC_COLLATE** environment variable.

When the **ls** command displays the contents of a directory, it does not show entries for files whose names begin with a **.** (dot) unless you use the **-a** or **-A** flag. If the command is executed by root, it uses the **-A** flag by default.

There are three main ways to format the output:

- List one entry per line.
- List entries in multiple columns by specifying either the **-C** or **-x** flag. The **-C** flag is the default format when output is to a TTY. The **ls** command displays single column output if file or directory names are too long.
- List entries in a comma-separated series by specifying the **-m** flag.

To determine the number of character positions in the output line, the **ls** command uses the **COLUMNS** environment variable. If this variable is not set, the command gets the current column value of the display. If the **ls** command cannot determine the number of character positions by either of these methods, it uses a default value of 80.

The mode displayed with the `-U` flag is the same as with the `-l` flag, except for the addition of an 11th character interpreted as follows:

Item	Description
E	Indicates a file has extended attributes (EA) information. The EA of a file is displayed by using the `getea` command.
-	Indicates a file does not have extended attributes information.
e	Indicates a file is encrypted.

Encryption takes precedence over the presence of the Access Control Lists (ACLs) and other EAs.

The mode displayed with the **-e** and **-l** flags is interpreted as follows:

If the first character is:

Item	Description
d	The entry is a directory.
b	The entry is a block special file.
c	The entry is a character special file.
l	The entry is a symbolic link, and either the **-N** flag was specified or the symbolic link did not point to an existing file.
p	The entry is a first-in, first-out (FIFO) special file.
s	The entry is a local socket.
-	The entry is an ordinary file.

The next nine characters are divided into three sets of three characters each. The first set of three characters show the owner's permission. The next set of three characters show the permission of the other users in the group. The last set of three characters shows the permission of anyone else with access to the file. The three characters in each set indicate, respectively, read, write, and execute permission of the file. Execute permission of a directory lets you search a directory for a specified file.

Permissions are indicated as follows:

Item	Description
r	Read
w	Write (edit)
x	Execute (search)
-	Corresponding permission not granted

The group-execute permission character is **s** if the file has set-group-ID mode. The user-execute permission character is **s** if the file has set-user-ID mode. The last character of the mode (usually **x** or **-**) is **T** if the 01000 (octal) bit of the mode is set (see the **chmod** command for the meaning of this mode). The indications of set-ID and 01000 bit of the mode are capitalized (**S** and **T**, respectively) if the corresponding execute permission is not set. The mode **t** indicates that the sticky bit is on for the file or the directory.

The mode displayed with the **-e** flag is the same as with the **-l** flag, except for the addition of an 11th character interpreted as follows:

Item	Description
+	Indicates a file has extended security information. For example, the file may have extended **ACL**, **TCB**, or **TP** attributes in the mode. The access control information (**ACL**) of a file is displayed by using the **aclget** command. The value of the **TCB** and **TP** attributes are displayed by using the **chtcb** command.
-	Indicates a file does not have extended security information.

When the size of the files in a directory are listed, the **ls** command displays a total count of blocks, including indirect blocks.

Flags

Item	Description
-A	Lists all entries except **.** (dot) and **..** (dot-dot).
-a	Lists all entries in the directory, including the entries that begin with a **.** (dot).
-b	Displays nonprintable characters in an octal (\n*nn*) notation.
-c	Uses the time of last modification of the i-node for either sorting (when used with the **-t** flag) or for displaying (when used with the **-l** flag). This flag has no effect if it is not used with either the **-t** or **-l** flag, or both.
-C	Sorts output vertically in a multicolumn format. This is the default method when output is to a terminal.
-d	Displays only the information for the directory named. Directories are treated like files, which is helpful when using the **-l** flag to get the status of a directory.
-e	Displays the mode (including security information), number of links, owner, group, size (in bytes), time of last modification, and name of each file. If the file is a special file, the size field contains the major and minor device numbers. If the file is a symbolic link, the path name of the linked-to file is printed preceded by a -> (minus, greater than) sign. The attributes of the symbolic link are displayed.
-E	Lists extent attributes of a file that exists in Vxfs file systems, such as space reservation, fixed extent size, and extent allocation flag information. You must specify the **-l** flag with this flag; otherwise, the **ls** command ignores the **-E** flag and completes the execution.
-f	Lists the name in each slot for each directory specified in the *Directory* parameter. This flag turns off the **-l**, **-t**, **-s**, and **-r** flags, and turns on the **-a** flag. The order of the listing is the order in which entries appear in the directory.
-F	Puts a / (slash) after each file name if the file is a directory, an * (asterisk) if the file can be executed, an = (equal sign) if the file is a socket, a | (pipe) sign if the file is a FIFO, and an @ for a symbolic link. Symbolic links that are named as operands are not followed unless you have specified the **-H** or **-L** flag.
-g	Displays the same information as the **-l** flag, except the **-g** flag suppresses display of the owner and symbolic link information.
-H	If a symbolic link referencing a file of type directory is specified on the command line, the **ls** command shall evaluate the file information and file type to be those of the file referenced by the link, and not the link itself; however, the **ls** command shall write the name of the link itself and not the file referenced by the link.

Item	Description
-i	Displays the i-node number in the first column of the report for each file. If the file system has an internal snapshot, the **.snapshot** directory and all its contents do not have unique i-node numbers.
-k	Sets the block size for **-s** option and the per-directory block count written for **-l**, **-n**, **-g**, and **-o** options to 1024 bytes.
-L	Lists the file or directory contents that the link references. This is the default action. Symbolic links are followed. If the **-l** option is used, the **-N** option becomes the default, and no symbolic links are followed. When the **-l** option is used, only the **-L** option can override the **-N** default.
-l	(Lower case L) Displays the mode, number of links, owner, group, size (in bytes), and time of last modification for each file. If the file is a special file, the size field contains the major and minor device numbers. If the time of last modification is greater than six months ago, the time field is shown in the format **month date year** where as files modified within six months the time field is shown as **month date time** format. If the file is a symbolic link, the path name of the linked-to file is printed preceded by a ->. The attributes of the symbolic link are displayed. The **-n**, **-g**, and **-o** flag overrides the **-l** flag. **Notes:** 1. A symbolically linked file is followed by an arrow and the contents of the symbolic link. 2. The performance of the **ls** command when used with the **-l** option can be improved by executing the **mkpasswd** command. This is helpful when a directory contains files owned by different users, such as the **/tmp** directory.
-m	Uses stream output format (a comma-separated series).
-n	Displays the same information as the **-l** flag, except that the **-n** flag displays the user and the group IDs instead of the user and group names.
-N	Does not follow symbolic links when determining the status of a file. **Note:** If both the **-L** and **-N** options are used, the last one will dominate. Also, any time a symbolic link is given that includes a / (slash) as the final character, the link will automatically be followed regardless of any options used.
-o	Displays the same information as the **-l** flag, except the **-o** flag suppresses display of the group and symbolic link information.
-p	Puts a slash after each file name if that file is a directory. This is useful when you pipe the output of the **ls** command to the **pr** command, as follows: `ls -p | pr -5 -t -w80`
-q	Displays nonprintable characters in file names as a ? (question mark).
-r	Reverses the order of the sort, giving reverse alphabetic or the oldest first, as appropriate.
-R	Lists all subdirectories recursively.
-s	Gives size in kilobytes (including indirect blocks) for each entry.
-S	Sorts with the primary key being file size (in decreasing order) and the secondary key being file name in the collating sequence (in increasing order).
-t	Sorts by time of last modification (latest first) instead of by name. For a symbolic link, the time used as the sort key is that of the symbolic link itself.

Item	Description
`-U`	Displays similar information as the `-l` flag. Displays the mode (including security information, named extended attribute information and encryption information), number of links, owner, group, size (in bytes), time of last modification, and name of each file. If the file is a special file, the size field contains the major and minor device numbers. If the file is a symbolic link, the path name of the linked-to file is printed preceded by a `->` (minus, greater than) sign. The attributes of the symbolic link are displayed.
-u	Uses the time of the last access, instead of the time of the last modification, for either sorting (when used with the **-t** flag) or for displaying (when used with the **-l** flag). This flag has no effect if it is not used with either the **-t** or **-l** flag, or both.
-x	Sorts output horizontally in a multi-column format.
-X	Prints long user names when used with other flags that display user names. The upper limit is determined by the **max_logname** ODM attribute in the PdAt and CuAt object classes. If a user name is greater than the **max_logname** attribute, it will be truncated to the number of characters as specified by the **max_logname** attribute, less one character.
-1	Forces output into one-entry-per-line format. This is the default when the output is not directed to a terminal.

Notes:

- If any of the **-l**, **-n**, **-s**, **-g**, or **-o** flag is specified, each file that is present in the directory is preceded by a status line that indicates the number of file system blocks occupied by files.
- If the **-k** flag is not specified along with any of the **-l**, **-n**, **-s**, **-g**, or **-o** flag, the status line indicates the number of file system blocks occupied by files in units of 512 bytes.
- If the **-k** flag is specified along with any of the **-l**, **-n**, **-s**, **-g**, or **-o** flag, the status line indicates the number of file system blocks occupied by files in units of 1024 bytes.
- Additionally, if necessary, the number of file system blocks occupied by files in the directory is rounded off to the next integral number of units.
- In the POSIX locale environment, the total %u\n is the output format which represents number of units in the directory.

Exit Status

This command returns the following exit values:

Item	Description
0	All files were written successfully.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list all files in the current directory, type:

   ```
   ls -a
   ```

   This lists all files, including **.** (dot), **..** (dot-dot), and other files with names beginning with a dot.

2. To display detailed information, type:

   ```
   ls   -l chap1  .profile
   ```

   This displays a long listing with detailed information about `chap1` and `.profile`.
3. To display detailed information about a directory, type:

   ```
   ls   -d   -l  .  manual  manual/chap1
   ```

   This displays a long listing for the directories `.` and `manual`, and for the file `manual/chap1`. Without the **-d** flag, this would list the files in the `.` and `manual` directories instead of the detailed information about the directories themselves.
4. To list the files in order of modification time, type:

   ```
   ls   -l -t
   ```

   This displays a long listing of the files that were modified most recently, followed by the older files.
5. To display detailed information with expanded user and group name, type:

   ```
   ls  -lX .profile
   ```

   This displays a long listing with detailed information about **.profile**.
6. To display data about whether extended attributes are set for the files in the current directory, type:

   ```
   ls -U
   ```

   - For releases AIX 5.3 and earlier:

     Example output:

     ```
     -rwSr-x---+   1 root     system           28 Apr 29 03:23 only_aixc
     -rwSr-x---E   1 root     system            4 Apr 29 03:23 only_aixc_ea
     -rw-r--r--E   1 root     system            4 Apr 29 03:23 only_ea
     ----------+   1 root     system          265 Apr 29 03:23 only_nfs4
     ----------E   1 root     system           64 Apr 29 03:23 only_nfs4_ea
     -rw-r--r---   1 root     system            4 Apr 29 03:23 only_regular
     ```

   - For releases AIX 6.1 and later:

     Example output:

     ```
     -rwSr-x---+    1 root     system           28 Apr 29 03:23 only_aixc
     -rwSr-x---E    1 root     system            4 Apr 29 03:23 only_aixc_ea
     -rw-r--r--E    1 root     system            4 Apr 29 03:23 only_ea
     ----------+    1 root     system          265 Apr 29 03:23 only_nfs4
     ----------E    1 root     system           64 Apr 29 03:23 only_nfs4_ea
     -rw-r--r---    1 root     system            4 Apr 29 03:23 only_regular
     -rwxrwxr-xe    2 root     system          256 May 25 16:27 encry_ex
     ```
7. To display information about the number of files system blocks in units of 512 bytes for the files in the current directory, type:

   ```
   ls -l
   ```

   Example output:

   ```
   total 16
   -rw-r--r--    1 root     system           22 Feb 05 05:29 sample1
   -rw-r--r--    1 root     system           12 Feb 05 05:29 sample2
   ```
8. To display information about the number of files system blocks in units of 1024 bytes for the files in the current directory, type:

   ```
    ls -lk
   ```

   Example output:

```
total 8
-rw-r--r--    1 root     system           22 Feb 05 05:29 sample1
-rw-r--r--    1 root     system           12 Feb 05 05:29 sample2
```

Files

Item	Description
/usr/bin/ls	Contains the **ls** command.
/etc/passwd	Contains user IDs.
/etc/group	Contains group IDs.
/usr/share/lib/terminfo/*	Contains terminal information.

ls-secldapclntd Command

Purpose

The **ls-secldapclntd** command lists the status of the **secldapclntd** daemon process.

Syntax

/usr/sbin/ls-secldapclntd

Description

The **ls-secldapclntd** command lists the **secldapclntd** daemon status. The information returned includes the following:

- The LDAP server the **secldapclntd** daemon is talking to
- The LDAP server port number
- The version of the LDAP protocol used
- User base DN
- Group base DN
- System (id) base DN
- User cache size
- User cache size used
- Group cache size
- Group cache size used
- Cache time out (time to live) value
- **secldapclntd** to LDAP server heart beat interval
- Number of thread used by **secldapclntd** daemon
- Authentication mechanism in use
- Attribute search mode
- Default user attribute entry location
- Timeout period (seconds) for LDAP client requests to the server
- User objectclass used in the LDAP server
- Group objectclass used in the LDAP server

Security

A user with the **aix.security.ldap** authorization is authorized to use this command.

Example

1. 1.To list the status of the **secldapclntd** daemon, type:

```
/usr/sbin/ls-secldapclntd
```

Files

Item	Description
/etc/security/ldap/ldap.cfg	Contains information needed by the **secldapclntd** daemon to connect to the server.

lsactdef Command

Purpose

Displays the action definitions of a resource or a resource class.

Syntax

To display the action definitions of a *resource:*

`lsactdef` [`-p` *property*] [`-s i` | o] [`-e`] [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] *resource_class* [*action1* [*action2* ...]]

To display the action definitions of a *resource class:*

`lsactdef -c` [`-p` *property*] [`-s i` | o] [`-e`] [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] *resource_class* [*action1* [*action2* ...]]

To display all resource class names:

`lsactdef`

Description

The `lsactdef` command displays a list of the action definitions of a resource or a resource class. By default, this command displays the action definitions of a *resource.* To see the action definitions of a *resource class,* specify the `-c` flag.

If you do not specify any actions on the command line, this command only displays actions that are defined as `public`. To override this default, use the `-p` flag or specify on the command line the names of the actions that have definitions you want to display.

To see the structured data definition that is required as input when this action is invoked, specify the `-s i` flag. To see the structured data definition linked with the output that results from invoking this action, specify the `-s o` flag.

By default, this command does not display action descriptions. To display action definitions and descriptions, specify the `-e` flag.

Flags

-c
Displays the action definitions for *resource_class*.

-d
Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the `-D` flag if you want to change the default delimiter.

-D *delimiter*
Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify a delimiter other than the default colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

-e
Specifies expanded format. Displays descriptions along with the action definitions.

-i
Specifies input format. Generates a template of *resource_data_input_file*. The output is displayed in long (stanza) format. The attribute's SD element data types are displayed as the value in the *attr=value* pairs. It is suggested that when you use this flag, the output of the `lsactdef` command be directed to a file. This flag overrides the `-s o` flag.

-l
Specifies "long" format — one entry per line. This is the default display format. If the `lsactdef` command is issued with the `-l` flag, but without a resource class name, the `-l` flag is ignored when the command returns the list of defined resource class names.

-p *property*
Displays actions with the specified *property*. By default, only the definitions for public actions are displayed. To display all action definitions regardless of the action property, use the `-p 0` flag.

Action properties:

0x0001
`long_running`

0x0002
`public`

A decimal or hexadecimal value can be specified for the property. To request the action definitions for all actions that have one or more properties, "OR" the properties of interest together and then specify the "OR"ed value with the `-p` flag. For example, to request the action definitions for all actions that are `long_running` or `public`, enter:

```
-p 0x03
```

-s i | o
Displays the structured data definition for the action input or action response.

i
Displays the action input structured data definitions. This is the default.

o
Displays the action response (output) structured data definitions.

-t
Specifies table format. Each attribute is displayed in a separate column, with one resource per line.

-x
Suppresses header printing.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software-service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

resource_class
Specifies the name of the resource class with the action definitions that you want to display. If *resource_class* is not specified, a list of all of the resource class names is displayed.

***action1* [*action2...*]**
Specifies one or more actions. If *resource_class* is specified, zero or more action names can be specified. If no actions are specified, all of the action definitions for *resource_class* are displayed. Enter specific action names to control which actions are displayed and in what order. Use blank spaces to separate action names.

Security

The user needs read permission for the *resource_class* specified in `lsactdef` to run `lsactdef`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for information about the ACL file and how to modify it.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To list the names of all of the resource classes, enter:

```
lsactdef
```

The output will look like this:

```
class_name
"IBM.Association"
"IBM.AuditLog"
"IBM.AuditLogTemplate"
"IBM.Condition"
"IBM.EventResponse"
"IBM.Host"
"IBM.Program"
"IBM.Sensor"
"IBM.ManagedNode"
...
```

2. To list the public resource action definitions for resource class IBM.AuditLog, enter:

```
lsactdef IBM.AuditLog
```

The output will look like this:

```
Resource Action Definitions for
class_name: IBM.AuditLog
action 1:
        action_name    = "GetRecords"
        display_name   = ""
        description    = ""
        properties     = {"public"}
        confirm_prompt = ""
        action_id      = 0
        variety_list   = {{1..1}}
        variety_count  = 1
        timeout        = 0
action 2:
        action_name    = "DeleteRecords"
        display_name   = ""
        description    = ""
        properties     = {"public"}
        confirm_prompt = ""
        action_id      = 1
        variety_list   = {{1..1}}
        variety_count  = 1
        timeout        = 0
....
```

3. To list the structured data definition required for invoking the action on resources in resource class IBM.AuditLog, action GetRecords, enter:

```
lsactdef -s i IBM.AuditLog GetRecords
```

The output will look like this:

```
Resource Action Input for: IBM.AuditLog
action_name GetRecords:
sd_element 1:
        element_name      = "MatchCriteria"
    display_name      = ""
    description       = ""
        element_data_type = "char_ptr"
    element_index     = 0
sd_element 2:
        element_name      = "IncludeDetail"
    display_name      = ""
    description       = ""
        element_data_type = "uint32"
        element_index     = 1
```

Location

/opt/rsct/bin/lsactdef

lsallq Command

Purpose

Lists the names of all configured queues.

Syntax

lsallq [**-c**]

Description

The **lsallq** command lists the names of all configured queues contained in the **/etc/qconfig** file. By specifying the **-c** flag, this listing is displayed in colon format. This flag is used mainly by SMIT.

You can also use the System Management Interface Tool (SMIT) **smit lsallq** fast path to run this command.

Flag

Item	Description
-c	Causes colon format output for use by SMIT.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list all of the queue names in the **/etc/qconfig** file, enter:

```
lsallq
```

A listing similar to the following is displayed:

```
lp0
lp1
lp2
```

2. To list all configured queues in colon format, enter:

   `lsallq -c`

   A listing similar to the following is displayed:

```
lp0
lp0:queue1
lp0:queue2
lp1
```

Files

Item	Description
/usr/bin/lsallq	Contains the **lsallq** command.
/etc/qconfig	Configuration file.

lsallqdev Command

Purpose

Lists all configured printer and plotter queue device names within a specified queue.

Syntax

lsallqdev [**-c**] **-q***Name*

Description

The **lsallqdev** command lists all configured device names within a specified queue in the **/etc/qconfig** file.

You can also use the System Management Interface Tool (SMIT) **smit lsallqdev** fast path to run this command.

Flags

Item	Description
-q *Name*	Specifies the queue name.
-c	Causes colon format output for use by SMIT.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list the names all of the configured queue devices within the `lp0` queue in the **/etc/qconfig** file, enter:

   ```
   lsallqdev -q lp0
   ```

   A listing similar to the following is displayed:

   ```
   lpd0
   lpd1
   lpd2
   ```

2. To list the names of all of the configured queue device within the `lp0` queue in the **/etc/qconfig** file in colon format, enter:

   ```
   lsallqdev -q lp0 -c
   ```

   A listing similar to the following is displayed:

   ```
   lp0:lpd1
   lp0:lpd2
   ```

Files

Item	Description
/usr/bin/lsallqdev	Contains the **lsallqdev** command.
/etc/qconfig	Configuration file.

lsarm command

Purpose

Displays Application Response Measurement (ARM) application and process usage information.

Syntax

lsarm **-a** [**-g**] [**-t**] [**-u**] [*ApplicationName* ...]

or

lsarm **-p** [**-a** [**-g**] [**-t**] [*ProcessID* ...]]

Description

The **lsarm** command displays information about applications registered with the operating system using the Application Response Measurement (ARM) APIs. The **-a** flag displays information about applications by application name. The **-p** option displays information about the applications used by a process.

Flags

Item	Description
-a	Displays application names.
-g	Displays group names associated with the application.

Item	Description
-p	Displays transaction names associated with the application.
-u	Displays process numbers using the application.
-t	Displays the applications used by a process.

Parameters

Item	Description
ApplicationName	Specifies a list of one or more applications for which the **lsarm** command should display information.
ProcessID	Specifies a list of one or more process IDs for which the **lsarm** command should display information.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Examples

1. To display all application names, type:

```
lsarm -a
```

2. To display group names, transaction class names, and process numbers using the 'database' application, type:

```
lsarm –a –g –t –u  database
```

3. To display the process IDs for all processes using ARM applications, type:

```
lsarm –p
```

4. To display the application names, group names, and transaction names used by process 25038, type:

```
lsarm –p –a –g –t 25038
```

Location

/usr/ewlm/sbin/lsarm

Related Information

lsassocmap Command

Purpose

Displays an association map.

Syntax

`lsassocmap` [`-c` *association_class*] [`-h`] [`-TV`] [*endpoint*...]

Description

The `lsassocmap` command displays the association classes available on a cluster, including the endpoints of each association. Names and endpoints of Common Information Model (CIM) association classes that are registered with the CIM resource manager are listed in table format, similar to the output of the `lscondresp` command.

If you specify the `lsassocmap` command without any parameters, it displays all of the association classes, endpoints, and roles. A *role* is the name of the class reference property in the association class definition. Roles can be used as parameters to the `-o` and `-R` flags of the `lsrsrcassoc` command to filter output. See "lsrsrcassoc Command" on page 2206 for more information.

The `-c` flag limits the associations displayed to only those provided by a specific association class. You can specify any number of classes by using the *endpoint* parameter; only associations containing those classes as references (endpoints) are displayed.

Parameters

***endpoint*...**
Specifies one or more endpoint classes. Only association classes containing references to one of the *endpoint* classes are displayed.

Flags

-c *association_class*
Displays associations for *association_class*.

-h
Writes the command usage statement to standard output.

-T
Writes the command trace messages to standard error. For your software service organization use only.

-V
Writes the command verbose messages to standard output.

Standard output

When the `-h` flag is specified, this command usage statement is written to standard output. When the `-V` flag is specified, this command verbose messages are written to standard output.

Standard error

When the `-T` flag is specified, this command trace messages are written to standard error.

Exit status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
The specified association class cannot be found.

Implementation specifics

This command is part of the `rsct.exp.cimrm` fileset, in the `rsct.exp` package on the AIX Expansion Pack and the Reliable Scalable Cluster Technology (RSCT) package for the Linux operating system.

Location

Item	Description
`/opt/rsct/bin/lsassocmap`	

Examples

To display associations that are available in a cluster, enter the following command:

```
lsassocmap
```

The following output is displayed for the AIX platform:

```
Association Class              Role 1          Associator 1           Role 2         Associator 2           Node
cimv2.IBMAIX_RunningOS         Antecedent      IBMAIX_OperatingSystem  Dependent      IBMAIX_ComputerSystem  c175nf14
cimv2.IBMAIX_OSProcess         GroupComponent  IBMAIX_OperatingSystem  PartComponent  IBMAIX_UnixProcess     c175nf14
cimv2.IBMAIX_CSProcessor       GroupComponent  IBMAIX_ComputerSystem   PartComponent  IBMAIX_Processor       c175nf14
cimv2.IBMAIX_HostedFileSystem  GroupComponent  IBMAIX_ComputerSystem   PartComponent  CIM_FileSystem         c175nf14
```

The following output is displayed for other platforms:

```
Association Class             Role 1          Associator 1           Role 2         Associator 2           Node
cimv2.Linux_RunningOS         Antecedent      Linux_OperatingSystem  Dependent      Linux_ComputerSystem  c175nf14
cimv2.Linux_OSProcess         GroupComponent  Linux_OperatingSystem  PartComponent  Linux_UnixProcess     c175nf14
cimv2.Linux_CSProcessor       GroupComponent  Linux_ComputerSystem   PartComponent  Linux_Processor       c175nf14
cimv2.Linux_HostedFileSystem  GroupComponent  Linux_ComputerSystem   PartComponent  CIM_FileSystem        c175nf14
```

lsattr Command

Purpose

Displays attribute characteristics and possible values of attributes for devices in the system.

Syntax

lsattr { **-D** [**-O**] | **-E** [**-O**] | **-P** [**-O**] | **-F** *Format* [**-Z** *Character*] } **-l** *Name* [**-a** *Attribute*] ...[**-f** *File*] [**-h**] [**-H**]

lsattr { **-D** [**-O**] | **-F** *Format* [**-Z** *Character*] } { [**-c** *Class*] [**-s** *Subclass*] [**-t** *Type*] } [**-a** *Attribute*] ... [**-f** *File*] [**-h**] [**-H**]

lsattr **-R** { **-l** *Name* | [**-c** *Class*] [**-s** *Subclass*] [**-t** *Type*] } **-a** *Attribute* [**-f** *File*] [**-h**] [**-H**]

lsattr **-l** *Name* { **-o** *operation* [...] } **-F** *Format* [**-Z** *Character*] [**-f** *File*] [**-h**] [**-H**]

lsattr { [**-c** *Class*] [**-s** *Subclass*] [**-t** *Type*] } { **-o** *operation* [...] } **-F** *Format* [**-Z** *Character*] [**-f** *File*] [**-h**] [**-H**]

Description

The **lsattr** command displays information about the attributes of a specific device or type of device. If you do not specify the device logical name with the **-l** *Name* flag, you must use a combination of one or all of the **-c** *Class*, **-s** *Subclass*, and **-t** *Type* flags to uniquely identify the predefined device.

You must specify one of the following flags with the **lsattr** command:

Item	Description
-D	Displays default values.
-E	Displays effective values (valid only for customized devices that are specified with the **-l** flag).
-F *Format*	Specifies the user-defined format.
-P	Displays device values when the device was last configured.
-R	Displays the range of legal values.

When you display the effective values of the attributes for a customized device, the information is obtained from the Configuration database, not the device. The database values reflect how the device is configured unless it is reconfigured with the **chdev** command by using the **-P** or **-T** flag. If the reconfiguration occurs, the information that is displayed by the **lsattr** command might not correctly indicate the current device configuration until after the next system boot.

If you use the **-D** or **-E** flag, the output defaults to the values for the attribute's name, value, description, and user-settable strings, unless it is also used with the **-O** flag.

The **-P** flag displays the attribute values when the device was last configured, or before modifying any of its attributes by using the **chdev** command with the **-P** or **-T** flag.

The **-O** flag displays the names of all the attributes that are specified, separated by colons. On the next line, the **-O** flag displays all of the corresponding attribute values, separated by colons. The **-H** flag can be used with either the **-D**, **-E**, or **-F** flag to display headers above the column names. You can define the format of the output with a user-specified format by using the **-F** *Format* flag, where the *Format* parameter is a quoted list of column names, separated by non-alphanumeric characters or white space. If the **-F** *Format* flag is specified, the **-Z** *Character* flag can also be specified to change the default record separator from a `newline` character to the indicated *Character*.

The **lsattr** command can display *"operation"* information from the Extended Predefined Attribute (**PdAtXtd**) object class. The operation information is accessed through the **-o** *operation* flag. The **-o** *operation* flag and the **-a** *attribute* flag cannot be specified in the same invocation of the **lsattr** command. The **-o** *operation* flag is also not valid with the **-R** flag. When the **-o** *operation* flag is specified, only fields from the **PdAtXtd** object class can be specified with the **-F** *Format* flag.

You can supply the flags either on the command line or by using the specified **-f** *File* flag.

Flags

Item	Description
-a *Attribute*	Displays information for the specified attributes of a specific device or type of device. You can use one **-a** flag for each attribute name or multiple attribute names. If you use one **-a** flag for multiple attribute names, the list of attribute names must be enclosed in quotation marks with spaces between the names. If you use the **-R** flag, you must specify only one **-a** flag with only one attribute name. If you do not specify either the **-a** or **-R** flag, the **lsattr** command displays all information for all attributes of the specified device. The **-a** *Attribute* flag cannot be used with the **-o** *Operation* flag. This combination of flags causes the **lsattr** command to exit with an error message.

Item	Description
-c *Class*	Specifies a device class name. This flag can be used to restrict the output to devices of a specified class. This flag cannot be used with the **-E** or **-l** flag.
-D	Displays the attribute names, default values, descriptions, and user-settable flag values for a specific device when it is not used with the **-O** flag. The **-D** flag displays only the attribute name and default value in colon format when it is used with the **-O** flag. This flag can be used with any combination of the **-c**, **-s**, and **-t** flags that uniquely identifies a device from the Predefined Devices object class, or with the **-l** flag. This flag cannot be used with the **-E**, **-F**, or **-R** flag.
-E	Displays the attribute names, current values, descriptions, and user-settable flag values for a specific device when it is not used with the **-O** flag. The **-E** flag displays only the attribute name and current value in colon format when it is used with the **-O** flag. This flag cannot be used with the **-c**, **-D**, **-F**, **-R**, **-s**, or **-t** flag.
-f *File*	Reads the necessary flags from the *File* parameter.
-F *Format*	Displays the output in a user-specified format, where the *Format* parameter is a quoted list of column names separated by nonalphanumeric characters or white space. If white space is used as the separator, the **lsattr** command displays the output in aligned columns. Only column names from the Predefined Attributes (**PdAt**), Customized Attributes (**CuAt**), and the Extended Predefined Attributes (**PdAtXtd**) object classes can be specified. In addition to the column names, there are two special purpose names that can be used: the name *description* can be used to obtain a display of attribute descriptions and *user_settable* can be used to determine whether an attribute can be changed. This flag cannot be used with the **-E**, **-D**, **-O**, or **-R** flag.
-H	Displays headers above the column output. The **-O** and **-R** flags take precedence over the **-H** flag.
-h	Displays the command usage message.
-l *Name*	Specifies the device logical name in the Customized Devices object class whose attribute names or values you want displayed.
-o *Operation*	Displays information for the specified operations of a specific device or type of device. You can use one **-o** flag for each operation name or multiple operation names. If you use one **-o** flag for multiple operation names, the list of operation names must be enclosed in quotation marks with spaces between the names. Wildcard characters can also be used for the operation name. The valid set of wildcard characters is the same set that is used by the **odmget** command. All operations that are associated with a specific device, or type of device, can be displayed by using an operation value of "?*". The **-o** *Operation* flag cannot be used with the **-a** *attribute* flag or the **-R** flag. Any combination of these flags causes the **lsattr** command to exit with an error message.
-O	Displays all attribute names separated by colons and on the second line, displays all the corresponding attribute values separated by colons. The attribute values are current values when the **-E** flag is specified and default values when the **-D** flag is specified. This flag cannot be used with the **-F** and **-R** flags.

Item	Description
-P	Displays the attribute names, values, descriptions, and user-settable flag values for a specific device when it is not used with the **-O** flag. The values that are displayed are those values with which the device was configured, before any of the device attributes were modified by using the **chdev** command with the **-P** or **-T** flag. When the **-P** flag is used with the **-O** flag, the **-P** flag displays only the attribute name and value in colon-separated format. This flag can be used with any combination of the **-c**, **-s**, and **-t** flags that uniquely identifies a device from the Predefined Devices object class, or with the **-l** flag. This flag cannot be used with the **-D**, **-E**, **-O**, or **-R** flag.
-R	Displays the legal values for an attribute name. The **-R** flag cannot be used with the **-D**, **-E**, **-F** and **-O** flags, but can be used with any combination of the **-c**, **-s**, and **-t** flags that uniquely identifies a device from the Predefined Devices object class, or with the **-l** flag. The **-R** flag displays the list attribute values in a vertical column as follows: `Value1` `Value2` `.` `.` `ValueN` The **-R** flag displays the range attribute values as `x...n(+i)` where x is the start of the range, n is the end of the range, and `i` is the increment.
-s *Subclass*	Specifies a device subclass name. This flag can be used to restrict the output to that for devices of a specified subclass. This flag cannot be used with the **-E** or **-l** flag.
-t *Type*	Specifies a device type name. This flag can be used to restrict the output to that for devices of a specified class. This flag cannot be used with the **-E** or **-l** flag.
-Z *Character*	The **-Z** *Character* flag is used with programs that must deal with ODM fields that might have embedded new line characters. The **-Z** *Character* flag is used to change the record separator character for each record, or line, of output generated. The new record separator is specified by using the *Character* argument to this flag. The **-Z** *Character* flag is only relevant when the **-F** *Format* flag is specified. The **-Z** *Character* flag cannot be used with the **-D**, **-E**, **-O**, or **-R** flag.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list the current attribute values for the `rmt0` tape device, enter the following command:

   ```
   lsattr -l rmt0 -E
   ```

   The system displays a message similar to the following output:

   ```
   mode          yes   Use DEVICE BUFFERS during writes     True
   block_size    1024  BLOCK size (0=variable length)       True
   extfm         yes   Use EXTENDED file marks              True
   ret           no    RETENSION on tape change or reset    True
   density_set_1 37    DENSITY setting #1                   True
   density_set_2 36    DENSITY setting #2                   True
   compress      yes   Use data COMPRESSION                 True
   ```

```
size_in_mb    12000 Size in Megabytes                    False
ret_error     no    RETURN error on tape change or reset True
```

2. To list the default attribute values for the `rmt0` tape device, enter the following command:

```
lsattr -l rmt0 -D
```

The system displays a message similar to the following output:

```
mode          yes   Use DEVICE BUFFERS during writes     True
block_size    1024  BLOCK size (0=variable length)       True
extfm         yes   Use EXTENDED file marks              True
ret           no    RETENSION on tape change or reset    True
density_set_1 37    DENSITY setting #1                   True
density_set_2 36    DENSITY setting #2                   True
compress      yes   Use data COMPRESSION                 True
size_in_mb    12000 Size in Megabytes                    False
ret_error     no    RETURN error on tape change or reset True
```

3. To list the current value of the **bus_intr_lvl** attribute for the `scsi0` SCSI adapter, enter the following command:

```
lsattr -l scsi0 -a bus_intr_lvl -E
```

The system displays a message similar to the following output:

```
bus_intr_lvl 1 Bus interrupt level False
```

4. To list the possible values of the login attribute for the `tty0` tty device, enter the following command:

```
lsattr -l tty0 -a login -R
```

The system displays a message similar to the following output:

```
enable
disable
share
delay
```

5. To list the default attribute values for an IBM 4340 parallel printer, enter the following command:

```
lsattr -c printer -s parallel -t ibm4340 -D
```

The system displays a message similar to the following output:

```
ptop       600       Printer TIME OUT period                    True
line       60        Number of LINES per page                   True
col        80        Number of COLUMNS per page                 True
ind        0         Number of columns to INDENT                True
plot       no        Send all characters to printer UNMODIFIED  True
backspace  yes       Send BACKSPACES                            True
cr         yes       Send CARRIAGE RETURNS                      True
form       yes       Send FORM FEEDS                            True
lf         yes       Send LINE FEEDS                            True
addcr      yes       Add CARRIAGE RETURNS to LINE FEEDS         True
case       no        Convert lowercase to UPPERCASE             True
tabs       yes       EXPAND TABS on eight position boundaries   True
wrap       no        WRAP CHARACTERS beyond the specified width True
mode       no        Return on ERROR                            True
interface  standard  Type of PARALLEL INTERFACE                 True
autoconfig available STATE to be configured at boot time        True
busy_delay 0         Microseconds to delay between characters   True
```

6. To list the possible values of the **ptop** attribute for an IBM 4340 parallel printer, enter the following command:

```
lsattr -c printer -s parallel -t ibm4340  -a ptop -R
```

The system displays a message similar to the following output:

```
1...1000 (+1)
```

7. To list the current attribute values for the `rmt0` tape device in colon-separated format, enter the following command:

```
lsattr -l rmt0 -E -O
```

The system displays a message similar to the following output:

```
#mode:block_size:extfm:ret:density_set_1:density_set_2:compress:size_in_mb:ret_error
yes:1024:yes:no:37:36:yes:12000:no
```

8. To display system attributes, enter the following command:

```
lsattr -E -l sys0
```

The system displays output similar to the following output:

```
keylock      normal        State of system keylock at boot time             False
maxbuf       20            Maximum number of pages in block I/O BUFFER CACHE True
maxmbuf      0             Maximum Kbytes of real memory allowed for MBUFS   True
maxuproc     128           Maximum number of PROCESSES allowed per user      True
autorestart  false         Automatically REBOOT system after a crash         True
iostat       false         Continuously maintain DISK I/O history            True
realmem      4194304       Amount of usable physical memory in Kbytes        False
conslogin    enable        System Console Login                              False
fwversion    IBM,SPH00221  Firmware version and revision levels              False
maxpout      0             HIGH water mark for pending write I/Os per file   True
minpout      0             LOW water mark for pending write I/Os per file    True
fullcore     false         Enable full CORE dump                             True
pre430core   false         Use pre-430 style CORE dump                       True
ncargs       256           ARG/ENV list size in 4K byte blocks               True
rtasversion  1             Open Firmware RTAS version                        False
modelname    IBM,7044-270  Machine name                                      False
systemid     IBM,011037D1F Hardware system identifier                        False
boottype     disk          N/A                                               False
SW_dist_intr false         Enable SW distribution of interrupts              True
cpuguard     disable       CPU Guard                                         True
frequency    93750000      System Bus Frequency                              False
```

Note: The same information is available in a more readable format by using SMIT. Select the **System Environments** -> **Change / Show Characteristics of Operating Systems** options to view this information.

Files

Item	Description
/usr/sbin/lsattr	Contains the **lsattr** command.

lsaudrec Command

Purpose

Lists records from the audit log.

Syntax

`lsaudrec` [`-l`] [`-a` | `-n` *node_name1*[`,`*node_name2*]...] [`-S` *subsystem_name*] [`-s` *selection_string*] [`-x`] [`-h`] [*field_name1* [*field_name2*...]]

Description

The `lsaudrec` command is used to list records in the audit log. The audit log is a facility for recording information about the system's operation. It can include information about the normal operation of the system as well as failures and other errors. It augments the error log functionality by conveying the

relationship of the error relative to other system activities. All detailed information about failures is still written to the AIX® error log.

Records are created in the audit log by subsystems that have been instrumented to do that. For example, the event response subsystem runs in the background to monitor administrator-defined conditions and then invokes one or more actions when a condition becomes true. Because this subsystem runs in the background, it is difficult for the operator or administrator to understand the total set of events that occurred and the results of any actions that were taken in response to an event. Because the event response subsystem records its activity in the audit log, the administrator can easily view its activity as well as that of other subsystems using this command.

Each record in the audit log contains named fields. Each field contains a value that provides information about the situation corresponding to the record. For example, the field named `Time` indicates the time at which the situation occurred. Each record has a set of common fields and a set of subsystem-specific fields. The common fields are present in every record in the audit log. The subsystem-specific fields vary from record to record. Their names are only significant when used with a subsystem name because they may not be unique across all subsystems. Each record is derived from a template that defines which subsystem-specific fields are present in the record and defines a format string that is used to generate a message describing the situation. The format string may use record fields as inserts. A subsystem typically has many templates.

The field names can be used as variables in a *selection string* to choose which records are displayed. A selection string is an expression that is made up of field names, constants, and operators. The syntax of a selection string is similar to an expression in the C programming language or the SQL "where" clause. The selection string is matched against each record using the referenced fields of each record to perform the match. Any records that match are displayed. The selection string is specified with the `-s` flag. For information on how to specify selection strings, see the *Administering RSCT* guide.

You can also specify field names as parameters to this command to choose which fields are displayed and the order in which they are displayed. The common field names are:

Field	Description
Time	The time when the situation occurred that the record corresponds to. The value is a 64-bit integer and represents the number of microseconds since UNIX Epoch (00:00:00 GMT January 1, 1970). See the constants below for specifying the time in more user-friendly formats.
Subsystem	The subsystem that generated the record. This is a string.
Category	Indicates the importance of the situation corresponding to the audit record, as determined by the subsystem that generated the record. The valid values are: `0` (informational) and `1` (error).
SequenceNumber	The unique 64-bit integer that is assigned to the record. No other record in the audit log will have the same sequence number.
TemplateId	The subsystem-dependent identifier that is assigned to records that have the same content and format string. This value is a 32-bit unsigned integer.
NodeName	The name of the node from which the record was obtained. This field name cannot be used in a selection string.

In addition to the constants in expressions, you can use the following syntax for dates and times with this command:

#mmddhhmmyyyy
: This format consists of a sequence of decimal characters that are interpreted according to the pattern shown. The fields in the pattern are, from left to right: *mm* = month, *dd* = day, *hh* = hour, *mm* = minutes, *yyyy* = year. For example, `#010523042004` corresponds to January 5, 11:04 PM, 2004. The fields can be omitted from right to left. If not present, the following defaults are used: year = the current year, minutes = 0, hour = 0, day = 1, and month = the current month.

#-mmddhhmmyyyy
This format is similar to the previous one, but is relative to the current time and date. For example, the value `#-0001` corresponds to one day ago and the value `#-010001` corresponds to one month and one hour ago. Fields can be omitted starting from the right and are replaced by 0.

The audit records considered for display and matched against the selection string can be restricted to a specific subsystem by using the `-S` flag. If this flag is specified, the subsystem-specific field names can be used in the selection string in addition to the common field names.

The nodes from which audit log records are considered for display and matched against the selection string can be restricted to a set of specific nodes by using the `-n` flag. If this flag is specified, the search is limited to the set of nodes listed. Otherwise, the search is performed for all nodes defined within the current management scope, as determined by the CT_MANAGEMENT_SCOPE environment variable.

The audit records are displayed in a table. Field names specified as parameters control which fields are displayed and the order in which they appear on each line. By default, the columns displayed are: the date and time, the subsystem name that generated the record, the severity of the situation, and the subsystem-specific message that describes the situation. If the management scope is not local, the node name is displayed in the first column.

Flags

-l
Indicates that long output should be produced. Long output includes subsystem-specific fields that are not included in the formatted message text.

-a
Specifies that records from all nodes in the domain are to be displayed. If both the `-n` and the `-a` flags are omitted, records from the local node only are displayed.

-n *node_name1*[,*node_name2*]...
Specifies the list of nodes containing audit log records that will be examined and displayed if they meet the other criteria, such as matching the specified selection string. Node group names can also be specified, which are expanded into a list of node names. If both the `-n` and the `-a` flags are omitted, records from the local node only are displayed.

-S *subsystem_name*
Specifies a subsystem name. If this flag is present, only records identified by *subsystem_name* are considered for display. The records displayed can be further restricted by the `-s` flag. If the subsystem name contains any spaces, it must be enclosed in single or double quotation marks.

For backward compatibility, the subsystem name can be specified using the `-n` flag *only* if the `-a` and the `-S` flags are *not* specified.

-s *selection_string*
Specifies a selection string. This string is evaluated against each record in the audit log. All records that match the selection string will be displayed. If the selection string contains any spaces, it must be enclosed in single or double quotation marks. For information on how to specify selection strings, see the *Administering RSCT* guide.

The names of fields in the record can be used in the expression. If the `-S` flag is not specified, only the names of common fields can be used. See the **Description** for a list of the common field names and their data types. If the `-S` flag is specified, the name of any field for the specified subsystem as well as the common field names can be used.

If this flag is omitted, the records that are displayed will depend on the `-S` flag. If the `-S` flag is omitted, all records from the audit log are displayed. Otherwise, all records for the subsystem identified by the `-S` flag are displayed.

-x
Excludes the header (suppresses header printing).

-h
Writes the command's usage statement to standard output.

Parameters

***field_name1* [*field_name2*...]**
Specifies one or more fields in the audit log records to be displayed. The order of the field names on the command line corresponds to the order in which they are displayed. If no field names are specified, `Time`, `Subsystem`, `Severity`, and `Message` are displayed by default. If the management scope is not local, `NodeName` is displayed as the first column by default. See the **Description** for information about these and other fields.

Security

In order to list records from an audit log when the `-S` flag is omitted, you must have read access to the target resource class on each node from which records are to be listed. When the `-S` flag is specified, you must have read access to the audit log resource corresponding to the subsystem identified by the `-S` flag on each node from which records are to be listed.

Authorization is controlled by the RMC access control list (ACL) file that exists on each node.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon is established. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that can be affected by this command.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines (in conjunction with the `-a` and `-n` flags) the management scope that is used for the session with the RMC daemon. The management scope determines the set of possible target nodes where audit log records can be listed. If the `-a` and `-n` flags are not specified, local scope is used. When either of these flags is specified, CT_MANAGEMENT_SCOPE is used to determine the management scope directly. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output.

Examples

1. To list all records in the audit log on every node in the current management scope as determined by the CT_MANAGEMENT_SCOPE environment variable, enter:

```
lsaudrec
```

2. To list all records that were logged in the last hour on every node in the current management scope as determined by the CT_MANAGEMENT_SCOPE environment variable, enter:

```
lsaudrec -s "Time > #-000001"
```

3. To list the time and sequence number of every record in the audit log for the subsystem `abc` on nodes `mynode` and `yournode`, enter:

```
lsaudrec -n mynode,yournode -S abc Time SequenceNumber
```

4. To list the records that are generated by the event-response resource manager (ERRM), enter:

```
lsaudrec -SERRM
```

5. To list the records that are related to a condition called `Condition1`, enter:

```
lsaudrec -SERRM -s"ConditionName=='Condition1'"
```

6. To list the records that are related to an event from `Condition1`, enter:

```
lsaudrec -SERRM -s"ConditionName=='Condition1' && Etype==91"
```

7. To list the records that are related to a rearm event from **Condition1**, enter:

```
lsaudrec -SERRM -s"ConditionName=='Condition1' && Etype==92"
```

8. To list the sensor resource manager records in the audit log on the local node, enter:

```
lsaudrec -SSSRM
```

The output will look like this:

```
Time                 Subsystem   Category    Description
11/10/05 21:52:32    SSRM        Error       The Command /SENSOR/sensor.ksh 1
                                             in Sensor SENSOR_NOUSER_1 execution fails.
11/10/05 21:52:36    SSRM        Error       The Command /SENSOR/sensor.nocmd 1 in Sensor
                                             SENSOR_NOCMD_1 exits with error 127.
```

9. To list, in long format, the sensor resource manager records in the audit log on the local node, enter:

```
lsaudrec -l -SSSRM
```

The output will look like this:

```
Time        = 11/10/05 21:52:32 243097
Subsystem   = SSRM
Category    = Error
Description = The Command /SENSOR/sensor.ksh 1 in Sensor SENSOR_NOUSER_1 execution fails.
ErrorMsg    = 2645-202 The user name "guest" that was specified for running the command does not exist.

Time        = 11/10/05 21:52:36 361726
Subsystem   = SSRM
Category    = Error
Description = The Command /SENSOR/sensor.nocmd 1 in Sensor SENSOR_NOCMD_1 exits with error 127.
StandardOut =
StandardErr = ksh: /u/diane/drmc/scripts/SENSOR/sensor.nocmd:  not found
```

10. To list error records only, enter:

```
lsaudrec -s"Category=1"
```

Location

/opt/rsct/bin/lsaudrec

lsauth Command

Purpose

Displays user and system-defined authorization attributes.

Syntax

lsauth [**-R** *load_module*] [**-C**] [**-f**] [**-a** *List*] {**ALL** | *Name* [,*Name*] ...}

Description

The **lsauth** command displays attributes of user-defined and system-defined authorizations from the authorization database. The command can be used to list attributes of all authorizations or specific authorizations. By default, the **lsauth** command displays all authorization attributes. To view selected attributes, use the **-a** *List* flag. If one or more attributes cannot be read, the **lsauth** command lists the information that is available.

By default, the **lsauth** command lists the attributes of each authorization on one line. It displays attribute information in the form of *Attribute* **=** *Value*, each separated by a blank space. To list the authorization attributes in stanza format, use the **-f** flag. To list the information as colon-separated records, use the **-C** flag.

If the system is configured to use multiple domains for the authorization database, the authorizations, as specified by the *Name* parameter, are searched from the domains in the order specified by the **secorder** attribute of the authorizations stanza in the **/etc/nscontrol.conf** file. If duplicate entries exist in multiple domains, only the first entry instance is listed. Use the **-R** flag to list the authorizations from a specific domain.

Flags

Item	Description
-a *List*	Lists the attributes to display. The *List* parameter requires a blank space between attributes to list multiple attributes. If you specify an empty list, only the authorization names are displayed. The *List* parameter can include any attribute defined in the **chauth** command, in addition to the following two attributes: **description** The text description of the authorization as indicated by the **dfltmsg**, **msgcat**, **msgset** and **msgnum** attributes for the authorization. **roles** A comma-separated list of roles containing the specified authorization in their authorization set.
-C	Displays the authorization attributes in colon-separated records, as follows: `#authorization:attribute1:attribute2: ...` `authorization:value1:value2: ...` `authorization2:value1:value2: ...` The output is preceded by a comment line that has details about the attribute represented in each colon-separated field. If you specify the **-a** flag, the order of the attributes matches the order specified in the **-a** flag. If an authorization does not have a value for a given attribute, the field is still displayed but is empty. The last field in each entry is ended by a newline character rather than a colon.
-f	Displays the output in stanzas, with each stanza identified by an authorization name. Each *Attribute* **=** *Value* pair is listed on a separate line: `Authorization:` `attribute1=value` `attribute2=value` `attribute3=value`
-R *load_module*	Specifies the loadable module to list authorizations from.

Parameters

Item	Description
ALL	Specifies to list attributes from all authorizations.
Name	Specifies the authorization name to list. Optionally, a wild card (.*) can be used at the end of a name to list an entire hierarchy. The entire string specified before the wild card must be a valid authorization name.

Security

The **lsauth** command is a privileged command. You must assume a role that has the following authorization to run the command successfully.

Item	Description
aix.security.auth.list	Required to run the command.

Files Accessed

Item	Description
File	Mode
/etc/security/authorizations	r

Examples

1. To display all attributes of the `custom` authorization, use the following command:

   ```
   lsauth custom
   ```

   All the attribute information appears, with each attribute separated by a blank space.
2. To display all attributes of the `custom` authorization from LDAP, use the following command:

   ```
   lsauth -R LDAP custom
   ```

   All the attribute information appears, with each attribute separated by a blank space.
3. To display the authorization ID and description for the `custom` authorization in stanza format, use the following command:

   ```
   lsauth -f -a id description custom
   ```

   Information similar to the following appears:

   ```
   custom:
       id=11000
       description="Custom Authorization"
   ```
4. To display the **msgcat**, **msgset** and **msgnum** attributes for the `custom.test` authorization in a colon format, use the following command:

   ```
   lsauth -C -a msgcat msgset msgnum custom.test
   ```

   Information similar to the following example appears:

   ```
   #name:ID:msgcat:msgset:msgnum
   custom.test:12000:custom_auths.cat:5:24
   ```
5. To display the description for the entire authorization hierarchy that begins with `aix.security`, use the following command:

   ```
   lsauth -a description aix.security.*
   ```

   The `aix.security` authorization and all its children are listed with one authorization per line and a space between the authorization name and the description attribute.

lsauthent Command

Purpose

Lists the authentication methods currently configured on the system.

Syntax

lsauthent

Description

The **lsauthent** command calls the **get_auth_method** subroutine in the **libauthm.a** library, translates a list of authentication methods returned, and prints the authentication methods configured to **stdout**. Each authentication method is outputted on a separate line.

The authentication methods are listed in the order in which they are configured. If none of the authentication methods are configured, **lsauthent** returns without printing anything.

The **lsauthent** command writes an error message to **stderr** and returns a -1 if **get_auth_method** fails.

Examples

If all of the authentication methods are configured as:

```
lsauthent
```

the output would consist of:

```
Kerberos 5
Kerberos 4
Standard AIX
```

lsC2admin Command

Purpose

Display the name of the current C2 System Administrative Host.

Syntax

lsC2admin

Description

The **lsC2admin** command displays the name of the administrative host. An administrative host must have been defined, and the system must have been installed in C2 mode for this command to operate successfully.

Exit Status

0
The administrative host information has been successfully displayed.

1
This system was not installed with C2 security.

2
This system has not been initialized to operate in C2 mode.

3
An error occurred while displaying the name of the administrative host.

Files

Item	Description
/usr/sbin/lsC2admin	Contains the lsC2admin command.

lsCCadmin Command

Purpose

Display the name of the current Common Criteria enabled System Administrative Host.

Syntax

lsCCadmin

Description

The **lsCCadmin** command displays the name of the administrative host. An administrative host must have been defined, and the system must have been installed in Common Criteria enabled mode for this command to operate successfully.

Exit Status

0
The administrative host information has been successfully displayed.

1
This system was not installed with Common Criteria enabled security.

2
This system has not been initialized to operate in Common Criteria enabled mode.

3
An error occurred while displaying the name of the administrative host.

Files

Item	Description
/usr/sbin/lsCCadmin	Contains the lsCCadmin command.

lscfg Command

Purpose

Displays configuration, diagnostic, and vital product data (VPD) information about the system.

Syntax

To Display Specific Data on all Systems

lscfg [**-v**] [**-p**] [**-s**] [**-l** *Name*]

Description

If you run the **lscfg** command without any flags, it displays the name, location and description of each device found in the current Customized VPD object class that is a child device of the **sys0** object. It will not display any device that has been marked `missing` in the Customized Device Object Class. The list is sorted by parent, child, and device location. Information on a specific device can be displayed with the **-l** flag.

Use the **lscfg** command to display vital product data (VPD) such as part numbers, serial numbers, and engineering change levels from either the Customized VPD object class or platform specific areas. Not all devices contain VPD data.

VPD data that is preceded by ME signifies that the VPD data was entered manually using a diagnostic service aid. For some devices, the vital product data is collected automatically from the devices through methods and added to the Customized VPD object class.

If you run the **lscfg** command with the **-p** flag, it displays device information stored in the platform specific data areas. When used with the **-v** flag, VPD data stored for these devices is also displayed. This information is obtained on a Common Hardware Reference Platform (CHRP) system from the open firmware device tree.

Item	Description
-l *Name*	Displays device information for the named device.
-p	Displays the platform-specific device information.
-v	Displays the VPD found in the Customized VPD object class. Also, displays platform specific VPD when used with the **-p** flag.
-s	Displays the device description on a separate line from the name and location.

Examples

1. To display the system configuration, enter:

```
         lscfg
```

   The system displays a message similar to the following:

```
INSTALLED RESOURCE LIST
```

   The following resources are installed on the machine:

```
+/- = Added or deleted from Resource List.
*   = Diagnostic support not available.


+ indicates the resource has been added to the Diagnostic Resource List. The resource is
added to the Diagnostic Resource list by default when the diagnostic fileset is installed. A
resource must be in the Diagnostic Resource List before diagnostics tasks can be
performed on the resource.

- indicates the resource was deleted from the Diagnostic Resource List.

The resource can be added or deleted from the Resource List by running the diag
command, and using the Task Selection menu to select either Add Resource to Resource
List, or Delete Resource from Resource List.

Diagnostic support for a resource, indicated by the + character is not necessarily
inclusive of all diagnostic tasks. Some resources are only supported with a subset of
diagnostic task, and that subset might or might not include the Run Diagnostics Task.

  Model Architecture: chrp
  Model Implementation: Multiple Processor, PCI bus

+ sys0                             System Object
+ sysplanar0                       System Planar
+ mem0                             Memory
+ L2cache0                         L2 Cache
+ proc0            U1.1-P1-C1      Processor
* pci3             U0.2-P1         PCI Bus
+ scsi0            U0.1-P1/Z1      Wide/Ultra-2 SCSI I/O Controller
+ rmt0             U1.1-P1/Z1-A3   SCSI 4mm Tape Drive (12000 MB)
+ cd0              U1.1-P1/Z1-A5   SCSI Multimedia CD-ROM Drive (650 MB)
+ hdisk0           U1.1-P1/Z1-A9   16 Bit LVD SCSI Disk Drive (4500 MB)
+ fd0              U0.1-P1-D1      Diskette Drive
..
..
```

2. To display the system configuration with the device description on a separate line, enter:

```
         lscfg  -s
```

The system displays a message similar to the following:

```
INSTALLED RESOURCE LIST
```

The following resources are installed on the machine:

```
+/- = Added or deleted from Resource List.
*   = Diagnostic support not available.

  Model Architecture: chrp
  Model Implementation: Multiple Processor, PCI bus

+ indicates the resource has been added to the Diagnostic Resource List. The resource is
added to the Diagnostic Resource list by default when the diagnostic fileset is installed. A
resource must be in the Diagnostic Resource List before diagnostics tasks can be
performed on the resource.

- indicates the resource was deleted from the Diagnostic Resource List.

The resource can be added or deleted from the Resource List by running the diag
command, and using the Task Selection menu to select either Add Resource to Resource
List, or Delete Resource from Resource List.

Diagnostic support for a resource, indicated by the + character is not necessarily
inclusive of all diagnostic tasks. Some resources are only supported with a subset of
diagnostic task, and that subset might or might not include the Run Diagnostics Task.

+ sys0
        System Object
+ sysplanar0
        System Planar
+ mem0
        Memory
+ L2cache0
        L2 Cache
+ proc0            U5734.100.1234567-P1-C1
        Processor
+ proc1            U5734.100.1234567-P1-C2
        Processor
+hdisk0            U5734.100.1234567-P1-D9
      16 Bit LVD SCSI Disk Drive (4500 MB)
+fd0               U5734.100.1234567-P1-D1
      Diskette Drive

..
..
```

3. To display the name, location, and description for devices specified by the logical name proc without VPD, enter:

```
lscfg -lproc\*
```

The system displays information for all devices with logical names beginning with proc, as follows:

```
  proc0            U1.1-P1-C1  Processor
  proc1            U1.1-P1-C1  Processor
  proc2            U1.1-P1-C1  Processor
  proc3            U1.1-P1-C1  Processor
  proc4            U1.1-P1-C2  Processor
  proc5            U1.1-P1-C2  Processor
  proc6            U1.1-P1-C2  Processor
  proc7            U1.1-P1-C2  Processor
```

4. To display the VPD for a specific device specified by the logical name ent0, enter:

```
         lscfg  -v -l ent0
```

The system displays the following:

```
 ent0             U0.1-P1-I2/E1   Gigabit Ethernet-SX PCI Adapter (14100401)

        Network Address.............0004AC7C00C4
        Displayable Message.........Gigabit Ethernet-SX PCI Adapter (14100401)

        EC Level....................E77998
```

```
        Part Number.................07L8916
        FRU Number..................07L8918
        Device Specific.(YL)........U0.1-P1-I2/E1
```

5. To display the VPD in the open firmware device tree for the corresponding node to the ent0 device, enter:

```
lscfg -vp -lent0
```

   The following is displayed:

```
  ent0             U0.1-P1-I2/E1  Gigabit Ethernet-SX PCI Adapter (14100401)

        Network Address.............0004AC7C00C4
        Displayable Message.........Gigabit Ethernet-SX PCI Adapter (14100401)

        EC Level....................E77998
        Part Number.................07L8916
        FRU Number..................07L8918
        Device Specific.(YL)........U0.1-P1-I2/E1


  PLATFORM SPECIFIC

  Name:  ethernet
    Model:  Galaxy, EtherLink 1000-SX-IBM
    Node:  ethernet@1
    Device Type:  network
    Physical Location: U0.1-P1-I2/E1
```

Files

Item	Description
/usr/sbin/lscfg	Contains the **lscfg** command.

lscifscred Command

Purpose

Lists the server or user entries stored in the `/etc/cifs_fs/cifscred` file.

Syntax

`lscifscred` [`-h` *RemoteHost*] [`-u` *user*]

Description

The `lscifscred` command lists all of the server or user entries that have passwords stored in the `/etc/cifs_fs/cifscred` file.

Flags

Item	Description
`-h` *RemoteHost*	Lists credentials matching the given remote host (CIFS server) only.
`-u` *user*	Lists credentials matching the given user name only.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Examples

1. To list all server and user entries on a server named `server1`, enter:

```
lscifscred -h server1
```

Location

`/usr/sbin/lscifscred`

lscifsmnt Command

Purpose

Lists the CIFS mounts defined in the `/etc/filesystems` file.

Syntax

`lscifsmnt` [`-c` | `-l` | `-p`] [*FileSystem*]

Description

The `lscifsmnt` command lists the specified CIFS mounts that are defined in the `/etc/filesystems` file.

Flags

Item	Description
`-c`	Specifies that the CIFS mount be listed in colon delimited format.
`-l`	Specifies that the CIFS mount be listed in standard format with each field separated by whitespace. This is the default.
`-p`	Specifies that the CIFS mount be listed in pipe delimited format.

Parameters

Item	Description
FileSystem	Specifies which file system to list the characteristics of. The default is to list all CIFS file systems.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Examples

1. To list all CIFS mounts defined in `/etc/filesystems`, enter:

```
lscifsmnt
```

Location

`/usr/sbin/lscifsmnt`

lsclass Command

Purpose

List Workload Management classes and their limits.

Syntax

lsclass [**-C** | **-D** | **-f**] [**-r**] [**-d** *Config_Dir*] [**-S** *SuperClass*] [*Class*]

Description

The **lsclass** command, with no argument, returns the list of superclasses, one per line. With a class name as argument, it prints the class. The subclasses can be displayed with the **-r** (recursive) flag, or with the **-S** *Superclass* flag.

When WLM is started, if an empty string is passed as the name of the configuration with the **-d** flag, **lsclass** lists the classes defined in the in-core WLM data structures.

The **lsclass** command does not require any special level of privilege and is accessible for all users.

Note: If this command is given a set of time-based configurations (either specified with the **-d** flag, or because the current configuration is a set), the **lsclass** command returns the classes of the regular configuration which applies (or would apply) at the time the command is issued.

Flags

Item	Description
-C	Displays the class attributes and limits in colon-separated records, as follows: `lsclass -C myclass` `#name:description:tier:inheritance:authuser:authgroup:` `adminuser:admingroup:rset:CPUshares:CPUmin:` `CPUsoftmax:CPUhardmax:memoryshares:memorymin:` `memorysoftmax:memoryhardmax:diskIOshares:diskIOmin:` `diskIOsoftmax:diskIOhardmax:totalCPUhardmax:` `totalCPUunit:totalDiskIOhardmax:totalDiskIOunit:` `totalConnecttimehardmax:totalConnecttimeunit:` `totalProcesseshardmax:totalThreadshardmax:` `totalLoginshardmax:` `classRealMem:classRealMemunit:classVirtMem:` `classVirtMemunit:classLargePages:classLargePagesunit:` `procVirtMem:procVirtMemunit:localshm:vmenforce:delshm` `myclass::0:no:::::::-:0:100:100:-:1:100:100:-:0:100:` `100:-:s:-:KB:-:s:-:-:-:-:KB:-:KB:-:KB:-:KB:no:proc:no`
-d *Config_Dir*	Use **/etc/wlm/***Config_Dir* as alternate directory for the definition files. If an empty string is passed (for example, `-d ""`), **lsclass** lists the classes defined in the in-core WLM data structures. If this flag is not present, the current configuration files in the directory pointed to by **/etc/wlm/current** are used.
-D	Displays the default values for the class attributes and limits in colon-separated records. Any other flag or argument used in conjunction with **-D** is ignored. For example: `lsclass -D` `#name:description:tier:inheritance:authuser:` `authgroup:adminuser:admingroup:rset:CPUshares:CPUmin:` `CPUsoftmax:CPUhardmax:memoryshares:memorymin:` `memorysoftmax:memoryhardmax:diskIOshares:diskIOmin:` `diskIOsoftmax:diskIOhardmax:totalCPUhardmax:` `totalCPUunit:totalDiskIOhardmax:totalDiskIOunit:` `totalConnecttimehardmax:totalConnecttimeunit:` `totalProcesseshardmax:totalThreadshardmax:totalLoginshardmax:` `classRealMem:classRealMemunit:classVirtMem:` `classVirtMemunit:classLargePages:classLargePagesunit:` `procVirtMem:procVirtMemunit:localshm:vmenforce:delshm` `::0:no:::::::-:0:100:100:-:0:100:100:-:0:100:100:-:s:-:` `KB:-:s:-:-:-:-:KB:-:KB:-:KB:-:KB:no:proc:no`
-f	Displays the output in stanzas, with each stanza identified by a class name. Each *Attribute***=***Value* pair is listed on a separate line: `Class:` `attribute1=value` `attribute2=value` `attribute3=value`
-r	Displays, recursively, the superclasses with all their subclasses. When specifying **-r**: • If *Class* is not specified, **lsclass** shows all the superclasses with all their subclasses. • If the name of a superclass is specified, **lsclass** displays the superclass with all its subclasses. • If the name of a subclass is specified, **-r** is ineffective (displays only the subclass).
-S *SuperClass*	Restricts the scope of the command to the subclasses of the specified superclass. Only subclasses are shown with the **-S** flag.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
classes	Contains the names and definitions of the classes.
limits	Contains the limits enforced on the classes.
shares	Contains the resource shares attributes for each class.

lscluster Command

Purpose

Lists the cluster configuration information.

Syntax

lscluster { **-i** | **-g** [**-k**] | **-d** | **-c** [**-n** *clustername*] } | { **-m** [*nodename*] | **-s** | **-i** *interfacename* | **-g** *interfacename* | **-d** *diskname* }

Description

The **lscluster** command displays the attributes associated with the cluster and the cluster configuration.

Flags

Item	Description
-c	Lists the cluster configuration.
-d	Lists the cluster storage interfaces.
-g	Lists the currently active network gateway interfaces as reported by each node.
-i	Lists the network device driver (NDD) and pseudo NDD interfaces that are currently configured on each of the Cluster Aware AIX (CAA) nodes. CAA might not use all of the interfaces to exchange heartbeat packets. **Note:** The storage framework communication (`sfwcom`) interface is displayed as UP only if this interface is configured and available. Otherwise, it is not displayed.
-k	Lists the assigned network gateway interfaces for each node. The interfaces can be down or not even configured. You can specify this flag only with the **-g** flag.
-m	Lists the cluster node configuration information. This information includes a list of points of contact. *Points of contact* are cluster configuration interfaces that are used by the cluster to exchange heartbeat packets. If a point of contact has no CAA traffic for an extended period, it is removed from the list of points of contact.
-n	Allows the cluster names to be queried for all interfaces, storage, or cluster configurations (applicable only with **-i**, **-d**, or **-c** flags).
-s	Lists the cluster network statistics on the local node.

Examples

1. To list the cluster configuration for all nodes, enter:

```
lscluster -m
```

   The sample of the output follows:

```
# lscluster -m
Calling node query for all nodes...
Node query number of nodes examined: 2

                    Node name: nodeA.ibm.com
                    Cluster shorthand id for node: 1
                    uuid for node: 84088524-b124-11e3-8210-32c8e74b1e02
                    State of node:  UP  NODE_LOCAL
                    Smoothed rtt to node: 0
                    Mean Deviation in network rtt to node: 0
                    Number of clusters node is a member in: 1
                    CLUSTER NAME       TYPE  SHID   UUID
                    Sample local        84ee37f4-b124-11e3-8210-32c8e74b1e02

                    Number of points_of_contact for node: 0
                    Point-of-contact interface & contact state
                     n/a

            ------------------------------

                    Node name: nodeB.ibm.com
                    Cluster shorthand id for node: 2
                    uuid for node: 8492a5a6-b124-11e3-8210-32c8e74b1e02
                    State of node:  UP
                    Smoothed rtt to node: 70
                    Mean Deviation in network rtt to node: 82
                    Number of clusters node is a member in: 1
                    CLUSTER NAME       TYPE  SHID   UUID
                    Sample local        84ee37f4-b124-11e3-8210-32c8e74b1e02

                    Number of points_of_contact for node: 2
                    Point-of-contact interface & contact state
                    dpcom  UP  RESTRICTED
                    en0  UP
```

2. To list the cluster configuration for the local node, enter:

```
lscluster -s
```

   The sample of the output follows:

```
 # lscluster -s
            Cluster Network Statistics:

            pkts seen: 33861217                  passed: 32052241
            IP pkts: 5778096                     UDP pkts: 1934943
            gossip pkts sent: 1463320            gossip pkts recv: 688759
            cluster address pkts: 0              CP pkts: 1808962
            bad transmits: 5                     bad posts: 4
            Bad transmit (overflow - disk ): 0
            Bad transmit (overflow - tcpsock): 0
            Bad transmit (host unreachable): 0
            Bad transmit (net unreachable): 0
            Bad transmit (network down): 0
            Bad transmit (no connection): 0
            short pkts: 0                        multicast pkts: 1808880
            cluster wide errors: 0               bad pkts: 0
            dup pkts: 0                          dropped pkts: 14
            pkt fragments: 1                     fragments queued: 0
            fragments freed: 0
            pkts pulled: 0                       no memory: 0
            rxmit requests recv: 10              requests found: 3
            requests missed: 7                   ooo pkts: 0
            requests reset sent: 7               reset recv: 0
            remote tcpsock send: 0               tcpsock recv: 0
            rxmit requests sent: 0
            alive pkts sent: 0                   alive pkts recv: 0
            ahafs pkts sent: 2                   ahafs pkts recv: 0
            nodedown pkts sent: 0                nodedown pkts recv: 1
```

```
            socket pkts sent: 62                    socket pkts recv: 54
            cwide pkts sent: 275321                 cwide pkts recv: 275318
            socket pkts no space: 0                 pkts recv notforhere: 0
            Pseudo socket pkts sent: 0              Pseudo socket pkts recv: 0
            Pseudo socket pkts dropped: 0
            arp pkts sent: 1                        arp pkts recv: 2
            stale pkts recv: 0                      other cluster pkts: 4
            storage pkts sent: 1                    storage pkts recv: 1
            disk pkts sent: 174                     disk pkts recv: 0
            unicast pkts sent: 275364               unicast pkts recv: 82
            out-of-range pkts recv: 0
            IPv6 pkts sent: 0                       IPv6 pkts recv: 122
            IPv6 frags sent: 0                      IPv6 frags recv: 0
            Unhandled large pkts: 0
            mrxmit overflow     : 0                 urxmit overflow: 0
```

3. To list the interface information for the local node, enter:

```
lscluster -i
```

The sample of output follows:

```
# lscluster -i
            Network/Storage Interface Query

            Cluster Name:  Sample
            Cluster uuid:  84ee37f4-b124-11e3-8210-32c8e74b1e02
            Number of nodes reporting = 2
            Number of nodes expected = 2

            Node nodeA.ibm.com
            Node uuid = 84088524-b124-11e3-8210-32c8e74b1e02
            Number of interfaces discovered = 2
                    Interface number 1 en0
                            ifnet type = 6 ndd type = 7
                            Mac address length = 6
                            Mac address =  32:C8:E7:4B:1E:02
                            Smoothed rrt across interface = 0
                            Mean Deviation in network rrt across interface = 0
                            Probe interval for interface = 100 ms
                            ifnet flags for interface = 0x1E080863
                            ndd flags for interface = 0x0021081B
                            Interface state  UP
                            Number of regular addresses configured on interface = 1
                            IPv4 ADDRESS: 9.3.199.216  broadcast 9.3.199.255  netmask
255.255.254.0
                            Number of cluster multicast addresses configured on interface = 1
                            IPv4 MULTICAST ADDRESS: 228.3.199.216  broadcast 0.0.0.0
netmask 0.0.0.0
                    Interface number 2 dpcom
                            ifnet type = 0 ndd type = 305
                            Mac address length = 0
                            Mac address =  00:00:00:00:00:00
                            Smoothed rrt across interface = 750
                            Mean Deviation in network rrt across interface = 1500
                            Probe interval for interface = 22500 ms
                            ifnet flags for interface = 0x00000000
                            ndd flags for interface = 0x00000009
                            Interface state  UP  RESTRICTED  AIX_CONTROLLED
                    Pseudo Interface
                            Interface State DOWN

            Node nodeB.ibm.com
            Node uuid = 8492a5a6-b124-11e3-8210-32c8e74b1e02
            Number of interfaces discovered = 2
                    Interface number 1 en0
                            ifnet type = 6 ndd type = 7
                            Mac address length = 6
                            Mac address =  32:C8:EF:AD:7C:02
                            Smoothed rrt across interface = 0
                            Mean Deviation in network rrt across interface = 0
                            Probe interval for interface = 990 ms
                            ifnet flags for interface = 0x1E084863
                            ndd flags for interface = 0x0021081B
                            Interface state  UP
                            Number of regular addresses configured on interface = 1
                            IPv4 ADDRESS: 9.3.199.128  broadcast 9.3.199.255  netmask
255.255.254.0
                            Number of cluster multicast addresses configured on interface = 1
                            IPv4 MULTICAST ADDRESS: 228.3.199.216  broadcast 0.0.0.0
```

```
netmask 0.0.0.0
                    Interface number 2 dpcom
                            ifnet type = 0 ndd type = 305
                            Mac address length = 0
                            Mac address =  00:00:00:00:00:00
                            Smoothed rrt across interface = 750
                            Mean Deviation in network rrt across interface = 1500
                            Probe interval for interface = 22500 ms
                            ifnet flags for interface = 0x00000000
                            ndd flags for interface = 0x00000009
                            Interface state  UP  RESTRICTED  AIX_CONTROLLED
                    Pseudo Interface
                            Interface State DOWN
```

4. To list the storage interface information for the cluster, enter:

```
lscluster -d
```

The sample of output follows:

```
# lscluster -d
            Storage Interface Query

            Cluster Name:  Sample
            Cluster uuid:  84ee37f4-b124-11e3-8210-32c8e74b1e02
            Number of nodes reporting = 2
            Number of nodes expected = 2
            Node nodeA.ibm.com
            Node uuid = 84088524-b124-11e3-8210-32c8e74b1e02
            Number of disk discovered = 1
                    hdisk4
                      state : UP
                      uDid  :
                      uUid  : 76c94719-7335-ded6-10e2-77d61ff7998c
                      type  : REPDISK
            Node nodeB.ibm.com
            Node uuid = 8492a5a6-b124-11e3-8210-32c8e74b1e02
            Number of disk discovered = 1
                    hdisk0
                      state : UP
                      uDid  : 382300c4f4f700004c0000000140799c6e39.3105VDASD03AIXvscsi
                      uUid  : 76c94719-7335-ded6-10e2-77d61ff7998c
                      type  : REPDISK
```

5. To list the cluster configuration, enter:

```
lscluster -c
```

The sample of the output follows:

```
# lscluster -c
Cluster Name: Sample
Cluste UUID: 8e1d89da-b39d-11e3-91e7-d24dc2d9d309
Number of nodes in cluster = 2
        Cluster ID for node nodeA.ibm.com: 1
        Primary IP address for node r5r3m25.aus.stglabs.ibm.com: 9.3.207.132
        Cluster ID for node nodeB.ibm.com: 2
        Primary IP address for node r5r3m26.aus.stglabs.ibm.com: 9.3.207.218
Number of disks in cluster = 1
        Disk = hdisk6 UUID = 57208624-fda4-d404-a7c0-8e425e2941a4 cluster_major = 0
cluster_minor = 1
Multicast for site LOCAL: IPv4 228.3.207.132 IPv6 ff05::e403:cf84
Communication Mode: multicast
Local node maximum capabilities: HNAME_CHG, UNICAST, IPV6, SITE
Effective cluster-wide capabilities: HNAME_CHG, UNICAST, IPV6, SI
```

6. To list the currently assigned network gateway interfaces for each node, enter the following command:

```
lscluster -g -k
```

The sample of the output follows:

```
# lscluster -g -k
Configured Network Gateway Interfaces

Cluster Name: Sample
```

```
Cluster UUID: 6b535f42-b3d3-11e9-8004-00145e742bf8
Number of nodes found = 2

Node node1.ibm.com
Node SHID = 1
Node UUID = 6b2d5f36-b3d3-11e9-8004-00145e742bf8
        Number of network gateways Configured = 1
                IPv4 ADDRESS: 9.3.xxx.xxx

Node node2.ibm.com
Node SHID = 2
Node UUID = 3f5c5970-b789-11e9-800a-00145e742bf8
        Number of network gateways Configured = 2
                IPv4 ADDRESS: 9.3.xxx.xxx
                IPv4 ADDRESS: 12.12.12.12
```

In the output, the IP address, 12.12.12.12, is not active but is displayed because of the **-k** flag.

lscomg Command

Purpose

Displays information about the communication groups of a peer domain.

Syntax

`lscomg` [`-l` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-i`] [`-h`] [`-TV`] [*communication_group*]

Description

The `lscomg` command displays information about the communication groups that are defined to the online peer domain on which the command runs. If you specify the name of a communication group, the `lscomg` command displays information about that communication group only.

Some of the communication group information that is displayed follows:

Field	Description
Name	The name of the communication group
Sensitivity	The number of missed heartbeats that constitute a failure
Period	The number of seconds between heartbeats
Priority	The relative priority of the communication group
Broadcast	Indicates whether broadcast should be used if it is supported by the underlying media
SourceRouting	Indicates whether source routing should be used if it is supported by the underlying media
NIMPath	The path to the Network Interface Module (NIM) that supports the adapter types in the communication group
NIMParameters	The NIM start parameters

Interface resources

Use the `-i` flag to display information about the interface resources that refer to *communication_group*.

For IP communication groups (`MediaType` = 1), `lscomg -i` displays the following information:

Field	Description
Name	The name of the interface resource that refers to *communication_group*.

Field	Description
NodeName	The host name of the interface resource that refers to *communication_group*.
IPAddress	The IP address of the interface resource that refers to *communication_group*.
SubnetMask	The subnet mask of the interface resource that refers to *communication_group*.
Subnet	The subnet of the interface resource that refers to *communication_group*

For disk heartbeating (`MediaType = 2`) and other non-IP types of communication groups (`MediaType = 0`), `lscomg -i` displays the following information:

Field	Description
Name	The name of the interface resource that refers to *communication_group*.
NodeName	The host name of the interface resource that refers to *communication_group*.
DeviceInfo	Information about the device.
MediaType	The type of interfaces that make up this communication group.

Flags

-l
Displays the information on separate lines (long format).

-t
Displays the information in separate columns (table format). This is the default format.

-d
Displays the information using delimiters. The default delimiter is a colon (`:`). Use the `-D` flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default colon (`:`) — when the information you want to display contains colons, for example. You can use this flag to specify a delimiter of one or more characters.

-x
Excludes the header (suppresses header printing).

-i
Displays information about the interface resource that refers to *communication_group*.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

communication_group
Specifies the name of the communication group about which you want to display information. You can specify a communication group name or a substring of a communication group name for this parameter. If you specify a substring, the command displays information about any defined communication group with a name that contains the substring.

Security

The user of the `lscomg` command needs read permission for the `IBM.CommunicationGroup` resource class. Read permission for the `IBM.NetworkInterface` resource class is required to display the

network interface information. By default, `root` on any node in the peer domain has read and write access to these resource classes through the configuration resource manager.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

6
The communication group definition does not exist.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is defined and online to the peer domain on which the communication group exists.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the `-f "-"` or `-F "-"` flag is specified, this command reads one or more node names from standard input.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

In these examples, nodeA is defined and online to peer domain `ApplDomain`.

1. To display general information about the communication groups for `ApplDomain`, run this command on `nodeA`:

   ```
   lscomg
   ```

   The following output is displayed:

```
Name  Sensitivity  Period  Priority  Broadcast  SourceRouting  NIMPath
NIMParameters
ComG1      2          2        1        no          yes        /opt/rsct/bin/hats_nim      -l 5
```

2. To display information about the interface resources that refer to the communication group `ComGrp1` for the peer domain `ApplDomain`, run this command on `nodeA`:

   ```
   lscomg -i ComGrp1
   ```

   The following output is displayed:

   ```
   Name    NodeName    IPAddr          SubnetMask       Subnet
   eth0    n24         9.234.32.45     255.255.255.2    9.235.345.34
   eth0    n25         9.234.32.46     255.255.255.2    9.235.345.34
   ```

Location

/opt/rsct/bin/lscomg

lscondition Command

Purpose

Lists information about one or more conditions.

Syntax

`lscondition` [`-a`] [`-m` | `-n` | `-e`] [`-C` | `-l` | `-t` | `-d` | `-D` *delimiter*] [`-A`] [`-q`] [`-U`] [`-x`] [`-h`] [`-TV`] [*condition1* [`,`*condition2*`,`...]`:`*node_name*]

Description

The `lscondition` command lists the following information about defined conditions:

Field	Description
Name	The name of the condition.
Node	The location of the condition (for management domain scope or peer domain scope).
MonitorStatus	The status of the condition.
ResourceClass	The resource class that is monitored by this condition.
EventExpression	The expression that is used in monitoring this condition.
EventDescription	A description of the `EventExpression` field.
RearmExpression	The expression used in determining when monitoring should restart for this condition after an event has occurred.
RearmDescription	A description of the `RearmExpression` field.
SelectionString	The selection string that is applied to the attributes of `ResourceClass` to determine which resources are included in the monitoring of this condition.

Field	Description
Severity	The severity of the condition: critical, warning, or informational.
NodeNames	The host names of the nodes where the condition is registered.
MgtScope	The RMC scope in which the condition is monitored.
Toggle	Specifies whether the condition toggles between the event and the rearm event.
Locked	Specifies whether the resource is locked or unlocked.
EventBatchingInterval	Specifies the time in seconds that is used to determine when the accumulated events are batched together and sent to the response. A value of 0 indicates that no batching is used.
EventBatchingMaxEvents	Specifies the maximum number of events that can be in a single batch of events. A value of 0 indicates that there is no maximum if the value of **EventBatchingInterval** is not 0.
BatchedEventRetentionPeriod	Specifies the time in hours that the batched event file is kept after all associated response scripts are run.
BatchedEventMaxTotalSize	Specifies that the total saved batched event file size can't exceed a certain size in megabytes (MB) per condition. RecordAuditLog Specifies the level of detail for ERRM log entries to the audit log (ALL, Error Only, or None).

For a list of all conditions, enter the `lscondition` command without any condition names specified. A list of all the condition names is returned with the monitoring status for each condition. The default format in this case is tabular. Specifying a node name following the condition names limits the display to the conditions defined on that node. You can list all of the conditions on a node by specifying a colon (`:`) followed by the node name. The node name is a node within the management scope, which is determined by the CT_MANAGEMENT_SCOPE environment variable. The management scope determines the list of nodes from which the conditions are listed. For local scope, only conditions on the local node are listed. Otherwise, the conditions from all nodes within the domain are listed.

For all of the information about all condition names, specify the `-A` flag with the `lscondition` command. The `-A` flag causes all information about a condition to be listed when no condition names are specified. When all the information about all conditions is listed, the default format is long. If a monitoring-status flag (`-e`, `-m`, or `-n`) is specified, the conditions with that status are listed.

When more than one condition is specified, the condition information is listed in the order in which the condition names are entered.

By default, when a condition name is specified with the `lscondition` command, all of the condition's attributes are displayed.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node.

Flags

-a

Specifies that this command applies to all nodes in the cluster. The cluster scope is determined by the **CT_MANAGEMENT_SCOPE** environment variable. If it is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management and peer domain exist, `lscondition -a` with **CT_MANAGEMENT_SCOPE** not set will list the management domain. In this case, to list the peer domain, set **CT_MANAGEMENT_SCOPE** to 2.

-A
Displays all of the attributes of the condition.

-C
Displays a `mkcondition` command template based on the condition. By modifying this template, you can create new conditions. If more than one condition is specified, the template for each `mkcondition` command appears on a separate line. This flag is ignored when no conditions are specified. This flag overrides the `–l` flag.

-d
Produces delimiter-formatted output. The default delimiter is a colon (`:`). Use the `–D` flag if you want to change the default delimiter.

-D *delimiter*
Produces delimiter-formatted output that uses the specified delimiter. Use this flag to specify something other than the default, colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

-e
Lists only those conditions that are monitored in error.

-l
Produces long-formatted output. Displays the condition information on separate lines.

-m
Lists only those conditions that are being monitored without error.

-n
Lists only those conditions that are not being monitored.

-q
Does not return an error when the condition does not exist.

-t
Displays the condition information in separate columns (table format).

-U
Indicates whether the resource is locked.

-x
Suppresses header printing.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

***condition1* [,*condition2*,...]**
Specifies the name of an existing condition that is defined on the host name *node_name*. You can specify more than one condition name. This parameter can be a condition name or a substring of a condition name. When it is a substring, any defined condition name that contains the substring will be listed.

node_name
Specifies the node where the condition is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the **CT_MANAGEMENT_SCOPE** environment variable.

Security

The user needs read permission for the `IBM.Condition` resource class to run `lscondition`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To list all conditions and their monitoring status, run this command:

```
lscondition
```

   The output will look like this:

```
 Name                       Node          MonitorStatus
"FileSystem space used"   "nodeA"       "Monitored"
"tmp space used"          "nodeA"       "Not monitored"
"var space used"          "nodeA"       "Error"
```

2. To list general information about the condition "FileSystem space used" in long form, run this command:

```
lscondition "FileSystem space used"
```

   The output will look like this:

```
Name             = "FileSystem space used"
Node             = "nodeA"
MonitorStatus    = "Monitored"
ResourceClass    = "IBM.FileSystem"
EventExpression  = "PercentTotUsed > 99"
EventDescription = "Generate event when space used is
                    greater than 99 percent full"
RearmExpression  = "PercentTotUsed < 85"
RearmDescription = "Start monitoring again after it is
                    less than 85 percent"
SelectionString  = ""
Severity         = "w"
NodeNames        = "{}"
MgtScope         = "l"
Toggle           = "Yes"
Locked           = "No"
```

3. To list the command that would create the condition "FileSystem space used", run this command:

```
lscondition -C "FileSystem space used"
```

   The output will look like this:

```
mkcondition -r IBM.FileSystem -a PercentTotUsed \
-e "PercentTotUsed > 99" -E "PercentTotUsed < 85" \
-d "Generate event when space used is greater than 99 percent full" \
-D "Start monitoring after it is less than 85 percent" \
-S w "FileSystem space used"
```

4. To list all conditions that have the string space in their names, run this command:

```
lscondition space
```

   The output will look like this:

```
Name             = "FileSystem space used"
MonitorStatus    = "Monitored"

Name             = "tmp space used"
```

```
MonitorStatus    = "Not Monitored"

Name             = "var space used"
MonitorStatus    = "Monitored"
```

5. To list the conditions that are in error, run this command:

```
lscondition -e
```

The output will look like this:

```
Name               MonitorStatus
"var space used"   "Error"
```

This example applies to clustered systems:

1. To list all conditions and their monitoring status, run this command:

```
lscondition -a
```

The output will look like this:

```
Name                       Node        MonitorStatus
"FileSystem space used"    "nodeA"     "Monitored"
"tmp space used"           "nodeB"     "Not monitored"
"var space used"           "nodeC"     "Error"
```

Location

/opt/rsct/bin/lscondition

lscondresp Command

Purpose

Lists information about a condition and any of its condition/response associations.

Syntax

To list the link between a condition and one or more responses:

`lscondresp` [`-a` | `-n`] [`-l` | `-t` | `-d` | `-D` *delimiter*] [`-q`] [`-U`] [`-x`] [`-z`] [`-h`] [`-TV`] [*condition*[`:`*node_name*] [*response1* [*response2*...]]]

To list all of the links to one or more responses:

`lscondresp` [`-a` | `-n`] [`-l` | `-t` | `-d` | `-D` *delimiter*] [`-q`] [`-x`] [`-z`] `-r` [`-U`] [`-h`] [`-TV`] *response1*[`:`*node_name*] [*response2*...]

Description

The `lscondresp` command lists information about a condition and its linked responses. A link between a condition and a response is called a *condition/response association.* The information shows which responses are linked with a condition and whether monitoring is active for a condition and its linked response. The following information is listed:

Field	Description
Condition	The name of the condition linked with a response.
Response	The name of the response linked with the condition.

Field	Description
State	The state of the response for the condition. The state indicates whether a specified response is active or not.
Node	The location of the condition and the response.
Locked	Indicates whether the resource is locked or unlocked.

To list a particular condition and response, specify both the condition and the response. To list all responses to a condition, specify the condition only. To list all conditions to which a response is linked, specify the response and the `-r` flag. To list all conditions and their linked responses, do not specify any condition or response parameters.

Specifying a node name limits the display to the condition/response associations that are defined on that node. List all of the condition/response associations on a node by specifying a colon (`:`) followed by the node name. The node name is a node within the management scope determined by the CT_MANAGEMENT_SCOPE environment variable. The management scope determines the list of nodes from which the condition/response associations are listed. For local scope, only condition/response associations on the local node are listed. For management domain scope and peer domain scope, the condition/response associations from all nodes within the domain are listed.

When neither the `-a` flag nor the `-n` flag is specified, all selected conditions for the responses are listed. Tabular format is the default.

Flags

-a
Lists only those responses that are active for the condition.

-n
Lists only those responses that are not active for the condition.

-l
Displays the condition information and response information on separate lines (long format).

-t
Displays the condition information and response information in separate columns (table format).

-d
Specifies delimiter-formatted output. The default delimiter is a colon (`:`). Use the `-D` flag if you want to change the default delimiter.

-D *delimiter*
Specifies delimiter-formatted output that uses *delimiter*. Use this flag to specify something other than the default colon (`:`). For example, when the data to be displayed contains colons, use this flag to specify another delimiter of one or more characters.

-q
Does not return an error if either the *condition* or the *response* does not exist.

-U
Indicates whether the resource is locked.

-x
Suppresses header printing.

-z
Specifies that this command applies to all nodes in the cluster. The cluster scope is determined by the CT_MANAGEMENT_SCOPE environment variable. If it is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management and peer domain exist, `lscondresp -z` with CT_MANAGEMENT_SCOPE not set will list the management domain. In this case, to list the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-r
Lists information about all of the condition/response associations for the specified responses. Use this flag to indicate that all command parameters specified are responses, not conditions.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

condition
The *condition* can be a condition name or a substring of a condition name. When it is a substring, any defined condition name that contains the substring and is linked to the response will be listed.

***response1* [*response2*...]**
This parameter can be a response name or a substring of a response name. You can specify more than one response name. When it is a substring, any defined response name that contains the substring and is linked to the condition will be listed.

node_name
Specifies the node where the condition or response is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

Security

The user needs read permission for the `IBM.Association` resource class to run `lscondresp`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

To see which resources are locked, run this command:

```
lscondresp -U
```

The output will look like this:

```
Condition             Response                 Node     State         Locked
"/tmp space used"     "E-mail root off-shift"  "nodeA"  "Not active"  "Yes"
"Page space in rate"  "E-mail root anytime"    "nodeA"  "Not active"  "No"
```

These examples apply to standalone systems:

1. To list all conditions with their linked responses, run this command:

   ```
   lscondresp
   ```

   The output will look like this:

   ```
   Condition                Response                       Node       State
   "FileSystem space used"  "Broadcast event on-shift"    "nodeA"    "Active"
   "FileSystem space used   "E-mail root anytime"         "nodeA"    "Not Active"
   "Page in Rate"           "Log event anytime"           "nodeA"    "Active"
   ```

2. To list information about the condition "FileSystem space used", run this command:

```
lscondresp "FileSystem space used"
```

The output will look like this:

```
Condition                Response                       Node       State
"FileSystem space used"  "Broadcast event on-shift"    "nodeA"    "Active"
"FileSystem space used   "E-mail root anytime"         "nodeA"    "Not Active"
```

3. To list information about the condition "FileSystem space used" for responses that are active, run this command:

```
lscondresp -a  "FileSystem space used"
```

The output will look like this:

```
Condition                Response                       Node       State
"FileSystem space used"  "Broadcast event on-shift"    "nodeA"    "Active"
```

4. To list information about the condition "FileSystem space used" with the linked response "Broadcast event on-shift", run this command:

```
lscondresp  "FileSystem space used" "Broadcast event on-
shift"
```

The output will look like this:

```
Condition                Response                       Node       State
"FileSystem space used"  "Broadcast event on-shift"    "nodeA"    "Active"
```

5. To list all conditions that have the string `space` in their names with their linked responses, run this command:

```
lscondresp space
```

The output will look like this:

```
Condition                Response                       Node       State
"FileSystem space used"  "Broadcast event on-shift"    "nodeA"    "Active"
"FileSystem space used   "E-mail root anytime"         "nodeA"    "Not Active"
```

These examples apply to management domains:

1. In this example, the condition "FileSystem space used" is defined on the management server. To list information about "FileSystem space used", run this command on the management server:

```
lscondresp "FileSystem space used"
```

The output will look like this:

```
Condition                 Response                       Node       State
"FileSystem space used"   "Broadcast event on-shift"    "nodeB"    "Active"
"FileSystem space used"   "E-mail root anytime"         "nodeB"    "Not Active"
```

2. In this example, the condition "FileSystem space used" is defined on the managed node `nodeC`. To list information about "FileSystem space used", run this command on the management server:

```
lscondresp "FileSystem space used":nodeC
```

The output will look like this:

```
Condition                 Response                       Node     State
"FileSystem space used"   "Broadcast event on-shift"   "nodeC"    "Active"
```

```
"FileSystem space used"    "E-mail root anytime"        "nodeC"    "Not Active"
```

This example applies to a peer domain:

1. In this example, the condition "FileSystem space used" is defined in the domain. To list information about "FileSystem space used", run this command on one of the nodes in the domain:

   ```
   lscondresp "FileSystem space used"
   ```

   The output will look like this:

   ```
   Condition                 Response                      Node       State
   "FileSystem space used"   "Broadcast event on-shift"    "nodeD"    "Active"
   "FileSystem space used"   "E-mail root anytime"         "nodeD"    "Not Active"
   "FileSystem space used"   "Broadcast event on-shift"    "nodeE"    "Active"
   "FileSystem space used"   "E-mail root anytime"         "nodeE"    "Not Active"
   ```

Location

/opt/rsct/bin/lscondresp

lsconn Command

Purpose

Displays the connections a given device, or kind of device, can accept.

Syntax

lsconn { **-p** *ParentName* | [**-c** *ParentClass*] [**-s** *ParentSubclass*] [**-t** *ParentType*] } { **-l** *ChildName* | **-k** *ChildConnectionKey* } [**-f** *File*] [**-F** *Format*] [**-h**] [**-H**]

Description

The **lsconn** command, when used with the **-p** *ParentName* flag, displays the connection locations on the parent device to which the device specified by the **-l** *ChildName* flag can be connected, or to which devices of the connection type specified by the **-k** *ChildConnectionKey* flag can be connected. If the **-k** and **-l** flags are not used, the **lsconn** command displays information about where a child device can be connected on the specified parent.

If the **-p** *ParentName* flag is not used, you must use a combination of one or all of the **-c** *ParentClass*, **-s** *ParentSubclass*, and **-t** *ParentType* flags to uniquely identify the predefined parent device.

You can display the default output, which is the connection location (or connection location and connection key if no child is specified), from the Predefined Connection object class. If you do not display the default, you can display the output in a user-specified format where the *Format* parameter is a quoted list of column names separated by nonalphanumeric characters or white space using the **-F** *Format* flag. You can insert headers above the columns using the **-H** flag.

Use the flags either on the command line or in the specified **-f** *File* flag.

Flags

Item	Description
-c *ParentClass*	Specifies the class name of a possible parent device in the Predefined Devices object class. This flag cannot be used with the **-p** flag.
-f *File*	Reads the necessary flags from the *File* parameter.

Item	Description
-F *Format*	Formats the output in a user-specified format, where the *Format* parameter is a quoted list of column names from the Predefined Connection object class separated, and possibly terminated, by non-alphanumeric characters or white space. If white space id used as the separator, the **lsconn** command displays the output in aligned columns.
-H	Displays headers above the column output.
-h	Displays the command usage message.
-k *ChildConnectionKey*	Specifies the connection key that identifies the subclass of the child device. This flag cannot be used with the **-l** flag.
-l *ChildName*	Specifies the logical name of a possible child device. This flag cannot be used with the **-k** flag.
-p *ParentName*	Specifies the parent device's logical name from the Customized Devices object class. This flag cannot be used with the **-c**, **-s**, or **-t** flag.
-s *ParentSubclass*	Specifies the subclass of a possible parent device in the Predefined Devices object class. This flag cannot be used with the **-p** flag.
-t *ParentType*	Specifies the device type of a possible parent device from the Predefined Devices object class. This flag cannot be used with the **-p** flag.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list all of the possible connection locations on the sa2 IBM 8-Port EIA-232/RS-422A (PCI) Adapter that will accept an RS-232 device connection, type the following:

   ```
   lsconn -p sa2 -k rs232
   ```

   The system displays a possible connections similar to the following:

   ```
   0
   1
   2
   3
   4
   5
   6
   7
   ```

2. To list all of the possible connection locations and connection types on the sa2 IBM 8-Port EIA-232/RS-422A (PCI) Adapter, type the following:

   ```
   lsconn -p sa2
   ```

   The system displays a message similar to the following:

   ```
   0 rs232
   1 rs232
   2 rs232
   3 rs232
   ```

```
4 rs232
5 rs232
6 rs232
7 rs232
0 rs422
1 rs422
2 rs422
3 rs422
4 rs422
5 rs422
6 rs422
7 rs422
```

Files

Item	Description
/usr/sbin/lsconn	Specifies the command file.

lscons Command

Purpose

Writes the name of the current console device to standard output.

Syntax

lscons [**-s**] [**-a** | **-O**]

lscons -b [**-s**] [**-a** | **-O**]

lscons -d [**-s**]

Description

The **lscons** command writes the name of the current console device to standard output. This command is also used to write the name of the device that is to be the console on the next start of the system to standard output. You can change the current console device using the **swcons** command. You can change the device to be the system console on the next start of the system using the **chcons** command.

Flags

Item	Description
-a	Displays a list of *attribute name = attribute value* pairs for the console device and console logging and tagging attributes. When used with the **-b** flag, the values are retrieved from the ODM. Without the **-b** flag, the values are retrieved from the console device driver. **Note:** This flag is not valid with the **-O** flag or the **-d** flag.
-b	Displays the full path name of the system console selected for the next startup of the system.
-d	Displays the full path name of the system console selected on the current startup of the system. **Note:** This flag is not valid with the **-O** flag or the **-a** flag.
-O	Similar to the **-a** flag but outputs the attribute names and values in a format suitable for use by SMIT. This flag is NOT valid with the **-d** flag. **Note:** This flag is not valid with the **-d** flag or the **-a** flag.
-s	Suppresses reporting of the path name.

Exit Status

This command returns the following exit values:

Item	Description
0	The device you are using is the current system console.
1	The device you are using is not the current system console.
2	The device you are using is the console device selected at system start but is not currently the device supporting console message output.
3	Flags specified are not valid.
4	System error occurred.

Examples

1. To display the full path name of the current system console, type:

```
lscons
```

2. To display the full path name of the system console effective on the next startup of the system, type:

```
lscons  -b
```

3. To display the full path name of the system console selected on the current startup of the system, type:

```
lscons -d
```

4. To test whether or not the current system console is directed to your display, type:

```
if lscons -s
then
echo "System messages are directed to my display" >/dev/tty
fi
```

Files

Item	Description
/usr/sbin/lscons	Contains the **lscons** command.

lscore Command

Purpose

Views the current core settings.

Syntax

`lscore` [`-R` *registry*] [*username* | `-d`]

Description

The `lscore` command will be the user interface to view the current core settings. It will have the following usage:

```
lscore [-R registry] [username|-d]
```

As with `chcore`, the `-d` flag will show the default values. Viewing settings for another user is a privileged operation; however, any user may view the default values.

Flags

Item	Description
`-d`	Changes the default setting for the system.
`-R` *registry*	Specifies the loadable I&A module.

Security

May only be run by root or another user with system authority.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Examples

1. To list the current settings for root, type:

   ```
   lscore root
   ```

   The output will look like:

   ```
   compression: on
   path specification: default
   corefile location: default
   naming specification: off
   ```

2. To list the default settings for the system, type:

   ```
   lscore -d
   ```

   The output will look like:

   ```
   compression: off
   path specification: on
   corefile location: /corefiles
   naming specification: off
   ```

lscosi Command

Purpose

Lists information related to a Common Operating System Image (COSI).

Syntax

lscosi [[**-l{1|2|3}**] . . .] [**-v**] [*COSI*]

Description

The `lscosi` command lists the status and detailed information related to a Common Operating System Image (COSI). The level of information to be listed depends on the numeric value specified by the `-l` flag, with a level ranging from 1 - 3 (3 being the most detailed). If a level is not specified, a default of level 1 information is displayed. If no argument is specified, the `lscosi` command lists any common images that

exist in the environment. The `bos.sysmgt.nim.master` fileset must be present on the system in order for the `lscosi` command to be successful. This command can also be executed on a thin server.

Flags

Item	Description
`-l{1|2|3}`	Specifies the level of information to display. **1** This level displays very limited information related to a COSI. The information listed shows only a brief summary of the COSI and the thin servers that might be using it. **2** This level displays more than just basic information related to a COSI. The level includes information pertaining to the software content of the COSI. **3** This level displays more in-depth information related to a COSI. The level includes information pertaining to the installation log of the COSI.
`-v`	Enables verbose debug output when the `lscosi` command runs.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

Access Control: You must have root authority to run the `lscosi` command.

Examples

1. To check if any common images exist in an environment, enter:

   ```
   lscosi
   ```

   When this command is entered without an argument, it merely lists common images in the environment. The output might be similar to the following list:

   ```
   52H_0442A_cosi
   52I_0444B2_GOLD_cosi
   52L_0534A_cosi
   53E_0545A_cosi
   53D_GOLD_cosi
   53A_GOLD_cosi
   52M_0544A_cosi
   ```

2. To list brief status information for a common image named `cosi1`, enter:

   ```
   lscosi cosi1
   ```

Information similar to the following is displayed:

```
53H_0538A_spot:
        class         = resources
        type          = spot
        plat_defined  = chrp
        Rstate        = ready for use
        prev_state    = verification is being performed
        location      = /export/nimvg/spot/53H_0538A_spot/usr
        version       = 5
        release       = 2
        mod           = 0
        oslevel_r     = 5300-05
        alloc_count   = 2
        server        = master
        if_supported  = chrp.mp ent
        Rstate_result = success

Thin Server:
Client1
Client2
```

3. To list software content for a common image named `cosi1`, enter:

```
lscosi -l2 cosi1
```

Software content similar to the following is displayed from the common image:

```
Fileset                   Level  State  Type  Description
(Uninstaller)
----------------------------------------------------------
bos.64bit                5.2.0.75  C     F    Base Operating System 64 bit Runtime
bos.diag.com             5.2.0.75  C     F    Common Hardware Diagnostics
bos.diag.rte             5.2.0.75  C     F    Hardware Diagnostics
.
.
.
```

4. To list both software content and status information for a common image named `cosi1`, enter:

```
lscosi -l1 -l2 cosi1
```

Location

`/usr/sbin/lscosi`

Files

Item	Description
`/etc/niminfo`	Contains variables used by NIM.

lsdev Command

Purpose

Displays devices in the system and their characteristics.

Syntax

lsdev [**-C**][**-c** *Class*] [**-s** *Subclass*] [**-t** *Type*] [**-f** *File*] [**-F** *Format* | **-r** *ColumnName*] [**-h**] [**-H**] [**-l** { *Name* | - }] [**-p** *Parent*] [**-S** *State*] [**-x**]

lsdev -P [**-c** *Class*] [**-s** *Subclass*] [**-t** *Type*] [**-f** *File*] [**-F** *Format* | **-r** *ColumnName*] [**-h**] [**-H**] [**-x**]

Description

The **lsdev** command displays information about devices in the Device Configuration database. You can display information about all devices in the Customized Devices object class using the **-C** flag. Any combination of the **-c** *Class*, **-s** *Subclass*, **-t** *Type*, **-l** *Name*, **-p** *Parent*, and **-S** *State* flags selects a subset of the customized devices. You can display information about all devices in the Predefined Devices object class using the **-P** flag. Any combination of the **-c** *Class*, **-s** *Subclass*, and **-t** *Type* flags selects a subset of the predefined devices.

You can display the default output one of the following ways:

- From the Customized Devices object class by using the **-C** flag
- From the Predefined Devices object class by using the **-P** flag

To override these two default outputs, use the **-F** *Format* flag to display the output in a format that you specify by using the *Format* parameter. The *Format* parameter is a quoted list of column names that are separated and possibly ended by non-alphanumeric characters or white space.

The **lsdev** command shows information only about devices that are based upon information in the Customized Devices (**Cudv**) object class or the Predefined Devices (**PdDv**) object class. Other object classes (such as the Customized Path (**CuPath**) object class) are not examined. This situation means that there might be conditions where a device might not be displayed. For example, if the **-p** *Parent* flag is used, but the parent that is identified in the Customized Devices object for a device does not match the *Parent* that is specified through the **-p** flag, the device is not displayed. However, the device might have a path to the specified *Parent* that is defined in the Customized Paths object class. Use the **lspath** command to show all MPIO-capable child devices of the specified parent.

You can use the System Management Interface Tool (SMIT) **smit lsdev** fast path to change device characteristics.

Flags

Item	Description
-C	Lists information about a device that is in the Customized Devices object class. The default information that is displayed is *name*, *status*, *location*, and *description*. The **-C** flag is not required, but is maintained for compatibility reasons. The **-C** flag cannot be specified with the **-P** flag. If neither is specified, the **lsdev** command behaves as if the **-C** flag was specified.
-c *Class*	Specifies a device class name. This flag can be used to restrict output to devices in a specified class.
-f *File*	Reads the necessary flags from the *File* parameter.
-F *Format*	Displays the output in a user-specified format, where the *Format* parameter is a quoted list of column names from the Predefined or Customized Devices object class, separated and possibly ended by nonalphanumeric characters or white space. If white space is used as the separator, the **lsdev** command displays the output in aligned columns. If you specify the **-F** *Format* flag with the **-C** flag, you can specify column names from both the Customized and Predefined Devices object classes. If you specify the **-F** *Format* flag with the **-P** flag, you can specify only column names from the Predefined Devices object class. In addition to the column names, the special purpose name *description* can be used to obtain a display of device descriptions. This flag cannot be used with the **-r** *ColumnName* flag. Also, the *physloc* special purpose name can be used to display a physical location code of the device.
-H	Displays headers above the column output.
-h	Displays the command usage message.

Item	Description
-l *Name*	Specifies the device logical name from the Customized Devices object class of the device for which information is listed. The *Name* argument to the **-l** flag can contain the same wildcard characters that can be used with the **odmget** command. If the *Name* argument is a dash, names are read from STDIN. Names on STDIN must be separated by a comma, a tab, a space, or a "newline" character. Names cannot contain wildcard characters. This flag cannot be used with the **-P** flag.
-p *Parent*	Specifies the device logical name from the Customized Devices object class for the parent of devices to be displayed. The **-p** *Parent* flag can be used to show the child devices of the specified *Parent*. The *Parent* argument to the **-p** flag might contain the same wildcard characters that can be used with the **odmget** command. This flag cannot be used with the **-P** flag.
-P	Lists information about a device that is in the Predefined Devices object class. The default information that is displayed is *class*, *type*, *subclass*, and *description*. This flag cannot be used with the **-C**, **-l**, or **-S** flags.
-r *ColumnName*	Displays the set of values in a column. For example, the *ColumnName* parameter takes the value of the *Class* parameter to list all of the classes. If you specify the **-r** *ColumnName* flag with the **-C** flag, you can specify column names from both the Customized and Predefined Devices object classes. If you specify the **-r** *ColumnName* flag with the **-P** flag, you can specify only column names from the Predefined Devices object class. This flag cannot be used with the **-F** *Format* flag.
-S *State*	Lists all devices in a specified state as named by the *State* parameter. The *State* parameter can have one of the following values: • *d*, *D*, *0* or *defined* for the Defined state • *a*, *A*, *1*, or *available* for the Available state • *s*, *S*, *2*, or *stopped* for the Stopped state This flag can be used to restrict output to devices in a specified state. This flag cannot be used with the **-P** flag.
-s *Subclass*	Specifies a device subclass name. This flag can be used to restrict output to devices in a specified subclass.
-t *Type*	Specifies a device type name. This flag can be used to restrict output to devices of a specified type.
-x	Displays the exported status for devices that are exported to a Workload Partition (WPAR).

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list all of the devices in the Predefined Devices object class with column headers, type the following command:

```
lsdev -P -H
```

The system displays a message similar to the following output:

```
class          type           subclass   description
logical_volume vgtype         vgsubclass Volume group
logical_volume lvtype         lvsubclass Logical volume
lvm            lvdd           lvm        LVM Device Driver
posix_aio      posix_aio      node       Posix Asynchronous I/O
aio            aio            node       Asynchronous I/O (Legacy)
pty            pty            pty        Asynchronous Pseudo-Terminal
mouse          030102         usbif      USB mouse
keyboard       030101         usbif      USB keyboard
.
.
.
disk           540mb2         scsi       540 MB SCSI Disk Drive
disk           540mb3         scsi       540 MB SCSI Disk Drive
disk           540mb4         scsi       540 MB SCSI Disk Drive
disk           540mb5         scsi       540 MB SCSI Disk Drive
disk           730mb2         scsi       730 MB SCSI Disk Drive
disk           810mb          scsi       810 MB SCSI Disk Drive
disk           810mb2         scsi       810 MB SCSI Disk Drive
bus            pcic           pci        PCI Bus
bus            isac           pci        ISA Bus
adapter        df1000f9       pci        FC Adapter
adapter        df1000f7       pci        FC Adapter
driver         efscsi         iocb       FC SCSI I/O Controller Protocol Device
adapter        c1110358       pci        USB OHCI Adapter (c1110358)
adapter        ad100501       pci        ATA/IDE Controller Device
adapter        4f111100       pci        IBM 8-Port EIA-232/RS-422A (PCI) Adapter
adapter        ccm            pci        Name of the Common Character Mode device driver
driver         hdlc           331121b9   IBM HDLC Network Device Driver
adapter        331121b9       pci        IBM 2-Port Multiprotocol Adapter (331121b9)
adapter        2b102005       pci        GXT130P Graphics Adapter
adapter        2b101a05       pci        GXT120P Graphics Adapter
adapter        23100020       pci        IBM 10/100 Mbps Ethernet PCI Adapter (23100020)
.
.
.
if             tr             TR         Token Ring Network Interface
if             vi             VI         Virtual IP Address Network Interface
if             xt             XT         X.25 Network Interface
tcpip          inet           TCPIP      Internet Network Extension
swap           paging         nfs        NFS Swap DEVICE
drawer         media1         media      SCSI Device Drawer
drawer         scsi1          dasd       SCSI DASD Drawer
adapter        4f111b00       pci        IBM 128-Port Async (PCI) Adapter
concentrator   16c232         sync_pci   16-Port RAN EIA-232 for 128-Port Adapter
concentrator   16e232         sync_pci   16-Port Enhanced RAN EIA-232 for 128-Port Adapter
concentrator   16e422         sync_pci   16-Port Enhanced RAN RS-422 for 128-Port Adapter
if             at             AT         ATM Network Interface
adapter        14105300       pci        IBM PCI 25MBPS ATM Adapter (14105300)
```

2. To list all of the devices in the Customized Devices object class, type the following command:

```
lsdev -C
```

The system displays a message similar to the following output:

```
sys0       Available              System Object
sysplanar0 Available              System Planar
mem0       Available              Memory
L2cache0   Available              L2 Cache
proc0      Available 00-00        Processor
pci0       Available              PCI Bus
pci1       Available              PCI Bus
isa0       Available 10-58        ISA Bus
siota0     Available 01-Q1        Tablet Adapter
ppa0       Available 01-R1        CHRP IEEE1284 (ECP) Parallel Port Adapter
sa0        Available 01-S1        Standard I/O Serial Port
sa1        Available 01-S2        Standard I/O Serial Port
```

```
paud0      Available 01-Q2        Ultimedia Integrated Audio
siokma0    Available 01-K1        Keyboard/Mouse Adapter
fda0       Available 01-D1        Standard I/O Diskette Adapter
scsi0      Available 10-60        Wide/Ultra-2 SCSI I/O Controller
scsi1      Available 10-61        Wide/Ultra-2 SCSI I/O Controller
sa2        Available 10-68        IBM 8-Port EIA-232/RS-422A (PCI) Adapter
sa3        Available 10-70        IBM 8-Port EIA-232/RS-422A (PCI) Adapter
sa4        Available 10-78        IBM 8-Port EIA-232/RS-422A (PCI) Adapter
.
.
hd3        Defined                Logical volume
hd1        Defined                Logical volume
hd10opt    Defined                Logical volume
inet0      Available              Internet Network Extension
en0        Available 10-80        Standard Ethernet Network Interface
et0        Defined   10-80        IEEE 802.3 Ethernet Network Interface
lo0        Available              Loopback Network Interface
pty0       Available              Asynchronous Pseudo-Terminal
gxme0      Defined                Graphics Data Transfer Assist Subsystem
rcm0       Available              Rendering Context Manager Subsystem
aio0       Defined                Asynchronous I/O (Legacy)
posix_aio0 Defined                Posix Asynchronous I/O
tty0       Available 01-S1-00-00  Asynchronous Terminal
tty1       Available 01-S2-00-00  Asynchronous Terminal
```

3. To list the adapters that are in the Available state in the Customized Devices object class, type the following command:

```
lsdev -C -c adapter -S a
```

   The system displays a message similar to the following output:

```
sa0     Available 01-S1    Standard I/O Serial Port
sa1     Available 01-S2    Standard I/O Serial Port
siokma0 Available 01-K1    Keyboard/Mouse Adapter
fda0    Available 01-D1    Standard I/O Diskette Adapter
scsi0   Available 10-60    Wide/Fast-20 SCSI I/O Controller
fcs0    Available 10-68    FC Adapter
scsi1   Available 10-88    Wide/Ultra-2 SCSI I/O Controller
fcs1    Available 20-60    FC Adapter
sioka0  Available 01-K1-00 Keyboard Adapter
siota0  Available 01-Q1    Tablet Adapter
ppa0    Available 01-R1    CHRP IEEE1284 (ECP) Parallel Port Adapter
paud0   Available 01-Q2    Ultimedia Integrated Audio
tok0    Available 10-70    IBM PCI Tokenring Adapter (14101800)
ent0    Available 10-80    IBM 10/100 Mbps Ethernet PCI Adapter (23100020)
sioma0  Available 01-K1-01 Mouse Adapter
```

4. To list all of the tape devices in the Predefined Devices object class, type the following command:

```
lsdev -P -c tape
```

   The system displays a message similar to the following output:

```
tape ost      fcp   Other FC SCSI Tape Drive
tape scsd     fcp   FC SCSI Tape Drive
tape ost      iscsi Other iSCSI Tape Drive
tape scsd     iscsi iSCSI Tape Drive
tape 1200mb-c scsi  1.2 GB 1/4-Inch Tape Drive
tape 150mb    scsi  150 MB 1/4-Inch Tape Drive
tape 3490e    scsi  3490E Autoloading Tape Drive
tape 4mm2gb   scsi  2.0 GB 4mm Tape Drive
tape 4mm4gb   scsi  4.0 GB 4mm Tape Drive
tape 525mb    scsi  525 MB 1/4-Inch Tape Drive
tape 8mm      scsi  2.3 GB 8mm Tape Drive
tape 8mm5gb   scsi  5.0 GB 8mm Tape Drive
tape 8mm7gb   scsi  7.0 GB 8mm Tape Drive
tape 9trk     scsi  1/2-inch 9-Track Tape Drive
tape ost      scsi  Other SCSI Tape Drive
tape scsd     scsi  SCSI Tape Drive
tape 4mm2gb2  scsi  2.0 GB 4mm Tape Drive
```

5. To list the supported device classes from the Predefined Devices object class, type the following command:

```
lsdev -P -r class
```

The system displays a message similar to the following output:

```
PCM
adapter
aio
array
bus
cdrom
concentrator
container
dial
disk
diskette
drawer
driver
gxme
if
keyboard
lft
logical_volume
lpfk
lvm
memory
mouse
pdisk
planar
port
posix_aio
printer
processor
pseudo
pty
rcm
rwoptical
swap
sys
tablet
tape
tcpip
tmscsi
tty
```

6. To list the supported subclasses in the Predefined Devices object class for the **disk** class, type the following command:

```
lsdev -P -c disk -r subclass
```

The system displays a message similar to the following output:

```
dar
fcp
fdar
ide
iscsi
scraid
scsi
vscsi
```

7. To list the name, class, subclass, and type of every device in the Available state in the Customized Devices object class with column headers, type the following command:

```
lsdev -C -H -S a -F 'name class subclass type'
```

The system displays a message similar to the following output:

```
name       class     subclass type
sys0       sys       node     chrp
sysplanar0 planar    sys      sysplanar_rspc
mem0       memory    sys      totmem
L2cache0   memory    sys      L2cache_rspc
proc0      processor sys      proc_rspc
pci0       bus       chrp     pci
pci1       bus       chrp     pci
isa0       bus       pci      isac
siota0     adapter   isa_sio  isa_tablet
ppa0       adapter   isa_sio  chrp_ecp
```

```
sa0         adapter   isa_sio  pnp501
sa1         adapter   isa_sio  pnp501
paud0       adapter   isa_sio  baud4232
siokma0     adapter   isa_sio  kma_chrp
fda0        adapter   isa_sio  pnp700
scsi0       adapter   pci      sym896
scsi1       adapter   pci      sym896
sa2         adapter   pci      4f111100
sa3         adapter   pci      4f111100
sa4         adapter   pci      4f111100
ent0        adapter   pci      23100020
mg20        adapter   pci      2b102005
sa5         adapter   pci      4f111100
sioka0      adapter   kma_chrp keyboard
sioma0      adapter   kma_chrp mouse
fd0         diskette  siofd    fd
cd0         cdrom     scsi     scsd
hdisk0      disk      scsi     scsd
kbd0        keyboard  std_k    ps2
mouse0      mouse     std_m    mse_3b
lvdd        lvm       lvm      lvdd
lft0        lft       node     lft
inet0       tcpip     TCPIP    inet
en0         if        EN       en
lo0         if        LO       lo
pty0        pty       pty      pty
rcm0        rcm       node     rcm
tty0        tty       rs232    tty
tty1        tty       rs232    tty
```

8. To list the name, class, location, and physloc of all adapter devices in the Customized Devices object class with column headers, type the following command:

   ```
   lsdev -C -c adapter -F 'name class location physloc'
   ```

   The system displays a message similar to the following output:

   ```
   ent0   adapter 02-08 UTMP0.02F.00004BA-P1-C3-T1
   scsi0  adapter 01-08 UTMP0.02F.00004BA-P1-C2-T1
   scsi1  adapter 01-09 UTMP0.02F.00004BA-P1-C2-T2
   scsi2  adapter 03-08 UTMP0.02F.00004BA-P1-C4-T1
   scsi3  adapter 03-09 UTMP0.02F.00004BA-P1-C4-T2
   vsa0   adapter       U9111.520.10004BA-V4-C0
   vscsi0 adapter       U9111.520.10004BA-V4-C2
   vscsi1 adapter       U9111.520.10004BA-V4-C3
   ```

9. To list all of the children of the `pci0` bus, type the following command:

   ```
   lsdev -p pci0
   ```

   The system displays a message similar to the following output:

   ```
   ent0  Available 10-80 IBM 10/100 Mbps Ethernet PCI Adapter (23100020)
   isa0  Available 10-58 ISA Bus
   scsi0 Available 10-60 Wide/Fast-20 SCSI I/O Controller
   scsi1 Available 10-88 Wide/Ultra-2 SCSI I/O Controller
   tok0  Available 10-70 IBM PCI Tokenring Adapter (14103e00)
   ```

10. To list the devices whose names are contained in the file `/tmp/f`, type:

    ```
    cat /tmp/f | lsdev -l -
    ```

    The system displays a message similar to the following output:

    ```
    pci0 Available  PCI Bus
    scsi0 Available 10-60 Wide/Fast-20 SCSI I/O Controller
    hdisk0 Available 10-60-00-8,0 16 Bit SCSI Disk Drive
    ```

11. To display the status of the devices that are exported to a WPAR as **Exported**, enter the following command:

    ```
    # lsdev -c disk -x
    ```

    The system displays a message similar to the following output:

```
hdisk0 Available 01-08-00-1,0 16 Bit LVD SCSI Disk Drive
hdisk1 Exported  01-08-00-2,0 Other SCSI Disk Drive
```

where the hdisk1 device is exported to a WPAR.

Files

Item	Description
/usr/sbin/lsdev	Contains the **lsdev** command.

lsdisp Command

Purpose

Lists the displays available on the system.

Syntax

lsdisp [-l]

Description

The **lsdisp** command lists the displays currently available on the system, displaying a logical name of the display, a physical slot number of a display adapter, the type of bus to which a graphics display is attached, a display name and a description of each of the displays. This command also lists the default display.

Flags

Item	Description
-l	Specifies the removal of all header information and `Default display' from format.

Examples

To list all available displays, enter:

```
lsdisp
```

The following output of the **lsdisp** command lists three available displays:

```
DEV_NAME  SLOT    BUS  ADPT_NAME  DESCRIPTION

ppr0      00-01   mca  POWER_G4   Midrange Graphics Adapter
gda0      00-03   mca  colordga   Color Graphics Display Adapter
ppr1      00-04   mca  POWER_Gt3  Midrange Entry Graphics Adapter

Default display = gda0
```

Files

Item	Description
bin/lsdisp	Contains the **lsdisp** command.

lsdom Command

Purpose

Displays domain attributes.

Syntax

lsdom [**-C**] [**-f**] [**-a** *Attr* [*Attr*]...] { ALL | Name [*, Name*] ...}

Description

The **lsdom** command displays the attributes of the domain, which is defined from the domain database.

The command enables you to list attributes of all domains or specific domains. By default the **lsdom** command displays all domain attributes. To view selected attributes, use the **-a** list flag. If one or more attributes cannot be read, the **lsdom** command lists as much information as possible.

By default, the **lsdom** command lists each domain's attributes in one line. It displays attribute information as *Attribute*=*Value* definitions, each separated by a blank space. To list the domain attributes in stanza format, use the **-f** flag. To list the information as colon-separated records, use the **-C** flag.

Flags

Item	Description
-C	Displays the domain attributes in colon-separated records. *#domain:attribute1:attribute2: ...* *domain1:value1:value2: ...* *domain2:value1:value2: ...*
-f	Displays the output in stanzas, with each stanza identified by a domain name. Each *Attribute*=*Value* pair is listed in a separate line. **Domain:** *attribute1=value*

Parameters

Item	Description
ALL	Indicates that the attributes of all domains must be listed.
Name	Indicates the domain name whose attributes must be listed.

Security

The **lsdom** command is a privileged command. Callers of the command must have activated a role that has the following authorization to run the command successfully.

Item	Description
aix.security.domains.list	Required to run the command.

Files Accessed

Item	Description
File	Mode

Item	Description
/etc/security/domains	r

Examples

1. To display all attributes of the domain hrdom:

```
lsdom hrdom
```

All the attribute information is shown with each attribute separated by a blank space.

lsevent Command

Purpose

Lists event-monitoring information from the audit log.

Syntax

To list events from the audit log:

`lsevent` [`-O` *entries*] [`-B` *MMddhhmmyyyy*] [`-E` *MMddhhmmyyyy*] [`-e a | r | b`] [`-i`] [`-a | n` *node1*[`,`*node2*...]] [`-w` *event_node*] [`-h`] [`-TV`]

To list responses from the audit log:

`lsevent -r` [`-O` *entries*] [`-B` *MMddhhmmyyyy*] [`-E` *MMddhhmmyyyy*] [`-e {a | r | b | e | A}...`] [`-i`] [`-a | n` *node1*[`,`*node2*...]] [`-h`] [`-TV`] [response [response...]]

To list events for a condition from the audit log:

`lsevent` [`-O` *entries*] [`-B` *MMddhhmmyyyy*] [`-E` *MMddhhmmyyyy*] [`-e a | r | b`] [`-i`] [`-a | n` *node1*[`,`*node2*...]] [`-w` *event_node*] [`-h`] [`-TV`] *condition*

To list responses for a condition from the audit log:

`lsevent -R` [`-O` *entries*] [`-B` *MMddhhmmyyyy*] [`-E` *MMddhhmmyyyy*] [`-e {a | r | b | e | A}...`] [-i] [`-a | n` *node1*[`,`*node2*...]] [`-w` *event_node*] [-h] [-TV] *condition* [*response* [*response*...]]

To list events and responses for a condition from the audit log:

`lsevent -A` [`-O` *entries*] [`-B` *MMddhhmmyyyy*] [`-E` *MMddhhmmyyyy*] [`-e {a | r | b | e | A}...`] [-i] [`-a | n` *node1*[`,`*node2*...]] [`-w` *event_node*] [-h] [-TV] condition [*response* [*response*...]]

Description

The `lsevent` command lists event-monitoring information from the audit log. The audit log contains information about monitored events or conditions, and responses that were run as a result. This information allows a system administrator to see how events are being processed. The `lsevent` command lists only the information from the audit log recorded by RSCT event response resource manager (ERRM). By using `lsevent`, you can list audit log information without knowing detailed information about ERRM audit log templates, as you would need using the `lsaudrec` command.

By default, without using options and operands, the `lsevent` command lists the events that are recorded in the audit log. These events describe the monitored events that occurred. To list the events for a particular condition, specify the condition name.

Response information can be listed separately or with the event information. Responses are run based on a condition or event occurring. Information about a response includes when it was run, what the response script was, the return code, the expected return code, standard error output, and standard output. To see standard output and the expected return code, the response resource must be defined to record it

by `mkresponse` or `chresponse`. To list only response information, use the `-r` flag. You can optionally specify one or more response names to limit the number of responses listed.

To list event information and response information for a condition, you can use the `-R` and `-A` flags with a condition name. Without `-R` and `-A`, when a condition is specified, the events for the condition are listed. Specify `-R` to list the responses for the condition. You can specify one or more response names to limit the output to those responses. Specify `-A` to list the events and the responses. You can specify one or more response names to limit the response output for `-A` as well. If a condition and at least one response are specified without specifying the `-R`, `-A`, or `-r` flags, `-R` is assumed.

The type of event listed can be controlled using the `-e` flag. You can list events, rearm events, and error events for a condition. The `-w` flag can be used to list events that occurred on a particular node. The `-w` flag has meaning when it is used in listing events. Status information is displayed when the `-i` flag is specified. When listing conditions, the status information includes showing when the condition was registered and unregistered, and when event errors occur. For response information, the status information shows that a response is about to run.

Use the `-B` and `-E` flags if you need to specify a time to limit the command output. By default, `lsevent` lists all audit log entries according to the flags specified, but you can specify a beginning time or an ending time if you are interested in a certain period. The time format is described below. The `-0` flag is used to limit the search of the audit log to the most recent records. The value used with the `-0` flag determines how many of the most recent records are searched for the other `lsevent` criteria specified. For example, using `lsevent -0 1000` causes `lsevent` to search the most recent 1000 records in the audit log for events. If `-a` or `-n` is used, `-0` cannot be used.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM `nodegrp` command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

Parameters

condition
Specifies the name of a condition for which audit log information is listed.

response
Specifies the name of a response for which audit log information is listed.

Flags

-a
Specifies that the `lsevent` command retrieves audit log information from all of the nodes in the cluster. The CT_MANAGEMENT_SCOPE environment variable determines the scope of the cluster. If CT_MANAGEMENT_SCOPE is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and CT_MANAGEMENT_SCOPE is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-A
Specifies that event and response information for a condition is to be listed.

-B *MMddhhmmyyyy*
Specifies to list the audit log entries beginning at the time indicated. This time indicates when the audit log entry was created. Time stamps are in the form *MMddhhmmyyyy*, where *MM* is the two-digit month (01-12), *dd* is the two-digit day (01-31), *hh* is the two-digit hour (00-23), *mm* is the two-digit minute (00-59), and *yyyy* is the four-digit year. The time can be truncated from right to left, except for *MM*. If not all digits are specified, the year defaults to the current year, minutes to 0, hour to 0, and day to 01. At a minimum, the month must be specified.

-e a | **r** | **b** | **e** | **A**
Specifies the type of event to list from the audit log. The following parameters can be specified along with the **-e** flag:

a
Lists events from conditions. It is the default setting.

r
Lists rearm events from conditions.

b
List events and rearm events from conditions.

e
Lists response information that is triggered by error events. This setting is meaningful only when `-r`, `-R`, or `-A` is specified.

A
Lists all types of events (events, rearm events, and error events).

More than one event type can be specified, for example: `-e ae`.

If the **-e** flag is specified with the **-r** or **-R** flags, the response log entry for the batch-enabled condition is always displayed because the batched events file can contain all type of events.

-E *MMddhhmmyyyy*
Specifies to list the audit log entries up to or ending at the time indicated. This time indicates when the audit log entry was created. Time stamps are in the form *MMddhhmmyyyy*, where *MM* is the two-digit month (01-12), *dd* is the two-digit day (01-31), *hh* is the two-digit hour (00-23), *mm* is the two-digit minute (00-59), and *yyyy* is the four-digit year. The time can be truncated from right to left, except for *MM*. If not all digits are specified, the year defaults to the current year, minutes to 0, hour to 0, and day to 01. At a minimum, the month must be specified.

-i
Specifies that status information for a condition or response is to be listed. The status information includes information about event registration, event errors, and responses about to be run.

n *node1*[,*node2*...]
Specifies the node or nodes from which the audit log information is to be retrieved. If node is not specified, the local node is used. *node* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

-O *entries*
Specifies that only the latest entries in the audit log are searched for information. *entries* determines how many of the most recent records are search for the other **lsevent** criteria specified. For example, using `-O 1000` causes the `lsevent` command to search the most recent 1000 records in the audit log for events.

-r
Specifies that all command parameters are response names and that response information is to be returned for the responses specified. There are no condition names in the parameter list. If no response names are specified, then information is listed for all responses.

-R
Specifies that only the response information for a condition is to be listed.

-w *event_node*
Specifies the node on which the event occurred. This flag is only meaningful in listing events.

-h
Writes this command usage statement to standard output.

-T
Writes the command trace messages to standard error. For your software service organization use only.

-V
Writes the command verbose messages to standard output.

Environment variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used.

Standard output

When the `-h` flag is specified, this command usage statement is written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Restrictions

If you are using the `lsevent` command, you must have read access to the ERRM audit log resource on each node from which records are to be listed.

Authorization is controlled by the RMC access control list (ACL) file that exists on each node.

Implementation specifics

This command is part of the `rsct.core` fileset for the AIX operating system and `rsct.core-v.r.m.s-0.`*platform*`.rpm` package for the Linux, Solaris, and Windows platforms, where *platform* is `i386`, `ppc`, `ppc64`, `s390`, or `x86_64`.

Location

/opt/rsct/bin/lsevent

Examples

1. To list the information for events that occurred, enter:

   ```
   lsevent
   ```

2. To list the event information for a condition named `Condition1`, enter:

   ```
   lsevent Condition1
   ```

3. To list the event response information, enter:

   ```
   lsevent -r
   ```

4. To list the event response information for a response named `Response1`, enter:

   ```
   lsevent -r Response1
   ```

5. To view the output of the event response named `Response1`, which is defined to save its output, enter:

   ```
   lsevent -r Response1
   ```

6. To see the events found in the latest 1000 audit log records, enter:

   ```
   lsevent -O 1000
   ```

7. To list the rearm event information for a condition named `Condition1`, enter:

   ```
   lsevent -e r Condition1
   ```

lsfilt Command

Purpose

Lists filter rules from either the filter table or the IP Security subsystem.

Syntax

lsfilt -v 4|**6** [**-n** *fid_list*] [**-a**] [**-d**]

Description

Use the **lsfilt** command to list filter rules and their status.

Note: Filter description fields are not listed in the kernel. No filter description text will be displayed when active or dynamic filter rules are listed.

Flags

Item	Description
-a	List only the active filter rules. The active filter rules are the rules being used by the filter kernel currently. If omitted, all the filter rules in the filter rule table will be listed.
-d	Lists the dynamic filter rules used for Internet Key Exchange (IKE) tunnels. This table is built dynamically as IKE negotiations start creating IP Security tunnels and their corresponding filter rules are added to the dynamic IKE filter table.
-n	Specifies the ID(s) of filter rule(s) that are displayed. The *fid_list* is a list of filter IDs separated by a space or "**,**" or "**-**". The **-n** is not for active filter rules. This flag cannot be used with the **-a** flag.
-v	IP version of the filter rule you want to list. Valid values for this flag are **4** and **6**. If this flag is not used, both IP version 4 and IP version 6 are listed.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

lsfont Command

Purpose

Lists the fonts available to the display.

Syntax

lsfont [**-l**]

Description

The **lsfont** command displays a list of the fonts available to the display. The font identifier can help you change fonts using the **chfont** command.

You can use the System Management Interface Tool (SMIT) **smit lsfont** fast path to run this command.

Flags

Item	Description
-l	Specifies the removal of all header information from format of data.

Examples

To list all fonts available to the display, enter:

```
lsfont
```

The following example displays the font identifier, font name, glyph size, and font encoding for each available font:

```
FONT  FILE            GLYPH  FONT
ID    NAME            SIZE   ENCODING
====  ==============  =====  =========
0     Erg22.iso1.snf  12x30  ISO8859-1
1     Erg11.iso1.snf   8x15  ISO8859-1
```

Files

Item	Description
/bin/lsfont	Contains the **lsfont** command.
/usr/lpp/fonts	Contains fonts directory.

lsfs Command

Purpose

Displays the characteristics of file systems.

Syntax

lsfs [**-q**] [**-c** | **-l**] [**-a** | **-v** *VfsType* | **-u** *MountGroup*| [*FileSystem*...]]

Description

The **lsfs** command displays characteristics of file systems, such as mount points, automatic mounts, permissions, and file system size. The *FileSystem* parameter reports on a specific file system. The following subsets can be queried for a listing of characteristics:

- All file systems
- All file systems of a certain mount group
- All file systems of a certain virtual file system type
- One or more individual file systems

The **lsfs** command displays additional Journaled File System (JFS) or Enhanced Journaled File System (JFS2) characteristics if the **-q** flag is specified.

You can use the System Management Interface Tool (SMIT) **smit lsfs** fast path to run this command.

Flags

Item	Description
-a	Lists all file systems (default).
-c	Specifies that the output should be in colon format.
-l	Specifies that the output should be in list format.
-q	Displays additional Journaled File System (JFS) or Enhanced Journaled File System (JFS2) characteristics specific to the file system type. This information is not reported for other virtual file system types. It is displayed in addition to other file system characteristics reported by the **lsfs** command.
-u *MountGroup*	Reports on all file systems of a specified mount group.
-v *VfsType*	Reports on all file systems of a specified type.

Examples

1. To show all file systems in the **/etc/filesystems** file, enter:

```
lsfs
```

2. To show all file systems of vfs type `jfs`, enter:

```
lsfs -v jfs
```

3. To show the file system size, the fragment size, the compression algorithm (if any), and the number of bytes per i-node as recorded in the superblock of the root file system, enter:

```
lsfs -q /
```

Files

Item	Description
/etc/filesystems	Lists the known file systems and defines their characteristics.

lsgroup Command

Purpose

Displays group attributes.

Syntax

lsgroup [**-R** *load_module*] [**-c** | **-C** | **-f**] [**-a** *List*] {**ALL** | *Group* [**,***Group*] ...}

Description

The **lsgroup** command displays group attributes. You can use this command to list all the system groups and their attributes or you can list all the attributes of individual groups. Since there is no default parameter, you must enter the **ALL** keyword to list all the system groups and their attributes. All the attributes that are described in the **chgroup** command are displayed. If the **lsgroup** command cannot read one or more attributes, it lists as much information as possible, but does not display empty attributes. To view a selected attribute, use the **-a** *List* flag.

Note: If the *domainlessgroups* attribute is set in the /etc/secvars.cfg file, the **lsgroup** command lists the users from the LDAP module and the LOCAL module, if present.

By default, the **lsgroup** command lists each group on one line. It displays attribute information as *Attribute*=*Value* definitions, each separated by a blank space. To list the group attributes in stanza format, use the **-f** flag. To list the information in colon-separated records, use the **-c** or **-C** flag.

You can use the System Management Interface Tool (SMIT) **smit lsgroup** fast path to run this command.

Flags

Item	Description
-a *List*	Specifies the attributes to display. The *List* parameter can include any attribute that is defined in the **chgroup** command, and requires a blank space between attributes. If you specify an empty list, only the group names are listed.

Item	Description
-c	Displays the attributes for each group in colon-separated records, as follows: `#name:  attribute1:  attribute2:  ...` `Group:  value1:      value2:      ...` If a value contains a : symbol, then in the output : symbol is prefixed with the #! symbols.
-C	Displays the group attributes in colon-separated records that are easier to parse than the output of the **-c** flag: `#name:attribute1:attribute2: ...` `Group1:value1:value2: ...` `Group2:value1:value2: ...` The output is preceded by a comment line that has details about the attribute represented in each colon-separated field. If you also specify the **-a** flag, the order of the attributes matches the order specified in the **-a** flag. If you do not have a value for a given attribute, the field is still displayed, but is empty. If a value contains a : symbol, then in the output the : symbol is prefixed with #! symbols. The last field in each entry ends with a newline character rather than a colon.
-f	Displays the group attributes in stanzas. Each stanza is identified by a group name. Each *Attribute*=*Value* pair is listed on a separate line: `group:` `  attribute1=value` `  attribute2=value` `  attribute3=value`
-R *load_module*	Specifies the loadable I&A module that is used to get the group attribute list. If the *domainlessgroups* attribute is set in the/etc/secvars.cfg file and the **-R** *LDAP* command is used, then the attribute list is obtained from the LOCAL module, if the group exists on the LOCAL module, and does not exist on the LDAP module. This condition also applies to the **-R** *files* command.

Exit Status

This command returns the following exit values:

Item	Description
0	The command runs successfully and all requested changes are made.
>0	An error occurred. The printed error message lists further details about the type of failure.

Security

Access Control: This command must be a general user program with execute (x) access for all users. Attributes are read with the access rights of the invoker, so all users might not be able to access all the information. This attribute depends on the access policy of your system. This command must have the *trusted computing base* attribute.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files Accessed:

Mode	File
r	/etc/group
r	/etc/security/group
r	/etc/passwd

Limitations

Listing a group might not be supported by all loadable I&A modules. If the loadable I&A module does not support listing a group, then an error is returned.

Examples

1. To display the attributes of the `finance` group in the default format, enter the following command:

   ```
   lsgroup finance
   ```

2. To display the `id`, members (`users`), and administrators (`adms`) of the `finance` group in stanza format, enter the following command:

   ```
   lsgroup -f -a id users adms finance
   ```

3. To display the attributes of all the groups in colon-separated format, enter the following command:

   ```
   lsgroup -c ALL
   ```

   All the attribute information is displayed, with each attribute separated by a blank space.

4. To display the attributes of the LDAP I&A loadable module group monsters, enter the following command:

   ```
   lsgroup -R LDAP monsters
   ```

Files

Item	Description
/usr/sbin/lsgroup	Contains the **lsgroup** command.
/etc/group	Contains the basic attributes of groups.
/etc/security/group	Contains the extended attributes of groups.
/etc/passwd	Contains user IDs, user names, home directories, login shell, and finger information.

lsiscsi Command

Purpose

Displays information for iSCSI target data.

Syntax

`lsiscsi` [`-l` *AdapterName*] [`-g` *group*] [`-p`] [`-u`] [`-F` *Format*]

Description

The `lsiscsi` command displays iSCSI target data from ODM. There are two categories of data stored in ODM. The first is for statically configured iSCSI targets, which require that all the relevant iSCSI

target information (such as target name, IP address, and port number) are specified in order for AIX to discover them. The second category of iSCSI target data is for iSCSI target devices that can be configured automatically, but require authentication from the host (such as passwords). These two categories of iSCSI target data are associated with the `static` and `auto` groups, respectively, specified by the `-g` flag.

Flags

Item	Description
`-F` *Format*	Displays the output in a user-specified format, where the *Format* parameter is a quoted list of column names, separated and possibly ended by nonalphanumeric characters or white space. If white space is used as the separator, the `lsiscsi` command displays the output in aligned columns.
`-g` *group*	Specifies which group this iSCSI target is associated with. There two valid groups are `static` and `auto`. The `static` group is for iSCSI targets that cannot be automatically discovered from this host; all relevant iSCSI target information for them (such as target name, IP address, and port number) must be specified. The `auto` group is for iSCSI targets that are automatically discovered, but require authentication information such as passwords.
`-l` *AdapterName*	Specifies the adapter name for the iSCSI TCP/IP Offload Engine (TOE) adapter that is attached to this iSCSI target. It can also specify the iSCSI protocol device for the iSCSI software solution device.
`-p`	Displays the iSCSI target's password used for iSCSI logins from this adapter.
`-u`	Displays the Challenge Handshake Authentication Protocol (CHAP) user name that can be used for each iSCSI target.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

The `lsiscsi` command is executable only by root.

Examples

1. To list all iSCSI target data for the iSCSI TOE adapter `ics0`, enter:

   ```
   lsiscsi -l ics0
   ```

   The system displays output similar to the following:

```
  10.1.2.116 3260 iqn.sn9216.iscsi-hw1
  10.1.2.116 3260 iqn.sn2105.iscsi-target
```

2. To list all iSCSI target data for this host, enter:

```
lsiscsi
```

The system displays output similar to the following:

```
  ics0 1 10.1.2.116 3260 iqn.sn9216.iscsi-hw1
  ics0 10.1.2.116 3260 iqn.sn2105.iscsi-target
  ics1 11.23.45.67 iqn.mds9216.iscsi_hw2.116 3260 iqn.sn2105.iscsi-target
```

Location

`/usr/sbin/lsiscsi`

Files

Item	Description
`src/bos/usr/sbin/iscsia`	Contains the common source files from which the iSCSI commands are built.

lsitab Command

Purpose

Lists records in the **/etc/inittab** file.

Syntax

lsitab { **-a** | *Identifier* }

Description

The **lsitab** command displays a record in the **/etc/inittab** file. You can display all of the records in the **/etc/inittab** file, or use the *Identifier* parameter to display a specific record. The *Identifier* parameter is a 14-character field that uniquely identifies an object.

Flags

Item	Description
-a	Specifies that all records in the **/etc/inittab** file are listed.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the record for `tty2`, enter:

```
lsitab "tty002"
```

The output is similar to: `tty002:2:respawn:/usr/sbin/getty /dev/tty2`

2. To display all of the records in the **/etc/inittab** file, enter:

```
lsitab -a
```

All of the records in the **/etc/inittab** file are displayed.

Files

Item	Description
/etc/inittab	Indicates which processes the **init** command starts.

lskbd Command

Purpose

List the current software keyboard map loaded into the system.

Syntax

lskbd

Description

The **lskbd** command displays the absolute pathname of the current software keyboard map loaded into the system.

To list the current software keyboard map enter:

```
lskbd
```

You can use the System Management Interface Tool (SMIT) **smit lskbd** fast path to run this command.

Note: This command can be used only on an LFT display.

Example

Following is an example of the listing displayed by the lskbd command:

```
The current software keyboard map = /usr/lib/nls/loc/C.lftkeymap
```

Files

Item	Description
/usr/bin/lskbd	Contains the **lskbd** command.
/usr/lib/nls/loc	Software keyboard map directory.

lskst Command

Purpose

Lists the entries in the kernel security tables.

Syntax

lskst -t *table* [**-C** | **-f**] [*Name* [, *Name*]...]

lskst -l

Description

The **lskst** command reads the kernel security tables (KST) and displays the information on standard output (**stdout**). The output of the **lskst** command might differ from what is displayed by the **lsauth**, **lsrole** and **lssecattr** commands if the associated file databases are modified after the databases are sent to the KST through the **setkst** command.

Specify the table to be displayed with the **-t** flag. By default, all the information in the specified table is displayed. Alternatively, a specific entry in the table can be selected by specifying the *Name* parameter.

By default, the **lskst** command lists the attributes of each entry on one line. It displays attribute information as *Attribute* **=** *Value* definitions, each separated by a blank space. To list the table attributes in stanza format, use the **-f** flag. To list the information as colon-separated records, use the **-C** flag.

Flags

Item	Description
-C	Displays the table attributes in colon-separated records as follows: `#name:attribute1:attribute2:...` `entry_name:value1:value2:...`
-f	Displays the output in stanzas, with each stanza identified by the entry name. Each *Attribute* **=** *Value* pair is listed on a separate line: *entry_name*: *attribute1=value* *attribute2=value* *attribute3=value*
-l	Displays the current value of the `loglevel` variable that is set in the kernel by using the **setkst** command.
-t *table*	Retrieves data from the specified security table from the KST. The parameter for the **-t** flag can be one of the following values: **auth** Authorizations table **role** Role table **cmd** Privileged command table **dev** Privileged device table **dom** Domains **domobj** Domain objects

Parameters

Item	Description
Name	Represents a specific entry of a kernel table. It can be an authorization, a role, a privileged command or a privileged device, depending on the table specified by the **-t** *table* flag.

Security

The **lskst** command is a privileged command. You must assume a role that has the following authorization to run the command successfully.

Item	Description
aix.security.kst.list	Required to run the command.

Examples

1. To retrieve all the entries in the role table from the KST, use the following command:

```
lskst -t role
```

2. To display the entry for the **/usr/bin/mycmd** command from the privileged command table in stanza format, use the following command:

```
lskst -t cmd -f /usr/bin/mycmd
```

3. To display the `aix.security` authorization table in the kernel, use the following command:

```
lskst -t auth aix.security
```

4. To retrieve all the entries in the domain object table from the KST, use the following command:

```
lskst -t domobj
```

lsldap Command

Note: If an LDAP user is created with a UID > 2 ^ 31, the `lsldap` command displays it as a negative number.

Purpose

Displays naming service objects from the configured LDAP directory server.

Syntax

`lsldap` [`-a`] [*entity* [*entry_name* | *filter*]]

Description

The `lsldap` command displays the naming service objects from the configured LDAP directory server. It queries the LDAP server through the `secldapclntd` daemon. Some or all of the objects of a particular entity can be listed by the `lsldap` command. By default, the `lsldap` command displays only the distinguished name (DN) of the returned objects. In addition, the `-a` flag can be used to view the attributes.

The `lsldap` command supports the following entities:

Entity	objectClass	Default attribute name
aapolicies	ibm-aixAccountingAdminPolicy	ibm-aixAdminPolicyName
aaprojects	ibm-aixAccountingProject	ibm-aixProjectName
admkeystore	ibm-usrkeystore	cn
aixpert	ibm-aixAixpert	ibm-aixpertLabel
aliases	mailGroup	cn
auditclass	AIXAuditClassStanza	auditclassstanza
auditconfig	AIXAuditConfig	auditconfig
authorizations	ibm-authorization	cn
automount	automountMap nisObject	automountMapNamenisMapName
bootparams	bootableDevice	cn
domains	ibm-aixRBACdomain	ibm-aixRBACdomainName
domobjs	ibm-aixRBACdomainObject	ibm-aixRBACdomainObjectName
efscookies	ibm-efskscookies	cn
ethers	ieee802Device	cn
group	posixgroupAIXAccessGroup	cngroupname
grpkeystore	ibm-grpkeystore	cn
hosts	ipHost	cn
netgroup	ipNetgroup	cn
networks	ipNetwork	cn
passwd	posixAccountAIXAccount	uidusername
privcmds	ibm-privcmd	cn
privdevs	ibm-privdev	cn
protocols	ipProtocol	cn
roles	aixaccessroles	rolename
rpc	oncRpc	cn
services	ipService	cn
privfiles	ibm-privfile	cn
usrkeystore	ibm-usrkeystore	cn

The automount entity has two object classes. The `lsldap` command treats `automountMap` with higher precedence over `nisMap` by always returning `automountMap` objects if it finds any, and returning nisMap objects only in the absence of `automountMap` objects.

For the **passwd** and group entities, the `lsldap` command returns the correct objects according to the LDAP client configuration. However, the correct attribute name corresponding to the object classes must be supplied for `lsldap passwd attribute=value` queries.

If an entity name is not specified from the command line, the `lsldap` command displays container entries of the entities and any other entries that are siblings of these containers. Users must have root permissions to list the container entries.

The *entry_name* parameter is the name of the object to be queried. For example, if the entity is **passwd**, the *entry_name* is the user account name. The *entry_name* parameter is equivalent to `default attribute name = entry_name`. The `lsldap` command accepts the `*` wildcard in *entry_name* for a substring search. All entries are returned if *entry_name* is not specified.

Instead of *entry_name*, a *filter* can also be supplied to search for entries that match certain criteria. Simple filters can be specified as *attributename=attributevalue*, where *attributename* is the LDAP attribute name.

The `lsldap` command prints the result to `stdout`. If the `-a` flag is not specified, `lsldap` prints entries that are found in the form of DNs, with each DN separated by a blank line. If the `-a` flag is specified, each entry is printed in the ldif format, with a blank line between entries.

Flags

Item	Description
`-a`	Displays all attributes of returned objects. By default only the DN of the objects are displayed.

Exit Status

Upon success, the `lsldap` command returns `0`. Upon failure, a nonzero value is returned, with one of the following error messages that are written to `stderr`:

Item	Description
`EIO`	Connection error.
`EINVAL`	Invalid parameters.
`EPERM`	No permission to run the operation.
`ENOMEM`	Not enough memory.
other `errno`	Other errors.

Security

The `lsldap` command can be run by any user. It is owned by the root user and security group, and has access permissions of 555.

When a non-privileged user runs the `lsldap –a passwd` command for a `netgroup` enabled LDAP module, the `lsldap` command does not display the user information if the `DisplayNetgroupUserInfo` attribute is set to no in the `ldap.cfg` file. By default, the user information is displayed in a `netgroup` enabled LDAP module by running the `lsldap` command irrespective of your user privileges.

When you list the **passwd** entity with the `-a` flag by root user, `lsldap` returns all attributes of the found users. However, when the same command is run by a nonprivileged user, `lsldap` returns only the same commonly readable attributes as returned by the `lsuser` command in addition to the object class information. For all other entities, regardless of which user runs the command the same output is generated.

Examples

1. To list all entries of the host entity, enter the following command:

   ```
   lsldap hosts
   ```

   Information similar to the following is returned:

   ```
   dn: cn=myhost+ipHostNumber=192.3.193.46,ou=Hosts,cn=aixdata
   ```

```
dn: cn=starfish+ipHostNumber=192.3.193.47,ou=Hosts,cn=aixdata


dn: cn=loopback+ipHostNumber=127.0.0.1,ou=Hosts,cn=aixdata
```

2. To list host `starfish` and all of its attributes, enter the following command:

```
lsldap -a hosts starfish
```

Information similar to the following is returned:

```
dn: cn=starfish+ipHostNumber=192.3.193.47,ou=Hosts,cn=aixdata
objectClass: top
objectClass: ipHost
objectClass: device
ipHostNumber: 192.3.193.47
cn: loopback
cn: localhost
```

3. To list users with names that begin with the letter b, enter the following command:

```
lsldap passwd "b*"
```

Information similar to the following is returned:

```
dn: uid=bin,ou=people,cn=aixdata


dn: uid=bob,ou=people,cn=aixdata
```

4. To list user `foo` and its attributes, enter the following command:

```
lsldap -a passwd foo
```

Information similar to the following is returned:

```
dn: uid=foo,ou=people,cn=aixdata
uid: foo
objectClass: account
objectClass: posixAccount
objectClass: shadowAccount
objectClass: top
cn: foo
uidNumber: 259
gidNumber: 1
homeDirectory: /home/foo
loginShell: /usr/bin/ksh
shadowlastchange: 12740
userpassword: {crypt}rNnLQ9TAD2u/k
shadowmin: 5
```

5. To list users who run `/usr/bin/ksh`, enter the following command:

```
lsldap passwd loginshell=/usr/bin/ksh
```

Information similar to the following is returned:

```
dn: uid=bin,ou=people,cn=aixdata


dn: uid=bob,ou=people,cn=aixdata


dn: uid=foo,ou=people,cn=aixdata
```

Restrictions

The `lsldap` command relies on the `secldapclntd` daemon to work.

Location

/usr/sbin/lsldap

lslicense Command

Purpose

Displays the number of fixed licenses and the status of the floating licensing.

Syntax

lslicense [**-A**] [**-c**]

Description

The **lslicense** command displays the number of fixed licenses and the status of the floating licensing.

Flags

Item	Description
-A	The **-A** flag causes the **lslicense** command to report the current number of available fixed licenses. When the **-A** flag is not specified, the maximum number of fixed licenses and license status is reported.
-c	Displays the output in : (colon) form.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the number of fixed licenses and the floating license status, enter:

```
lslicense
```

Example output would be:

```
Maximum number of fixed licenses is 10.
Floating licensing is enabled.
```

2. To display the number of fixed licenses and the floating license status in a colon format, enter:

   `lslicense -c`

   Example output would be:

   ```
   #fixed:floating
   10:on
   ```

3. To display license information including the number of available fixed licenses, enter:

   ```
   lslicense -A
   ```

   Output similar to the following will display:

   ```
   Maximum number of fixed licenses is 2.
   Floating licensing is disabled.
   Number of available fixed licenses is 2.
   ```

lslpclacl Command

Purpose

Displays the access controls for the least-privilege (LP) resource class (`IBM.LPCommands`).

Syntax

To display the access controls for the `IBM.LPCommands` resource class:

- On the local node:

  `lslpclacl [ -l | -i | -t | -d | -D` *delimiter* `] [-p] [-E] [-x] [-h] [-TV]`

- On all nodes in a domain:

  `lslpclacl -a [ -l | -i | -t | -d | -D` *delimiter* `] [-p] [-E] [-x] [-h] [-TV]`

- On a subset of nodes in a domain:

  `lslpclacl { -n` *host1*`[,`*host2*`,... ] } [ -l | -i | -t | -d | -D` *delimiter* `] [-p] [-E] [-x] [-h] [-TV]`

Description

The `lslpclacl` command displays the access control list (ACL) that is associated with the least-privilege (LP) resource class (`IBM.LPCommands`). The accesses contained in the ACL entries are displayed. The `IBM.LPCommands` Class ACL controls access to the `IBM.LPCommands` class operations. By default, this command displays information in table format (`-t`).

This command displays the following ACL information:

Field	Description
Identity	The network identity of the user. See the lpacl command for a description of the network identity.

Field	Description
Permissions	The permissions allowed for `Identity`. The valid values are: **a** Administrator permission **r** Read permission (consists of the e, l, q, and v permissions) **w** Write permission (consists of the c, d, o, and s permissions) **x** Execute permission **c** Refresh permission **d** Define and undefine permission **e** Event permission **l** Enumerate permission **o** Online, offline, and reset permission **q** Query permission **s** Set permission **v** Validate permission **0** No permission
NodeName	The location of the `IBM.LPCommands` resource class (for management domain scope or peer domain scope).
PeerDomain	The name of the RSCT peer domain in which the `IBM.LPCommands` resource class is defined. This field is displayed when the `-p` flag is specified.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the `-a` flag. If you want this command to run on a subset of nodes in a domain, use the `-n` flag. Otherwise, this command runs on the local node.

Flags

-a

Displays the `IBM.LPCommands` Class ACLs on all nodes in the domain. The `CT_MANAGEMENT_SCOPE` environment variable setting determines the cluster scope. If `CT_MANAGEMENT_SCOPE` is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The `lslpclacl` command runs once for the first valid scope that the LP resource manager finds. For example, suppose that a management domain and a peer domain exist and the `CT_MANAGEMENT_SCOPE` environment variable is not set. In this case, `lslpclacl –a` runs

in the management domain. To run `lslpclacl –a` in the peer domain, you must set `CT_MANAGEMENT_SCOPE` to 2.

-i
Generates a template in a form that can be used, after appropriate editing, as file input to the `chlpclacl` command.

-l
Displays the information about separate lines (long format).

-t
Displays the information in separate columns (table format). It is the default.

-d
Displays the information using delimiters. The default delimiter is a pipe symbol (`|`). Use the `-D` flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default pipe symbol (`|`) when the information that you want to display contains pipe symbols, for example. You can use this flag to specify a delimiter of one or more characters.

-n *host1*[,*host2*,...]
Specifies the node in the domain from which the `IBM.LPCommands` Class ACL is displayed. By default, the `IBM.LPCommands` Class ACL is displayed on the local node. This flag is valid only in a management domain or a peer domain. If `CT_MANAGEMENT_SCOPE` is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope found.

-p
Displays the name of the RSCT peer domain in which the `IBM.LPCommands` resource class is defined.

-E
Displays read permission as `elqv` instead of `r` and write permission as `cdos` instead of `w`.

-x
Excludes the header (suppresses header printing).

-h
Writes the command usage statement to standard output.

-T
Writes the command trace messages to standard error.

-V
Writes the command verbose messages to standard output.

Environment variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When `CT_CONTACT` is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If `CT_CONTACT` is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used, unless the `-a` flag or the `-n` flag is specified.

Standard output

When the `-h` flag is specified, this command usage statement is written to standard output. When the `-V` flag is specified, this command verbose messages are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
The resource was not found.

Security

To run the `lslpclacl` command, you need read permission in the Class ACL of the `IBM.LPCommands` resource class. Permissions are specified in the LP ACLs on the contacted system. See "lpacl Information" on page 1951 for general information about LP ACLs and the *Administering RSCT* guide for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for the AIX and Linux operating systems.

Location

/opt/rsct/bin/lslpclacl

Examples

1. To list the IBM.LPCommands Class ACLs on nodeA in table format, run this command on nodeA:

```
lslpclacl
```

   The following output is displayed:

```
Identity                        Permissions     NodeName
joe@LOCALHOST                   ra              nodeA
bill@0x374bdcbe384ed38a         rwa             nodeA
jane@0x374bdcbe384ed38a         rwa             nodeA
```

2. To list the IBM.LPCommands Class ACLs on nodeA in long format, run this command on nodeA:

```
lslpclacl -l
```

   The following output is displayed:

```
Class ACLs for LPRM
NodeName nodeA
    Identity    =     joe@LOCALHOST
    Permissions =     ra

    Identity    =     bill@0x374bdcbe384ed38a
    Permissions =     rwa

    Identity    =     jane@0x374bdcbe384ed38a
    Permissions =     rwa
```

3. To list the IBM.LPCommands Class ACLs on nodeA in delimited format, run this command on nodeA:

```
lslpclacl -d
```

   The following output is displayed:

```
Identity|Permissions|NodeName
joe@LOCALHOST|ra|nodeA
bill@0x374bdcbe384ed38a|rwa|nodeA
jane@0x374bdcbe384ed38a|rwa|nodeA
```

4. To list the IBM.LPCommands Class ACLs on nodeA in the active domain, run this command:

```
lslpclacl -a
```

   The following output is displayed:

```
Identity                        Permissions     NodeName
joe@LOCALHOST                   ra              node1.pok.ibm.com
bill@0x374bdcbe384ed38a         rwa             node1.pok.ibm.com
jane@0x374bdcbe384ed38a         rwa             node1.pok.ibm.com
joe@LOCALHOST                   ra              node2.pok.ibm.com
jane@0x374bdcbe384ed38a         rwa             node2.pok.ibm.com
```

5. To list the IBM.LPCommands Class ACLs on nodeA in the active domain and list the peer domain name, run this command:

```
lslpclacl -ap
```

   The following output is displayed:

```
Identity                        Permissions     NodeName                PeerDomain
joe@LOCALHOST                   ra              node1.pok.ibm.com       PD1
bill@0x374bdcbe384ed38a         rwa             node1.pok.ibm.com       PD1
jane@0x374bdcbe384ed38a         rwa             node1.pok.ibm.com       PD1
joe@LOCALHOST                   ra              node2.pok.ibm.com       PD1
jane@0x374bdcbe384ed38a         rwa             node2.pok.ibm.com       PD1
```

lslpcmd Command

Purpose

Lists information about the least-privilege (LP) resources on one or more nodes in a domain.

Syntax

To display LP resource information:

- On the local node:

  `lslpcmd` [–A | *resource_name1* [, *resource_name2* , ...] | –R *RunCmdName1* [, *RunCmdName2* , ...]] [`-h`] [`-TV`]

- On all nodes in a domain:

  `lslpcmd -a` [–A | *resource_name1* [, *resource_name2* , ...] | –R *RunCmdName1* [, *RunCmdName2* , ...]] [`-h`] [`-TV`]

- On a subset of nodes in a domain:

  `lslpcmd -n` *host1* [, *host2* , ...] [–A | *resource_name1* [, *resource_name2* , ...] | –R *RunCmdName1* [, *RunCmdName2* , ...]] [`-h`] [`-TV`]

Description

The `lslpcmd` command displays information about LP resources on one or more nodes in a domain. LP resources are `root` commands or scripts to which users are granted access based on permissions in the LP access control lists (ACLs). Use this command to display the attributes of one or more LP commands by specifying the *resource_name1* , [*resource_name2* , ...] parameter. If you omit this parameter, the `lslpcmd` command lists the names of all of the LP commands. Use the –A flag to list all of the LP commands and all of their attributes and values. Use the –R flag to list one or more LP resources that have a particular `RunCmdName` value.

The `lslpcmd` command lists the following information about defined LP resources:

Field	Description
Name	The name of the LP resource.
CommandPath	The fully-qualified path of the LP resource.
Description	A description of the LP resource.
Lock	The lock setting. Valid values are: `0` (the lock is not set) and `1` (the lock is set).
CheckSum	The `CheckSum` value of the LP resource to which `CommandPath` points. The LP resource manager assigns a value of `0` if the LP resource does not exist or if the user did not update the `CheckSum` value after the LP resource was made available.
RunCmdName	The LP resource name that is used as a parameter with the `runlpcmd` command.
FilterScript	The path to the filter script.
FilterArg	The list of arguments to pass to `FilterScript`.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the `-a` flag. If you want this command to run on a subset of nodes in a domain, use the `-n` flag. Otherwise, this command runs on the local node.

Flags

-a
Displays information about one or more LP resources on all nodes in the domain. The CT_MANAGEMENT_SCOPE environment variable's setting determines the cluster scope. If CT_MANAGEMENT_SCOPE is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The `lslpcmd` command runs once for the first valid scope that the LP resource manager finds. For example, suppose a management domain and a peer domain exist and the CT_MANAGEMENT_SCOPE environment variable is not set. In this case, `lslpcmd –a` runs in the management domain. To run `lslpcmd –a` in the peer domain, you must set CT_MANAGEMENT_SCOPE to 2.

-n *host1*[,*host2*,...]
Specifies the node or nodes in the domain on which the LP resource is to be listed. By default, the LP resource is changed on the local node. The –n flag is valid only in a management or peer domain. If the CT_MANAGEMENT_SCOPE variable is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The `lslpcmd` command runs once for the first valid scope that the LP resource manager finds.

–A
Displays all of the LP resources with their attributes and values.

–R
Display all attributes of the LP resources that have the same RunCmdName value.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error.

-V
Writes the command's verbose messages to standard output.

Parameters

***resource_name1*[,*resource_name2*,...]**
Specifies one or more LP resources for which you want to display information.

Security

To run the `lslpcmd` command, you need:

- read permission in the Class ACL of the `IBM.LPCommands` resource class.
- read permission in the Resource ACL.

  As an alternative, the Resource ACL can direct the use of the Resource Shared ACL if this permission exists in the Resource Shared ACL.

Permissions are specified in the LP ACLs on the contacted system. See the `lpacl` file for general information about LP ACLs and the *RSCT Administration Guide* for information about modifying them.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
The resource was not found.

Environment Variables

CT_CONTACT
Determines the system that is used for the session with the RMC daemon. When `CT_CONTACT` is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If `CT_CONTACT` is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the LP resources that are processed.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to process the LP resources. The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. When the `-V` flag is specified, this command's verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To list the names of all LP resources on the local node, enter:

```
lslpcmd
```

   The output will look like this:

```
lpcommand1
lpcommand2
```

2. To list the names and attributes of all LP resources on the local node, enter:

```
lslpcmd -A
```

   The output will look like this:

```
Name=lpcommand1
CommandPath=/tmp/my_command
Description=
Lock=1
CheckSum=112
RunCmdName=lpcommand1
FilterScript=
FilterArg=
---------------------------------
Name=lpcommand2
CommandPath=/tmp/cmds/this_command
Description=
Lock=0
CheckSum=0
RunCmdName=lpcommand2
FilterScript=
FilterArg=
---------------------------------
```

3. To list the attributes of the LP resource `lpcommand1` on the local node, enter:

```
lslpcmd lpcommand1
```

   The output will look like this:

```
Name=lpcommand1
CommandPath=/tmp/my_command
Description=
Lock=1
CheckSum=100
RunCmdName=lpcommand1
FilterScript=
FilterArg=
```

4. To list the attributes of LP resources that have a `RunCmdName` value of `rpower` on the local node, enter:

```
lslpcmd -R rpower
```

   The output will look like this:

```
Name=lpcommand1
CommandPath=/opt/csm/bin/rpower
Description=
Lock=1
CheckSum=112
RunCmdName=rpower
FilterScript=/tmp/test1
FilterArg=node1,node2,node3
------------------------------
Name=lpcommand2
CommandPath=/opt/csm/bin/rpower
Description=
Lock=0
CheckSum=112
RunCmdName=rpower
FilterScript=/tmp/test1
```

```
FilterArg=node4,node5,node6
------------------------------
⋮
```

Location

/opt/rsct/bin/lslpcmd
Contains the `lslpcmd` command

lslpp Command

Purpose

Lists installed software products.

Syntax

lslpp [**-R** { *path* | **ALL** }] { **-d** | **-E** | **-f** | **-h** | **-i** | **-l** | **-L** | **-p** } [**-a**] [**-c**] [**-J**] [**-q**] [**-I**] [**-O** { [**r**] [**s**] [**u**] }] [*FilesetName ...* | -b *File* | **all**]

lslpp [**-R** { *path* | **ALL** }] **-w** [**-c**] [**-q**] [**-O** { [**r**] [**s**] [**u**] }] [*FileName ...* | **all**]

lslpp [**-R** { *path* | **ALL** }] **-L -c** [**-v**]

lslpp [**-R** { *path* | **ALL** }] **-S** [**A**|**O**]

lslpp [**-R** { *path* | **ALL** }] **-e**

Description

The **lslpp** command displays information about installed filesets or fileset updates. The *FilesetName* parameter is the name of a software product. The *File* parameter specifies a bundle file to use as a fileset list.

When only the **-l** (lowercase L) flag is entered, the **lslpp** command displays the latest installed level of the fileset specified for formatted filesets. The base level fileset is displayed for formatted filesets. When the **-a** flag is entered along with the **-l** flag, the **lslpp** command displays information about all installed filesets for the *FilesetName* specified. The **-I** (uppercase i) flag combined with the **-l** (lowercase L) flag specifies that the output from the **lslpp** command should be limited to base level filesets.

The `lslpp` command and the **compare_report** command both show information about interim fixes installed on the system. The `lslpp -L` or `lslpp -Lc` command and the `lslpp -e` command must be run by root. Any interim fix information returned is used by the `compare_report` command. The information includes an interim fix label and a level value. The interim fix label is the equivalent of a fileset name, and its level is based on the time (*YY.MM.DD.HHMMSS*, where *YY* is the year, *MM* is the month, *DD* is the day, *HH* is the hour, *MM* is the minute, and *SS* is the second) in which the interim fix was packaged. If a non-root user runs these commands, only software products and levels are returned, and interim fix information is not included. If a root user runs the `lslpp -e` command and the `lslpp -L` command, interim fix information can be shown.

The **-d**, **-f**, **-h**, **-i**, **-l** (lowercase L), **-L**, and **-p** flags request different types of output reports.

The **-a**, **-c**, **-J**, and **-q** flags specify the amount and format of the information that is displayed in the report.

The **-O** flag specifies that data is to come from a specified part of the fileset. The part may be the root part, **-Or**, the share part, **-Os**, or the **usr** part, **-Ou**.

The default value for the *FilesetName* parameter is **all**, which displays information about all installed software products. Pattern matching characters, such as * (asterisk) and ? (question mark), are valid in the *FilesetName* parameter. You don't have to enclose these characters in '' (single quotation marks).

However, using single quotation marks prevents you from searching the contents of your present directory.

Output Values

Much of the output from the **lslpp** command is understandable without an explanation. Other fields contain data that needs to be defined. The following sections define terms used in several of the output fields.

State Values

The **state** field in the **lslpp** output gives the state of the fileset on your system. It can have the following values:

State	Definition
APPLIED	The specified fileset is installed on the system. The **APPLIED** state means that the fileset can be rejected with the **installp** command and the previous level of the fileset restored. This state is only valid for Version 4 fileset updates and 3.2 migrated filesets.
APPLYING	An attempt was made to apply the specified fileset, but it did not complete successfully, and cleanup was not performed.
BROKEN	The specified fileset or fileset update is broken and should be reinstalled before being used.
COMMITTED	The specified fileset is installed on the system. The **COMMITTED** state means that a commitment has been made to this level of the software. A committed fileset update cannot be rejected, but a committed fileset base level and its updates (regardless of state) can be removed or deinstalled by the **installp** command.
EFIXLOCKED	The specified fileset is installed on the system and is locked by the interim fix manager (the **emgr** command).
OBSOLETE	The specified fileset was installed with an earlier version of the operating system but has been replaced by a repackaged (renamed) newer version. Some of the files that belonged to this fileset have been replaced by versions from the repackaged fileset.
COMMITTING	An attempt was made to commit the specified fileset, but it did not complete successfully, and cleanup was not performed.
REJECTING	An attempt was made to reject the specified fileset, but it did not complete successfully, and cleanup was not performed.

Action Values

The **action** field in the **lslpp** output identifies the installation action that was taken for the fileset. The following values may be found in this field:

Action	Definition
APPLY	An attempt was made to apply the specified fileset.
CLEANUP	An attempt was made to perform cleanup for the specified fileset.
COMMIT	An attempt was made to commit the specified fileset.
REJECT	An attempt was made to reject the specified fileset.

Status Values

The **status** field in the **lslpp** output identifies the resultant status in the history of installation actions. The following values may be found in this field:

Status	Definition
BROKEN	The fileset was left in a broken state after the specified action.
CANCELED	The specified action was canceled before it completed.
COMPLETE	The commitment of the fileset has completed successfully.

Flags

Item	Description
-a	Displays **all** the information about filesets specified when combined with other flags. This flag shows all updates when combined with the **-l** flag and all history when combined with the **-h** flag. This flag cannot be specified with the **-f** flag.
-b *File*	Specifies a bundle file to search for fileset names. The filesets listed in the bundle are then listed as if they had been specified explicitly as *FilesetName* parameters. To mimic installp behavior, the installp image names are automatically wildcarded. For example, a bundle file entry of `I:bos.abc` will behave as if `bos.abc*` was specified as a *FilesetName* parameter. **Note:** This might also return results for `bos.abcdef`. If the file does not reside in one of the known bundle locations, the full path and file name, including extension, must be specified.
-c	Displays information as a list separated by colons. This flag cannot be specified with the **-J** flag.
-d	Displays filesets that are dependents of the specified software. A dependent fileset is one that has the specified software as a prerequisite, corequisite, ifrequisite, or installed requisite.
-e	Displays every interim fix installed on the system.
-E	Lists license agreements.
-f	Displays the names of the files added to the system during installation of the specified fileset. This flag cannot be specified with the **-a** flag.
-h	Displays the installation and update history information for the specified fileset. You cannot use this flag with the **-J** flag.
-I	(uppercase i) Limits the inputs to software products.
-i	Displays the product information for the specified fileset.
-J	Generates output in a form suitable for the System Management Interface Tool (SMIT) command to list output. This flag can only be specified with the **-l** (lowercase L) and **-L** flags.
-l	(lowercase L) Displays the name, most recent level, state, and description of the specified fileset.

Item	Description
-L	Displays the name, most recent level, state, type, and a description of the specified fileset. Part information (**usr**, **root**, and **share**) is consolidated into the same listing. For formatted filesets, it displays the most recent maintenance or technology level for the specified filesets. In addition, this flag lists any subsystem selective fixes that were installed on top of the maintenance or technology level. RPM and ISMP images are also listed. When combined with the **-c** flag, there is a difference in the **Type** field when used with an installp image. A blank value indicates an installp image without any updates. A value of **F** indicates an installp image with updates. When combined with the **-c** flag, the build date, which is specified by the year and the week in the form of **yyww** (for example, 0852), is displayed for the fileset, if there is one. Additional fields are displayed with the **-Lc** output, as indicated in the header of the output.
-O	Lists information for the specified part of the fileset. When the **-O** flag is not specified information is listed for all parts. This option is designed for use by the **nim** command to list software product information for diskless or dataless workstations. You can use the following flags with this flag: **-r** Indicates to list information for the root part. **-s** Indicates to list information for the **/usr/share** part. **-u** Indicates to list information for the **/usr** part.
-p	Displays requisite information for the specified fileset.
-q	Suppresses the display of column headings.
-R { *path* | **ALL** }	Indicates a user-specified installation location.
-S [**A**|**O**]	Displays a list of automatically installed filesets and a list of optionally installed filesets. If the **-S** flag is followed by **A**, then only the automatically installed filesets are listed. If the **-S** flag is followed by **O**, then only the optionally installed filesets are listed.
-v	Displays only information from the vendor database, which contains ISMP product information. This flag is only valid when used with both the **-L** and the **-c** flags.
-w	Lists fileset that owns this file.

You must specify one of the mutually exclusive flags: **-d**, **-e**, **-E**, **-f**, **-h**, **-i**, **-l**, **-L**, **-p**, **-S**, and **-w**.

Examples

1. To list the installation state for the most recent level of installed filesets for all of the **bos.rte** filesets, type:

```
lslpp -l "bos.rte.*"
```

2. To list the installation state for the base level and updates for the fileset **bos.rte.filesystem**, type:

```
lslpp -La bos.rte.filesystem
```

3. To list the installation history information of all the filesets in the **bos.net** software package, type:

```
lslpp -ha 'bos.net.*'
```

4. To list the names of all the files of the **bos.rte.lvm** fileset, type:

```
lslpp -f bos.rte.lvm
```

5. To list the fileset that owns **installp**, type:

```
lslpp -w /usr/sbin/installp
```

Output similar to the following displays:

```
File                            Fileset
  Type
-----------------------------------------------------------------
/usr/sbin/installp              bos.rte.install          File
```

6. To list the fileset that owns all file names that contain `installp`, type:

```
lslpp -w "*installp*"
```

Output similar to the following displays:

```
File                            Fileset
  Type
-----------------------------------------------------------------
/usr/sbin/installp              bos.rte.install          File
/usr/clvm/sbin/linstallpv       prpq.clvm                File
/usr/lpp/bos.sysmgt/nim/methods/c_installp
                                bos.sysmgt.nim.client    File
```

7. To display all files in the inventory database, type:

```
lslpp -w
```

8. To display the installation state for the RPM `cdrecord` image , type:

```
lslpp -L cdrecord
```

9. To display the installation state for all the filesets contained in the Server bundle located at `/usr/sys/inst.data/sys_bundles/Server.bnd`, type:

```
lslpp -L -b Server
```

or:

```
lslpp -L -b /usr/sys/inst.data/sys_bundles/Server.bnd
```

Files

Item	Description
/etc/objrepos/history	Specifies installation and update history information of all software products on the root.
/usr/lib/objrepos/history	Specifies installation and update history information of all software products on the **/usr** file system.
/usr/share/lib/objrepos/history	Specifies installation and update history information of all software products on the **/usr/share** file system.
/etc/objrepos/lpp	Specifies installation information of all software products on the root.
/usr/lib/objrepos/lpp	Specifies installation information of all software products on the **/usr** file system.

Item	Description
/usr/share/lib/objrepos/lpp	Specifies installation information of all software products on the **/usr/share** file system.
/etc/objrepos/product	Specifies installation and update information of all software products on the root.
/usr/lib/objrepos/product	Specifies installation and update information of all software products on the **/usr** file system.
/usr/share/lib/objrepos/product	Specifies installation and update information of all the software products on the **/usr/share** file system.
/etc/objrepos/inventory	Specifies names and locations of files in a software product on the root.
/usr/lib/objrepos/inventory	Specifies names and locations of files in a software product on the **/usr** file system.
/usr/share/lib/objrepos/inventory	Specifies names and locations of files in a software product on the **/usr/share** file system.
`/usr/sys/inst.data/sys_bundles/` and `/usr/sys/inst.data/user_bundles/`	Known locations for bundle files. Bundle files should have a `.bnd` extension.

lslpracl Command

Purpose

Displays the access controls for a least-privilege (LP) resource.

Syntax

To display the access controls for an LP resource:

- On the local node:

  `lslpracl` [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-L`] [`-p`] [`-E`] [`-x`] [`-h`] [`-TV`] [*name*]

- On all nodes in a domain:

  `lslpracl -a` [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-L`] [`-p`] [`-E`] [`-x`] [`-h`] [`-TV`] [*name*]

- On a subset of nodes in a domain:

  `lslpracl` { `-n` *host1*[,*host2*,...] } [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-L`] [`-p`] [`-E`] [`-x`] [`-h`] [`-TV`] [*name*]

Description

The `lslpracl` command displays the access control list (ACL) that is associated with a least-privilege (LP) resource. The accesses contained in the ACL entries are displayed. The Resource ACL controls access to the LP resources. If no LP resource name is specified, the Resource ACLs for all LP resources are listed. By default, this command displays information in table format (`-t`).

This command displays the following ACL information:

Field	Description
Name	The name of the LP resource. See “lpacl Information” on page 1951 for a description of the network identity.

Field	Description
Identity	The network identity of the user.
Permissions	The permissions allowed for `Identity`. The valid values are: **a** Administrator permission **r** Read permission (consists of the e, l, q, and v permissions) **w** Write permission (consists of the c, d, o, and s permissions) **x** Execute permission **c** Refresh permission **d** Define and undefine permission **e** Event permission **l** Enumerate permission **o** Online, offline, and reset permission **q** Query permission **s** Set permission **v** Validate permission **0** No permission
NodeName	The location of the LP resource (for management domain scope or peer domain scope).
PeerDomain	The name of the RSCT peer domain in which the LP resource is defined. This field is displayed when the `-p` flag is specified.

If the Resource ACL indicates that the Resource Shared ACL controls access to the LP resource, the ID is displayed as `Uses Resource Shared ACL` and there is no permission value. Use the `-L` flag to display the Resource Shared ACL when it is used by the Resource ACLs that are being displayed.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the `-a` flag. If you want this command to run on a subset of nodes in a domain, use the `-n` flag. Otherwise, this command runs on the local node.

Parameters

name
Specifies the name of the LP resource.

Flags

-a
Displays the Resource ACLs on all nodes in the domain. The `CT_MANAGEMENT_SCOPE` environment variable setting determines the cluster scope. If `CT_MANAGEMENT_SCOPE` is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The `lslpracl` command runs once for the first valid scope that the LP resource manager finds. For example, suppose that a management domain and a peer domain exist and the `CT_MANAGEMENT_SCOPE` environment variable is not set. In this case, `lslpracl –a` runs in the management domain. To run `lslpracl –a` in the peer domain, you must set `CT_MANAGEMENT_SCOPE` to 2.

-i
Generates a template in a form that can be used, after appropriate editing, as file input to the `chlpracl` command.

-l
Displays the information about separate lines (long format).

-t
Displays the information in separate columns (table format). It is the default.

-d
Displays the information using delimiters. The default delimiter is a pipe symbol (`|`). Use the `-D` flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default pipe symbol (`|`) when the information you want to display contains pipe symbols, for example. You can use this flag to specify a delimiter of one or more characters.

-n *host1*[,*host2*,...]
Specifies the node in the domain from which the Resource ACL is displayed. By default, the Resource ACL is displayed on the local node. This flag is valid only in a management domain or a peer domain. If `CT_MANAGEMENT_SCOPE` is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope found.

-L
Displays the accesses of the Resource Shared ACL if the Resource ACL indicates that access is controlled by the Resource Shared ACL.

-p
Displays the name of the RSCT peer domain in which the LP resource is defined.

-E
Displays read permission as `elqv` instead of `r` and write permission as `cdos` instead of `w`.

-x
Excludes the header (suppresses header printing).

-h
Writes the command usage statement to standard output.

-T
Writes the command trace messages to standard error.

-V
Writes the command verbose messages to standard output.

Environment variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used, unless the -a flag or the -n flag is specified.

Standard output

When the -h flag is specified, this command usage statement is written to standard output. When the -V flag is specified, this command verbose messages are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
The resource was not found.

Security

To run the `lslpracl` command, you need:

- read permission in the Class ACL of the `IBM.LPCommands` resource class.
- read permission in the Resource ACL.

  As an alternative, the Resource ACL can direct the use of the Resource Shared ACL if this permission exists in the Resource Shared ACL.

Permissions are specified in the LP ACLs on the contacted system. See “lpacl Information” on page 1951 for general information about LP ACLs and the *Administering RSCT* guide for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for the AIX and Linux operating systems.

Location

/opt/rsct/bin/lslpracl

Examples

1. To list the Resource ACLs for the LP resource `lpcommand1` on nodeA in table format, run this command on nodeA:

   ```
   lslpracl lpcommand1
   ```

   The following output is displayed:

   ```
   Resource ACLs for LPRM
   Name            Identity                Permissions     NodeName
   lpcommand1      joe@LOCALHOST           rx              nodeA
   lpcommand1      bill@0x374bdcbe384ed38a rx              nodeA
   lpcommand1      jane@0x374bdcbe384ed38a rwax            nodeA
   ```

2. To list the Resource ACLs for the LP resource `lpcommand1` on nodeA in long format, run this command on nodeA:

   ```
   lslpracl -l lpcommand1
   ```

   The following output is displayed:

   ```
   Resource ACLs for LPRM
   Name lpcommand1, NodeName nodeA
       Identity    =     joe@LOCALHOST
       Permissions =     rx

       Identity    =     bill@0x374bdcbe384ed38a
       Permissions =     rx

       Identity    =     jane@0x374bdcbe384ed38a
       Permissions =     rwax
   ```

3. To list the Resource ACLs for the LP resource `lpcommand1` on nodeA in delimited format, run this command on nodeA:

   ```
   lslpracl -d lpcommand1
   ```

   The following output is displayed:

   ```
   Resource ACLs for LPRM
   Name|Identity|Permissions|NodeName
   lpcommand1|joe@LOCALHOST|rx|nodeA
   ```

```
lpcommand1|bill@0x374bdcbe384ed38a|rx|nodeA
lpcommand1|jane@0x374bdcbe384ed38a|rwax|nodeA
```

4. To list the Resource ACLs for the LP resource `lpcommand1` in the active domain, run this command on nodeA:

```
lslpracl -a lpcommand1
```

The following output is displayed:

```
Resource ACLs for LPRM
Name            Identity                Permissions     NodeName
lpcommand1      joe@LOCALHOST           rx              nodeA.pok.ibm.com
lpcommand1      bill@0x374bdcbe384ed38a rx              nodeA.pok.ibm.com
lpcommand1      jane@0x374bdcbe384ed38a rwax            nodeA.pok.ibm.com
lpcommand1      joe@LOCALHOST           rx              nodeB.pok.ibm.com
lpcommand1      jane@0x374bdcbe384ed38a rwax            nodeB.pok.ibm.com
```

5. To list the Resource ACLs for all LP resources on nodeA, run this command on nodeA:

```
lslpracl
```

The following output is displayed:

```
Resource ACLs for LPRM
Name            Identity                Permissions     NodeName
lpcommand1      joe@LOCALHOST           rx              nodeA
lpcommand1      bill@0x374bdcbe384ed38a rx              nodeA
lpcommand1      jane@0x374bdcbe384ed38a rwax            nodeA
lpcommand2      jim@LOCALHOST           rx              nodeA
lpcommand2      jane@0x374bdcbe384ed38a rwax            nodeA
lpcommand3      mary                    rwax            nodeA
lpcommand4      bob@LOCALHOST           rx              nodeA
lpcommand4      sam@0x374bdcbe384ed38a  rwax            nodeA
```

6. To list the Resource ACLs for the LP resource `lpcommand1` in the active domain and list the peer domain name, run this command on nodeA:

```
lslpracl -ap lpcommand1
```

The following output is displayed:

```
Resource ACLs for LPRM
Name            Identity                Permission NodeName             PeerDomain
lpcommand1      joe@LOCALHOST           rx         nodeA.pok.ibm.com    PD1
lpcommand1      bill@0x374bdcbe384ed38a rx         nodeA.pok.ibm.com    PD1
lpcommand1      jane@0x374bdcbe384ed38a rwax       nodeA.pok.ibm.com    PD1
lpcommand1      joe@LOCALHOST           rx         nodeB.pok.ibm.com    PD1
lpcommand1      jane@0x374bdcbe384ed38a rwax       nodeB.pok.ibm.com    PD1
```

7. To list the Resource ACLs for the LP resource `lpcommand2` on nodeA, run this command on nodeA:

```
lslpracl lpcommand2
```

The following output is displayed:

```
Resource ACLs for LPRM
Name            Identity                   Permissions   NodeName
lpcommand2      Uses Resource Shared ACL                 nodeA
```

8. To list the Resource ACLs for the LP resource `lpcommand2` on nodeA, and show the Resource Shared ACL if it is used, run this command on nodeA:

```
lslpracl -L lpcommand2
```

The following output is displayed:

```
Resource ACLs for LPRM
Name            Identity                Permissions     NodeName
```

```
lpcommand2      bill@0x374bdcbe384ed38a rx              nodeA
lpcommand2      jane@0x374bdcbe384ed38a rwax            nodeA
```

lslpriacl Command

Purpose

Displays the access controls for the least-privilege (LP) Resource Initial ACL.

Syntax

To display the access controls for the Resource Initial ACL:

- On the local node:

  `lslpriacl` [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-p`] [`-E`] [`-x`] [`-h`] [`-TV`]

- On all nodes in a domain:

  `lslpriacl -a` [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-p`] [`-E`] [`-x`] [`-h`] [`-TV`]

- On a subset of nodes in a domain:

  `lslpriacl` { `-n` *host1*[`,`*host2*`,`...] } [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-p`] [`-E`] [`-x`] [`-h`] [`-TV`]

Description

The `lslpriacl` command displays the access control list (ACL) that is associated with the least-privilege (LP) Resource Initial ACL. The accesses contained in the ACL entries are displayed. The Resource Initial ACL is used as the Initial ACL that gets copied to the Resource ACL when an LP resource is created. By default, this command displays information in table format (`-t`).

This command displays the following ACL information:

Field	Description
Identity	The network identity of the user. See “lpacl Information” on page 1951 for a description of the network identity.

Field	Description
Permissions	The permissions allowed for `Identity`. The valid values are: **a** Administrator permission **r** Read permission (consists of the e, l, q, and v permissions) **w** Write permission (consists of the c, d, o, and s permissions) **x** Execute permission **c** Refresh permission **d** Define and undefine permission **e** Event permission **l** Enumerate permission **o** Online, offline, and reset permission **q** Query permission **s** Set permission **v** Validate permission **0** No permission
NodeName	The location of the `IBM.LPCommands` resource class (for management domain scope or peer domain scope).
PeerDomain	The name of the RSCT peer domain in which the `IBM.LPCommands` resource class is defined. This field is displayed when the `-p` flag is specified.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the `-a` flag. If you want this command to run on a subset of nodes in a domain, use the `-n` flag. Otherwise, this command runs on the local node.

Flags

-a

Displays the Resource Initial ACLs on all nodes in the domain. The `CT_MANAGEMENT_SCOPE` environment variable setting determines the cluster scope. If `CT_MANAGEMENT_SCOPE` is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The `lslpriacl` command runs once for the first valid scope that the LP resource manager finds. For example, suppose that a management domain and a peer domain exist and the `CT_MANAGEMENT_SCOPE` environment variable is not set. In this case, `lslpriacl –a` runs

in the management domain. To run `lslpriacl –a` in the peer domain, you must set `CT_MANAGEMENT_SCOPE` to 2.

-i
Generates a template in a form that can be used, after appropriate editing, as file input to the `chlpriacl` command.

-l
Displays the information about separate lines (long format).

-t
Displays the information in separate columns (table format). This is the default.

-d
Displays the information using delimiters. The default delimiter is a pipe symbol (`|`). Use the `-D` flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default pipe symbol (`|`) when the information you want to display contains pipe symbols, for example. You can use this flag to specify a delimiter of one or more characters.

-n *host1*[,*host2*,...]
Specifies the node in the domain from which the Resource Initial ACL is displayed. By default, the Resource Initial ACL is displayed on the local node. This flag is valid only in a management domain or a peer domain. If `CT_MANAGEMENT_SCOPE` is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope found.

-p
Displays the name of the RSCT peer domain in which the `IBM.LPCommands` resource class is defined.

-E
Displays read permission as `elqv` instead of `r` and write permission as `cdos` instead of `w`.

-x
Excludes the header (suppresses header printing).

-h
Writes the command usage statement to standard output.

-T
Writes the command trace messages to standard error.

-V
Writes the command verbose messages to standard output.

Environment variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When `CT_CONTACT` is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If `CT_CONTACT` is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used, unless the `-a` flag or the `-n` flag is specified.

Standard output

When the `-h` flag is specified, this command usage statement is written to standard output. When the `-V` flag is specified, this command verbose messages are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
The resource was not found.

Security

To run the `lslpriacl` command, you need read permission in the Class ACL of the `IBM.LPCommands` resource class. Permissions are specified in the LP ACLs on the contacted system. See "lpacl Information" on page 1951 for general information about LP ACLs and the *Administering RSCT* guide for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for the AIX and Linux operating systems.

Location

/opt/rsct/bin/lslpriacl

Examples

1. To list the Resource Initial ACLs on nodeA in table format, run this command on nodeA:

```
lslpriacl
```

   The following output is displayed:

```
Resource Initial ACLs for LPRM
Identity                        Permissions     NodeName
joe@LOCALHOST                   rx              nodeA
bill@0x374bdcbe384ed38a         rwx             nodeA
jane@0x374bdcbe384ed38a         rwax            nodeA
```

2. To list the Resource Initial ACLs on nodeA in long format, run this command on nodeA:

```
lslpriacl -l
```

   The following output is displayed:

```
Resource Initial ACLs for LPRM
NodeName c175n06.ppd.pok.ibm.com
   Identity    =     joe@LOCALHOST
   Permissions =     rx

   Identity    =     bill@0x374bdcbe384ed38a
   Permission  =     rwx

   Identity    =     jane@0x374bdcbe384ed38a
   Permissions =     rwax
```

3. To list the Resource Initial ACLs on nodeA in delimited format, run this command on nodeA:

```
lslpriacl -d
```

   The following output is displayed:

```
Resource Initial ACLs for LPRM
Identity|Permissions|NodeName
joe@LOCALHOST|rx|nodeA
bill@0x374bdcbe384ed38a|rwx|nodeA
jane@0x374bdcbe384ed38a|rwax|nodeA
```

4. To list the Resource Initial ACLs in the active domain, run this command:

```
lslpriacl -a
```

   The following output is displayed:

```
Resource Initial ACLs for LPRM
Identity                        Permissions     NodeName
joe@LOCALHOST                   rx              nodeA.pok.ibm.com
bill@0x374bdcbe384ed38a         rwx             nodeA.pok.ibm.com
jane@0x374bdcbe384ed38a         rwax            nodeA.pok.ibm.com
joe@LOCALHOST                   rx              nodeB.pok.ibm.com
jane@0x374bdcbe384ed38a         rwax            nodeB.pok.ibm.com
```

5. To list the Resource Initial ACLs in the active domain and list the peer domain name, run this command:

```
lslpriacl -ap
```

   The following output is displayed:

```
Resource Initial ACLs for LPRM
Identity                        Permissions     NodeName                PeerDomain
joe@LOCALHOST                   rx              nodeA.pok.ibm.com       PD1
bill@0x374bdcbe384ed38a         rwx             nodeA.pok.ibm.com       PD1
jane@0x374bdcbe384ed38a         rwax            nodeA.pok.ibm.com       PD1
joe@LOCALHOST                   rx              nodeB.pok.ibm.com       PD1
jane@0x374bdcbe384ed38a         rwax            nodeB.pok.ibm.com       PD1
```

lslprsacl Command

Purpose

Displays the access controls for the least-privilege (LP) Resource Shared ACL.

Syntax

To display the access controls for the Resource Shared ACL:

- On the local node:

  `lslprsacl [ -l | -i | -t | -d | -D` *delimiter* `] [-p] [-E] [-x] [-h] [-TV]`

- On all nodes in a domain:

  `lslprsacl -a [ -l | -i | -t | -d | -D` *delimiter* `] [-p] [-E] [-x] [-h] [-TV]`

- On a subset of nodes in a domain:

  `lslprsacl { -n` *host1*`[,`*host2*`,... ] } [ -l | -i | -t | -d | -D` *delimiter* `] [-p] [-E] [-x] [-h] [-TV]`

Description

The `lslprsacl` command displays the access control list (ACL) that is associated with the least-privilege (LP) Resource Shared ACL. The accesses contained in the ACL entries are displayed. The Resource Shared ACL controls access to LP resources in which the Resource ACL indicates that the Resource Shared ACL is used. By default, this command displays information in table format (`-t`).

This command displays the following ACL information:

Field	Description
Identity	The network identity of the user. See “lpacl Information” on page 1951for a description of the network identity.

Field	Description
Permissions	The permissions allowed for `Identity`. The valid values are: **a** Administrator permission **r** Read permission (consists of the e, `l`, q, and v permissions) **w** Write permission (consists of the c, d, o, and s permissions) **x** Execute permission **c** Refresh permission **d** Define and undefine permission **e** Event permission **l** Enumerate permission **o** Online, offline, and reset permission **q** Query permission **s** Set permission **v** Validate permission **0** No permission
NodeName	The location of the `IBM.LPCommands` resource class (for management domain scope or peer domain scope).
PeerDomain	The name of the RSCT peer domain in which the `IBM.LPCommands` resource class is defined. This field is displayed when the `-p` flag is specified.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the `-a` flag. If you want this command to run on a subset of nodes in a domain, use the `-n` flag. Otherwise, this command runs on the local node.

Flags

-a

Displays Resource Shared ACLs on all nodes in the domain. The `CT_MANAGEMENT_SCOPE` environment variable setting determines the cluster scope. If `CT_MANAGEMENT_SCOPE` is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The `lslprsacl` command runs once for the first valid scope that the LP resource manager finds. For example, suppose that a management domain and a peer domain exist and the `CT_MANAGEMENT_SCOPE` environment variable is not set. In this case, `lslprsacl –a` runs

in the management domain. To run `lslprsacl -a` in the peer domain, you must set `CT_MANAGEMENT_SCOPE` to 2.

-i
Generates a template in a form that can be used, after appropriate editing, as file input to the `chlprsacl` command.

-l
Displays the information on separate lines (long format).

-t
Displays the information in separate columns (table format). This is the default.

-d
Displays the information using delimiters. The default delimiter is a pipe symbol (`|`). Use the `-D` flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default pipe symbol (`|`) when the information you want to display contains pipe symbols, for example. You can use this flag to specify a delimiter of one or more characters.

-n *host1*[,*host2*,...]
Specifies the node in the domain from which the Resource Shared ACL is displayed. By default, the Resource Shared ACL is displayed on the local node. This flag is valid only in a management domain or a peer domain. If `CT_MANAGEMENT_SCOPE` is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope found.

-p
Displays the name of the RSCT peer domain in which the `IBM.LPCommands` resource class is defined.

-E
Displays read permission as `elqv` instead of `r` and write permission as `cdos` instead of `w`.

-x
Excludes the header (suppresses header printing).

-h
Writes the command usage statement to standard output.

-T
Writes the command trace messages to standard error.

-V
Writes the command verbose messages to standard output.

Environment variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When `CT_CONTACT` is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If `CT_CONTACT` is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used, unless the `-a` flag or the `-n` flag is specified.

Standard output

When the `-h` flag is specified, this command usage statement is written to standard output. When the `-V` flag is specified, this command verbose messages are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
The resource was not found.

Security

To run the `lslprsacl` command, you need read permission in the Class ACL of the `IBM.LPCommands` resource class. Permissions are specified in the LP ACLs on the contacted system. See "lpacl Information" on page 1951 for general information about LP ACLs and the *Administering RSCT* guide for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for the AIX and Linux operating systems.

Location

/opt/rsct/bin/lslprsacl

Examples

1. To list the Resource Shared ACLs on nodeA in table format, run this command on nodeA:

```
lslprsacl
```

   The following output is displayed:

```
Resource Shared ACLs for LPRM
Identity                        Permissions     NodeName
joe@LOCALHOST                   rx              nodeA
bill@0x374bdcbe384ed38a         rwx             nodeA
jane@0x374bdcbe384ed38a         rwax            nodeA
```

2. To list the Resource Shared ACLs on nodeA in long format, run this command on nodeA:

```
lslprsacl -l
```

   The following output is displayed:

```
Resource Shared ACLs for LPRM
NodeName c175n06.ppd.pok.ibm.com
   Identity    =     joe@LOCALHOST
   Permissions =     rx

   Identity    =     bill@0x374bdcbe384ed38a
   Permissions =     rwx

   Identity    =     jane@0x374bdcbe384ed38a
   Permissions =     rwax
```

3. To list the Resource Shared ACLs on nodeA in delimited format, run this command on nodeA:

```
lslprsacl -d
```

   The following output is displayed:

```
Resource Shared ACLs for LPRM
Identity|Permissions|NodeName
joe@LOCALHOST|rx|nodeA
bill@0x374bdcbe384ed38a|rwx|nodeA
jane@0x374bdcbe384ed38a|rwax|nodeA
```

4. To list the Resource Shared ACLs in the active domain, run this command:

```
lslprsacl -a
```

   The following output is displayed:

```
Identity                        Permissions     NodeName
joe@LOCALHOST                   rx              nodeA.pok.ibm.com
bill@0x374bdcbe384ed38a         rwx             nodeA.pok.ibm.com
jane@0x374bdcbe384ed38a         rwax            nodeA.pok.ibm.com
joe@LOCALHOST                   rx              nodeB.pok.ibm.com
jane@0x374bdcbe384ed38a         rwax            nodeB.pok.ibm.com
```

5. To list the Resource Shared ACLs in the active domain and list the peer domain name, run this command:

```
lslprsacl -ap
```

   The following output is displayed:

```
Resource Shared ACLs for LPRM
Identity                        Permissions     NodeName                PeerDomain
joe@LOCALHOST                   rx              nodeA.pok.ibm.com       PD1
bill@0x374bdcbe384ed38a         rwx             nodeA.pok.ibm.com       PD1
jane@0x374bdcbe384ed38a         rwax            nodeA.pok.ibm.com       PD1
joe@LOCALHOST                   rx              nodeB.pok.ibm.com       PD1
jane@0x374bdcbe384ed38a         rwax            nodeB.pok.ibm.com       PD1
```

lslv Command

Purpose

Displays information about a logical volume.

Syntax

To Display Logical Volume Information

lslv [**-L**] [**-l**| **-m**] [**-n** *PhysicalVolume*] *LogicalVolume*

To Display Logical Volume Allocation Map

lslv [**-L**] [**-n** *PhysicalVolume*] **-p** *PhysicalVolume* [*LogicalVolume*]

Description

The **lslv** command displays the characteristics and status of the *LogicalVolume* or lists the logical volume allocation map for the physical partitions on the *PhysicalVolume*. The logical volume can be a name or identifier.

Note: If the **lslv** command cannot find information for a field in the Device Configuration Database, it will insert a question mark (?) in the value field. As an example, if there is no information for the LABEL field, the following is displayed:

```
LABEL: ?
```

The command attempts to obtain as much information as possible from the description area when it is given a logical volume identifier.

You can use the System Management Interface Tool (SMIT) **smit lslv** fast path to run this command.

Flags

Item	Description
-L	Specifies no waiting to obtain a lock on the Volume group. **Note:** If the volume group is being changed, using the **-L** flag gives unreliable date.

Item	Description
-l	Lists the following fields for each physical volume in the logical volume: **PV** Physical volume name. **Copies** The following three fields: • The number of logical partitions containing at least one physical partition (no copies) on the physical volume • The number of logical partitions containing at least two physical partitions (one copy) on the physical volume • The number of logical partitions containing three physical partitions (two copies) on the physical volume **In band** The percentage of physical partitions on the physical volume that belong to the logical volume and were allocated within the physical volume region specified by Intra-physical allocation policy. **Distribution** The number of physical partitions allocated within each section of the physical volume: outer edge, outer middle, center, inner middle, and inner edge of the physical volume.
-m	Lists the following fields for each logical partition: **LPs** Logical partition number. **PV1** Physical volume name where the logical partition's first physical partition is located. **PP1** First physical partition number allocated to the logical partition. **PV2** Physical volume name where the logical partition's second physical partition (first copy) is located. **PP2** Second physical partition number allocated to the logical partition. **PV3** Physical volume name where the logical partition's third physical partition (second copy) is located. **PP3** Third physical partition number allocated to the logical partition.
-n *PhysicalVolume*	Accesses information from the specific descriptor area of *PhysicalVolume* variable. The information may not be current since the information accessed with the -n flag has not been validated for the logical volumes. If you do not use the -n flag, the descriptor area from the physical volume that holds the validated information is accessed and therefore the information that is displayed is current. The volume group need not be active when you use this flag.

Item	Description
-p *PhysicalVolume*	Displays the logical volume allocation map for the *PhysicalVolume* variable. If you use the *LogicalVolume* parameter, any partition allocated to that logical volume is listed by logical partition number. Otherwise, the state of the partition is listed as one of the following: **used** Indicates that the partition is allocated to another logical volume. **free** Indicates that the specified partition is not being used on the system. **stale** Indicates that the specified partition is no longer consistent with other partitions. The computer lists the logical partitions number with a question mark if the partition is stale.

If no flags are specified, the following status is displayed:

Item	Description
LOGICAL VOLUME	Name of the logical volume. Logical volume names must be unique system-wide and can range from 1 to 15 characters.
VOLUME GROUP	Name of the volume group. Volume group names must be unique system-wide and can range from 1 to 15 characters.
LOGICAL VOLUME IDENTIFIER (LV identifier)	Identifier of the logical volume.
PERMISSION	Access permission; read-only or read-write.
VOLUME GROUP STATE (VG state)	State of the volume group. If the volume group is activated with the **varyonvg** command, the state is either active/complete (indicating all physical volumes are active) or active/partial (indicating all physical volumes are not active). If the volume group is not activated with the **varyonvg** command, the state is inactive.
LOGICAL VOLUME STATE (LV state)	State of the logical volume. The Opened/stale status indicates the logical volume is open but contains physical partitions that are not current. Opened/syncd indicates the logical volume is open and synchronized. Closed indicates the logical volume has not been opened.
TYPE	Logical volume type.
WRITE VERIFY	Write verify state of on or off.
MIRROR WRITE CONSISTENCY	Mirror write consistency state of on or off.
MAX LPs	Maximum number of logical partitions the logical volume can hold.
PP size	Size of each physical partition.
COPIES	Number of physical partitions created for each logical partition when allocating.
SCHEDULE POLICY(Sched policy)	Sequential or parallel scheduling policy.
LPs	Number of logical partitions currently in the logical volume.
PPs	Number of physical partitions currently in the logical volume.

Item	Description
Stale partitions	Number of physical partitions in the logical volume that are not current.
BB POLICY	Bad block relocation policy.
INTER-POLICY	Inter-physical allocation policy.
INTRA-POLICY	Intra-physical allocation policy.
UPPER BOUND	If the logical volume is super strict, upper bound is the maximum number of disks in a mirror copy.
RELOCATABLE	Indicates whether the partitions can be relocated if a reorganization of partition allocation takes place.
MOUNT POINT	File system mount point for the logical volume, if applicable.
LABEL	Specifies the label field for the logical volume.
Each LP copy on a separate PV?	The strictness value. Current state of allocation, strict, nonstrict, or superstrict. A strict allocation states that no copies for a logical partition are allocated on the same physical volume. If the allocation does not follow the strict criteria, it is called nonstrict. A nonstrict allocation states that at least one occurrence of two physical partitions belong to the same logical partition. A superstrict allocation states that no partition from one mirror copy may reside the same disk as another mirror copy.
SERIALIZE IO?	Serialization of overlapping IOs state of yes or no. If serialization is turned on (yes), then overlapping IOs are not allowed on a block range, and only a single IO in a block range is processed at any one time. Most applications, such as file systems and databases, perform serialization; therefore, serialization should be turned off (no). The default setting for new logical volumes is no.
STRIPE WIDTH	The number of physical volumes being striped across.
STRIPE SIZE	The number of bytes per stripe.
INFINITE RETRY	Lists the infinite retry option of the logical volume.
PREFERRED READ	Lists the preferred logical volume copy for the read operation.
ENCRYPTION	Lists the data encryption option of the logical volume. This information is available in AIX 7 with 7200-05, or later.

Examples

1. To display information about the testlv logical volume, enter:

```
lslv testlv
```

Information about the testlv logical volume, its logical and physical partitions, and the volume group to which it belongs is displayed as shown in the following example output:

```
LOGICAL VOLUME:     testlv                 VOLUME GROUP:   testvg
LV IDENTIFIER:      000e8b6e0000d900000001476c303bc8.1 PERMISSION:     read/write
VG STATE:           active/complete        LV STATE:       closed/syncd
TYPE:               jfs                    WRITE VERIFY:   off
MAX LPs:            512                    PP SIZE:        128 megabyte(s)
COPIES:             3                      SCHED POLICY:   parallel
LPs:                10                     PPs:            30
```

```
STALE PPs:          0                       BB POLICY:      relocatable
INTER-POLICY:       minimum                 RELOCATABLE:    yes
INTRA-POLICY:       middle                  UPPER BOUND:    32
MOUNT POINT:        N/A                     LABEL:          None
MIRROR WRITE CONSISTENCY: on/ACTIVE
EACH LP COPY ON A SEPARATE PV ?: yes
Serialize IO ?:     NO
INFINITE RETRY:     no                      PREFERRED READ: 3
```

2. To display the logical volume allocation map for hdisk2, enter:

```
lslv -p hdisk2
```

An allocation map for hdisk2 is displayed, showing the state of each partition. Since no *LogicalVolume* parameter was included, the map does not contain logical partition numbers specific to any logical volume.

3. To display information about logical volume lv03 by physical volume, enter:

```
lslv -l lv03
```

The characteristics and status of lv03 are displayed, with the output arranged by physical volume.

4. To display information about physical volume hdisk3 gathered from the descriptor area on hdisk2, enter:

```
lslv -n hdisk2 -p hdisk3 lv02
```

An allocation map, using the descriptor area on hdisk2, is displayed. Because the *LogicalVolume* parameter is included, the number of each logical partition allocated to that logical volume is displayed on the map.

5. To display information about a specific logical volume, using the identifier, enter:

```
lslv 00000256a81634bc.2
```

All available characteristics and status of this logical volume are displayed.

File

Item	Description
/usr/sbin	Contains the **lslv** command.

lsmaster Command

Purpose

Displays the characteristics for the configuration of an NIS controller server.

Syntax

/usr/sbin/lsmaster [**-c** | **-l**]

Description

The **lsmaster** command displays the characteristics of an NIS controller server. The host names of the worker servers are listed along with the currently served domains.

You can use the System Management Interface Tool (SMIT) **smit lsmaster** fast path to run this command.

Flags

Item	Description
-c	Specifies that the output should be in colon format. This flag is the default.
-l	Specifies that the output should be in list format.

Examples

To list the NIS controller server characteristics in colon format, enter:

```
lsmaster -c
```

Files

Item	Description
/var/yp/*domainname* directory	Contains the NIS maps for the NIS domain.

lsmcode Command

Purpose

Displays microcode and firmware levels of the system and adapters and devices.

Syntax

lsmcode [**-A** | **-d** *Name*] [**-r** | **-c**] [**-t** [**service** | **system** | **adapter** | **adapter-boot** | **raid-dasd** | **backplane**]]

Description

The **lsmcode** command when run without any flags, displays the platform system firmware microcode level and the service processor microcode levels, if supported. Not all systems contain a service processor, nor do all systems support displaying the system processor level. Information on a specific device is displayed with the **-d** flag.

If you run the **lsmcode** command with the **-r** or **-c** flag, it displays the microcode levels in a **printf** format; that is, not a menu. This method is preferred if running **lsmcode** from a script.

Flags

Item	Description
-A	Displays microcode level information for all supported devices. Using this flag assumes the **-r** flag.
-c	Displays the microcode/firmware levels without using menus.
-d *Name*	Displays microcode level information for the named device.

Item	Description
-r	Displays the microcode/firmware levels in a tabular format. The microcode level is preceded by a Type if supported or required. Current supported Types are as follows: **system** System Firmware **service** Service Processor **adapter** Adapter Functional microcode **adapter-boot** Adapter Boot Microcode **raid-dasd** DASD Microcode in a RAID array **backplane** Backplane Microcode in a RAID subsystem
-t	Specifies the microcode type. The microcode level information of the specified type is displayed. You can use the **-t** flag only when a device supports multiple types. You can use the **-A** flag to find any devices supporting multiple types.

Examples

1. To display the system firmware level and service processor (if present), type:

   ```
   lsmcode -c
   ```

   The system displays a message similar to the following:

   ```
   System Firmware level is TCP99256
   ```

2. To display the system firmware level and service processor (if present) in raw mode, type:

   ```
   lsmcode -r
   ```

   The system displays a message similar to the following:

   ```
   system:TCP99256
   ```

3. To display the adapter microcode levels for a RAID adapter **scraid0**, type:

   ```
   lsmcode -r -d scraid0
   ```

   The system displays a message similar to the following:

   ```
   adapter:4.20.18|adapter-boot:4.00.26
   raid-dasd:22:FFC #:DDYS-T0.524D3031.53393446
   raid-dasd:26:FFC #:DDYS-T0.524D3031.53393446
   raid-dasd:2e:FFC #:DDYS-T0.525A3034.53393243
   ```

4. To display the microcode level for a tape drive **rmt0**, type:

   ```
   lsmcode -r -d rmt0
   ```

   The system displays a message similar to the following:

   ```
   C009
   ```

5. To display the microcode level for all supported devices, type:

```
lsmcode -A
```

The system displays a message similar to the following:

```
sys0!system:TCP99256
rmt0!C009
scraid0!adapter:4.20.18|adapter-boot:4.00.26
raid-dasd:22:FFC #:DDYS-T0.524D3031.53393446
raid-dasd:26:FFC #:DDYS-T0.524D3031.53393446
raid-dasd:2e:FFC #:DDYS-T0.525A3034.53393243
.....
```

6. To display the microcode level of the **adapter** microcode type for a RAID adapter **scraid0**, type:

```
lsmcode -rd scraid0 -t adapter
```

The system displays a message similar to the following:

```
adapter:4.50.01
```

Files

Item	Description
/usr/sbin/lsmcode	Contains the **lsmcode** command.

Related Information

The **diag** command.

lsmksysb Command

Purpose

Lists or restores the contents of a volume group backup on a specified media.

Syntax

lsmksysb [**-b** *blocks*] [**-f** *device*] [**-a**] [**-c**] [**-l**] [**-n**] [**-r**] [**-s**] [**-d** *path*] [**-B**] [**-D**] [**-L**] [**-V**] [*file_list*]

Description

The **lsmksysb** command lists the contents of a volume group backup from tape, file, CD-ROM, or other source and can be used to restore files from a valid backup source. The **lsmksysb** command also works for multi-volume backups such as multiple CDs, DVDs, USB disks, or tapes.

The **lsmksysb -r** and **restorevgfiles** commands perform identical operations and must be considered interchangeable.

Flags

Item	Description
-a	Verifies the physical block size of the tape backup, as specified by the **-b** *block* flag. You might need to alter the block size if necessary to read the backup. The **-a** flag is valid only when a tape backup is used.
-b *blocks*	Specifies the number of 512-byte blocks to read in a single input operation, as defined by the *blocks* parameter. If the *blocks* parameter is not specified, the number of blocks read defaults to 100.

Item	Description
-B	Prints the volume group backup log to **stdout**. This flag displays the past 256 backups (roughly). The log is in alog format and is kept in **/var/adm/ras/vgbackuplog**. Each line of the log is a semicolon-separated list of the file or device name, the command that is used to make backup, date, shrink size, full size of the backup, and recommended maintenance or technology level (if any). **Note:** The shrink size is the size of the data on all file systems. The full size is total size of each file system (unused + data).
-c	Produces colon-separated output. This flag works only with the **-l** and **-L** flags.
-d *path*	Specifies the directory path to which the files are restored, as defined by the *path* parameter. If the **-d** parameter is not used, the current working directory is used. This can be a problem if the current working directory is root. We recommend writing to a temporary folder instead of to root.
-D	Produces debug output.
-f *device*	Specifies the type of device containing the backup (file, tape, CD-ROM, or other source) as defined by the *device* parameter. When **-f** is not specified, *device* defaults to **/dev/rmt0**.
-l	Displays useful information about a volume group backup. This flag requires the **-f** *device* flag. This flag causes **lsmksysb** to display information such as volume group, date and time backup was made, uname output from backed up system, oslevel, recommended maintenance or technology level, backup size in megabytes, and backup shrink size in megabytes. The shrink size is the size of the data on all file systems. The full size is the total size of each file system (unused + data). The **-l** flag also displays the logical volume and file system information of the backed up volume group, equivalent to running "**lsvg -l** *vgname*".
-L	Displays lpp fileset information about a **mksysb** backup only. This flag requires the **-f** *device* flag and displays the equivalent information to that produced by invoking "**lslpp -l**" on the running backed up system. This flag does not produce output about any volume group backup other than that produced by **mksysb**.
-n	Does not restore ACLs, PCLs, or extended attributes
-r	Specifies to restore the backup files, as defined by the *file-list* parameter. If the *file-list* parameter is not specified, then all files in the backup are restored. If the **-r** flag is not used, then executing the **lsmksysb** command lists only the files in the specified backup.
-s	Specifies that the backup source is a user volume group and not rootvg.
-V	Verifies a tape backup. This flag requires the **-f** *device* flag and works for tape devices only. The **-V** flag causes **lsmksysb** to verify the readability of the header of each file on the volume group backup and print any errors that occur to **stderr**.

Parameters

Item	Description
file_list	Identifies the list of files to be restored. This parameter is used only when the **-r** flag is specified. The full path of the files relative to the current directory must be specified in the space-separated list. All files in the specified directory are restored unless otherwise directed. If you are restoring all files in a directory, we recommend writing to a temporary folder instead of to root.

Examples

1. To list the contents of the system backup that is on the default device **/dev/rmt0**, enter the following command:

   ```
   lsmksysb
   ```

2. To list the contents of the system backup that is on device **/dev/cd1**, enter the following command:

   ```
   lsmksysb -f /dev/cd1
   ```

3. To list the contents of the system backup that is on device **/dev/cd1**, which is a user volume group that is not rootvg, enter the following command:

   ```
   lsmksysb -f /dev/cd1 -s
   ```

4. To restore **/etc/filesystems** from the system backup that is on device **/dev/cd1**, enter the following command:

   ```
   lsmksysb -f /dev/cd1 -r ./etc/filesystems
   ```

5. To restore all files in the **/myfs/test** directory of the non-rootvg backup, which is on device **/dev/cd1**, and write the restored files to **/data/myfiles**, enter the following command:

   ```
   lsmksysb -f /dev/cd1 -r -s -d /data/myfiles ./myfs/test
   ```

6. To display colon separated lpp information about a **mksysb** backup tape that is on **/dev/rmt0**, enter the following command:

   ```
   lsmksysb -Lc -f /dev/rmt0
   ```

7. To display the backup log of the volume group to **stdout**, enter the following command:

   ```
   lsmksysb -B
   ```

8. To list volume group and general backup data about a backup that is on **/tmp/mybackup**, enter the following command:

   ```
   lsmksysb -l -f /tmp/mybackup
   ```

9. To verify the readability of each header on a volume group backup tape in **/dev/rmt0**, enter the following command:

   ```
   lsmksysb -V -f /dev/rmt0
   ```

10. To list the contents of the system backup that is on device **/dev/usbms0,** enter the following command:

    ```
    lsmksysb –f /dev/usbms0
    ```

Files

Item	Description
/usr/bin/lsmksysb	Contains the **lsmksysb** command

lsmp Command

Purpose

Displays mirror pool information.

Syntax

lsmp [**-L**] [**-A**] [**-n** *descriptorphysicalvolume*] [**-m** *mirrorpoolname*] *vgname*

Description

The **lsmp** command displays mirror pool information for a volume group.

Note: The volume group must be varied on to list the mirror pool information. If the volume group is varied off, you must specify the source disk name using the **-n** flag to list the information.

Flags

Item	Description
-A	Displays information specific to asynchronous mirroring, if it exists.
-L	Specifies no waiting to obtain a lock on the volume group. **Note:** If the volume group is being changed, using the **-L** flag gives unreliable data.
-m *mirrorpoolname*	Displays mirror pool information only for the mirror pool that is specified by the *mirrorpoolname* parameter. If you do not specify the flag, information for all mirror pools that belong to the volume group is displayed.
-n *descriptorphysicalvolume*	Accesses information from the descriptor area that is specified by the *descriptorphysicalvolume* variable. The information might not be current because the information accessed with the **-n** flag has not been validated for the logical volumes. If you do not use the **-n** flag, the descriptor area from the physical volume that holds the validated information is accessed, and therefore the current information is displayed. The volume group does not need to be active when you use the **-n** flag.

Parameters

Item	Description
vgname	Specifies the volume group name.

Examples

1. To display all mirror pool information for a volume group, enter the following command:

   ```
   lsmp vg1
   ```

   The following output is displayed:

```
VOLUME GROUP:      vg1                  Mirror Pool Super Strict: no

MIRROR POOL:       mp1                  Mirroring Mode:    SYNC
MIRROR POOL:       mp2                  Mirroring Mode:    SYNC
```

2. To display all mirror pool information for a volume group and include asynchronous mirroring information in the output, enter the following command:

```
lsmp -A glvm_vg
```

The following output is displayed:

```
VOLUME GROUP:      glvm_vg              Mirror Pool Super Strict: yes

MIRROR POOL:       mp_bvr               Mirroring Mode:    ASYNC
ASYNC MIRROR STATE:  inactive           ASYNC CACHE LV:      mp_pok_lv
ASYNC CACHE VALID:   yes                ASYNC CACHE EMPTY:   yes
ASYNC CACHE HWM:     60                 ASYNC DATA DIVERGED: no

MIRROR POOL:       mp_pok               Mirroring Mode:    ASYNC
ASYNC MIRROR STATE:  active             ASYNC CACHE LV:      bvr_pok_lv
ASYNC CACHE VALID:   yes                ASYNC CACHE EMPTY:   no
ASYNC CACHE HWM:     90                 ASYNC DATA DIVERGED: no
```

3. To display information for the mirror pool `mp_pok` from the `glvm_vg` volume group and include asynchronous mirroring information in the output, enter the following command:

```
lsmp -A –m mp_pok glvm_vg
```

The following output is displayed:

```
VOLUME GROUP:      glvm_vg              Mirror Pool Super Strict: yes

MIRROR POOL:       mp_pok               Mirroring Mode:    ASYNC
ASYNC MIRROR STATE:  active             ASYNC CACHE LV:      bvr_pok_lv
ASYNC CACHE VALID:   yes                ASYNC CACHE EMPTY:   no
ASYNC CACHE HWM:     90                 ASYNC DATA DIVERGED: no
```

lsmpio Command

Purpose

Displays information about the MultiPath I/O (MPIO) storage devices.

Syntax

`lsmpio` [`-l` *device_name*] [`-o`]

`lsmpio` `-S` [`-l` *device_name*] [`-d`]

`lsmpio` `-z` [`-l` *device_name*]

`lsmpio` `-q` [`-l` *device_name*]

`lsmpio` `-a` [`-r`] [`-e`] [`-z`]

`lsmpio` `-h`

Description

The **lsmpio** command displays information about AIX MPIO storage devices. This command can be used only for AIX MPIO storage devices that are controlled by path-control modules (PCMs) that are enabled for the **lsmpio** command support. Some AIX® MPIO storage devices do not support the **lsmpio** command queries. However, all AIX MPIO storage devices support **Path operational status** queries.

The **lsmpio** command displays the following types of information:

- Path operational status
- Path statistics
- Device inquiry data
- Parent adapter information

Path operational status

You can run the **lsmpio** command without any flags or with the **-l** flag to display the path operational status.

```
# lsmpio
name   path_id status  path_status     parent connection
================================================================================
hdisk1234  0   Enabled Opt,Sel,Deg,Rsv fscsi0 500a098186a7d4ca,0008000000000000
hdisk1234  1   Enabled Non             fscsi0 500a098196a7d4ca,0008000000000000
hdisk1234  2   Enabled Opt,Sel         fscsi1 500a098186a7d4ca,0008000000000000
hdisk1234  3   Enabled Non             fscsi1 500a098196a7d4ca,0008000000000000
```

The output is similar to the output that is displayed by running the **lspath** command as shown in the following command:

```
lspath -F "path_id status parent connection"
```

The valid values of the `status` column are *Enabled*, *Disabled*, *Failed*, or *Missing*. The extended `path_status` field might contain one or more three-letter status abbreviations to provide more detailed path status.

Note: Not all extended `path_status` fields are applicable to all MPIO storage devices. Some `path_status` values appear only if the storage area network (SAN) fabric supports notification about SAN congestion.

The possible values for the `path_status` field follow:

Opt
Indicates that the MPIO disk path is an optimized path. This value indicates an MPIO disk path that attaches to a preferred controller in a device that has multiple controllers. The PCM selects one of the preferred MPIO disk paths for I/O operations, whenever possible.

Non
Indicates that the MPIO disk path is a non-optimized path. On a device with preferred MPIO disk paths, this path is not considered as preferred path. The PCM avoids the selection of this path for I/O operations, unless all preferred paths fail.

Act
Indicates that the MPIO disk path is an active path on a device that has active and passive controllers. The PCM selects active MPIO disk paths for I/O operations on such a device.

Pas
Indicates that the MPIO disk path is a passive path on a device that has active and passive controllers. The PCM avoids the selection of passive paths.

Sel
Indicates that the MPIO disk path is being selected for I/O operations, for the time when the `lsmpio` command is to be run.

Rsv
Indicates that the MPIO disk path has experienced an unexpected reservation conflict. This value might indicate a usage or configuration error, with multiple hosts accessing the same disk.

Fai
Indicates that the MPIO disk path experienced a failure. It is possible for a path to have a path status value of *Enabled* and still have an extended path status value of *Fai*. This scenario indicates that operations that are sent on this MPIO disk path are failing, but AIX MPIO has not marked the path as *Failed*. In some cases, AIX MPIO leaves one path to the device in *Enabled* state, even when all MPIO disk paths are experiencing errors.

Deg
Indicates that the MPIO disk path is in a degraded state. This scenario indicates that the MPIO disk path was being used for I/O operations. Those operations experienced errors, thus causing the PCM to temporarily avoid the use of the path. Any additional errors might cause the MPIO disk path to fail.

Clo
Indicates that the MPIO disk path is closed. If all MPIO disk paths to a device are closed, the device is considered to be closed. If only some MPIO disk paths are closed, then those paths might have experienced errors during the last time the device was opened. The AIX MPIO periodically attempts to recover closed paths until the device path is open.

PFa
Indicates that the remote port failed. When the PCM receives an event that indicates that the remote port used by the MPIO disk path is no longer part of the SAN fabric, the PCM marks the path as a failed path. The PCM clears the **PFa** state when the remote port rejoins the SAN fabric.

PCn
Indicates that the SAN fabric reported network traffic congestion to the remote port that is used by the MPIO disk path. The PCM does not use this MPIO disk path if other MPIO disk paths without the network traffic congestion are available for the MPIO device. The PCM automatically clears the **PCn** state if the SAN fabric does not report any recent network traffic congestion events.

PDg
Indicates a degraded remote port. The degraded port might experience many errors even though the MPIO disk path to the remote port remains active. Such degraded port cannot process data. The PCM avoids such disk path. To clear the degraded status of the port, you must first disable the degraded port on the switch that the port belongs to and then enable that port. If the PDg state occurs repeatedly on a path, investigate the ports and fibers used by the path to identify any issues with the SAN hardware.

LCn
Indicates that the link that is associated with the AIX adapter is congested because large amount of data is being sent to or from the AIX adapter. The PCM avoids such MPIO disk paths if other MPIO disk paths without the link congestion are available. The PCM automatically clears the **LCn** state if the SAN fabric does not report any recent link congestion events.

Deferred
This status is possible only on a new path added to an open NVMe disk. The new path cannot be used for I/O due to prevailing conditions set by old or existing paths. The new path is used (`Sel`) when the disk is closed and reopened again.

CtlrRstErr
Indicates that the NVMe controller in the target that is associated with the MPIO disk path might not be reset. A reset is required before enabling the controller.

CtlrRdyErr
Indicates that the NVMe controller in the target that is associated with the MPIO disk path could not be enabled (to make it ready). The controller must become ready for the path to work.

HostDDErr
Indicates that the NVMe host driver in AIX detected a software error or ran out of system resources.

Offline
Indicates that the NVMe controller has been made offline manually preventing it from being used for disk I/O operations.

AdminCmdErr
Indicates that an NVMe Admin command, such as **`Identify`**, has failed. Certain NVMe Admin commands are issued during MPIO disk path initialization. If the command fails, the MPIO disk path is marked as failed.

FabCmdErr
Indicates that an NVMe Fabrics command, such as **`Set Property`**, has failed. Certain NVMe Fabrics commands are issued during MPIO disk path initialization. If the command fails, the MPIO disk path is marked as failed.

ConnectErr
Indicates that the **Connect** command has failed. The **Connect** command is a Fabrics command that is used to form an association between the host and an NVMe controller in the target device. A successful connection is one of the first steps that are required to establish a working MPIO disk path.

AuthErr
Indicates that the target device requires the host to be authenticated but authentication failed.

HbaErr
Indicates that a Host Bus Adapter containing a locally attached NVMe drive has failed or the PCIe slot it resides in has failed. This can happen when the NVMe drive causes repeated errors in accessing system memory.

CrqErr
Indicates that the **Create I/O Queue** command has failed. This NVMe command is used to create queues in a locally attached NVMe drive. The MPIO disk path is unusable if I/O queues are not associated with it.

LinkDwnErr
Indicates that the Fiber Channel Link on the host port is currently down.

FCloginErr
Indicates that the Fiber Channel (FC) Login (*PLOGI*, *FLOGI* or *PRLI*) has failed.

CassErr
Indicates that the **Create Association** service has failed. The **Create Association** service is an NVMe over FC Link Service to establish a transport level association between the host and an NVMe controller. This association is required before the path to an FC attached NVMe storage subsystem can be established.

CiocErr
Indicates that the **Create I/O Connection** service has failed. The **Create I/O Connection** service is an NVMe over FC Link Service to establish a transport level I/O connection between the host and a NVMe controller. This connection is required before the path to an FC attached NVMe storage subsystem can be established.

FCDDErr
Indicates that the Fiber Channel driver in AIX has encountered a software error or ran out of system resources.

EndpntMiss
Indicates that one or more ports of the NVMe storage subsystem is missing on the FC fabric. The physical connectivity of the storage subsystem might have been impaired.

Path statistics

The `-S` flag, along with the optional `-d` flag, causes the `lsmpio` command to display normal or detailed path statistics. The optional `-l` flag restricts the display to contain statistics for just one MPIO storage device. The statistics include how many times the MPIO disk path has been selected for an I/O operation, how many errors have occurred on the MPIO disk path, and how many times the MPIO disk path has failed. The detailed statistics information breaks down the failure counts, into counts of different types of failures.

Device inquiry data

The `-q` flag of the `lsmpio` command causes the AIX MPIO to query the device, by using Small Computer System Interface (SCSI) commands to retrieve and display information about the attached device. Because each queried device is opened and queried by using SCSI commands, this operation might take time to run for many devices.

Parent adapter information

The `-a` flag, along with the optional `-r` flag, causes the `lsmpio` command to display information about the Fibre Channel adapters that are used by the AIX MPIO storage devices. The information includes details about the local adapter identifier, such as the worldwide name of Fibre Channel adapters and the current state of the MPIO disk path, if available. The `-r` flag adds information about remote ports, which are accessed by the Fibre Channel adapter.

You can use the `-z` flag to reset all statistics to zero.

Flags

Item	Description
`-a`	Lists parent Fibre channel adapter information.
`-d`	Displays detailed statistics. This flag is only valid with the `-S` flag.
`-e`	When this flag is used with the **`-a`** flag, error counts for the local adapter ports are displayed. When this flag is used with the **`-a`** and **`-r`** flags, error counts for local adapter ports and remote ports are displayed. These error counts indicate an issue with the physical connectivity between the local adapter and the remote storage port. The error counts indicate recent errors for different time ranges.
`-h`	Displays command usage information.
`-l` *disk_name*	Specifies a device. If this flag is included, the command operates on a single device. If this flag is omitted, the command operates on all AIX MPIO devices. This flag can be used by itself for the summary path status, or with the `-q`, `-S`, or `-z` flags.
`-o`	Indicates that the AIX disk driver attempts to access all multipath I/O (MPIO) that are associated with the specified MPIO disks. The attempt includes accessing the MPIO paths that were marked as failed when the MPIO disk was last closed. This flag provides an up-to-date status about the MPIO path. **Attention:** If you attempt to access failed MPIO disk paths, the response time for the **`lsmpio`** command can be slower. The response time can be delayed by a few seconds or several minutes depending on several factors such as the number of MPIO disks that are impacted and the scope of the **`lsmpio`** command.
`-q`	Queries the device information. This command uses standard SCSI commands to query the device for information. The precise information returned varies, depending on the device type.
`-r`	Displays the remote port information. This flag is used along with the `-a` flag to display information about remote ports that are accessed by an adapter. The information that is returned might depend on the network protocol that is used by the adapter.
`-S`	Displays statistics for one or all devices. This flag displays basic counters for path use and path errors. If the `-d` flag is used along with this flag, it displays more detailed statistics.
`-z`	Resets all statistics. If this flag is used, it causes the PCM to reset all statistical counters back to zero. **Note:** You can use the **`-z`** flag with the **`-a`** flag to reset the adapter error counts.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Examples

Examples of displaying MPIO information:

1. To display the summary information for the hdisk1234 device, enter the following command:

```
lsmpio -l hdisk1234
```

The system displays an output similar to the following output:

```
name   path_id status  path_status     parent connection
================================================================================
hdisk1234  0   Enabled Opt,Sel,Deg,Rsv fscsi0 500a098186a7d4ca,0008000000000000
hdisk1234  1   Enabled Non             fscsi0 500a098196a7d4ca,0008000000000000
hdisk1234  2   Enabled Opt,Sel         fscsi1 500a098186a7d4ca,0008000000000000
hdisk1234  3   Enabled Non             fscsi1 500a098196a7d4ca,0008000000000000
```

2. To display detailed device statistics for the hdisk10 device, enter the following command:

```
lsmpio -Sdl hdisk10
```

The system displays an output similar to the following output:

```
Disk: hdisk10
    Path statistics since Tue May 21 17:38:43 CDT 2013
    Path 0: (fscsi0:500a098186a7d4ca,8000000000000)
        Path Selections:                                    0
        Adapter Errors:                                     0
            Software:                       0
            Hardware:                       0
            Transport Dead:                 0
            Transport Busy:                 0
            Transport Fault:                0
            No Device Response:             0
            Target Port ID Changed:         0
        Command Timeouts:                                   0
        Reservation Conflicts:                              0
        SCSI Queue Full:                                    0
        SCSI Busy:                                          0
        SCSI ACA Active:                                    0
        SCSI Task Aborted:                                  0
        SCSI Aborted Command:                               0
        SCSI Check Condition:                               0
            Medium Error:                   0
            Hardware Error:                 0
            Not Ready:                      0
            Other:                          0
        Last Error:                                       N/A
        Last Error Time:                                  N/A
        Path Failure Count:                                 0
            Due to Adapter Error:           0
            Due to I/O Error:               0
            Due to Health Check:            0
            Due to SCSI Sense:              0
            Due to Qualifier Bit:           0
            Due to Opening Error:           0
        Last Path Failure:                                N/A
        Last Path Failure Time:                           N/A
```

Note: If some SCSI error counts are reported, it does not indicate a problem or indicate that I/O operations have failed. During regular processing, temporary, recoverable errors might be reported frequently, and therefore, the I/O operation might be attempted again.

3. To display MPIO adapter information with the remote port information, enter the following command:

```
lsmpio -ar
```

The system displays an output similar to the following output:

```
Adapter Driver: fscsi0 - AIX PCM
    Adapter WWPN:  10000000c94c7bd6
    Link State:    Up
                        Paths      Paths      Paths      Paths
    Remote Ports       Enabled   Disabled     Failed    Missing         ID
    500a098186a7d4ca         31          0          0          0    0x20a00
    500a098196a7d4ca         31          0          0          0    0x20b00
    500507630a18016b         19          0          0          0    0x31200
    500507630a18416b         19          0          0          0    0x31300
    500507630a18816b         19          0          0          0    0x31400
```

```
    500507630a18c16b         19          0          0          0    0x31500

Adapter Driver: fscsi1 - AIX PCM
    Adapter WWPN:  10000000c94c7bd7
    Link State:    Up
                        Paths      Paths      Paths      Paths
    Remote Ports        Enabled   Disabled     Failed    Missing         ID
    500a098186a7d4ca         31          0          0          0    0x20a00
    500a098196a7d4ca         31          0          0          0    0x20b00
    500507630a18016b         19          0          0          0    0x31200
    500507630a18416b         19          0          0          0    0x31300
    500507630a18816b         19          0          0          0    0x31400
    500507630a18c16b         19          0          0          0    0x31500
    5001738000330150          1          0          0          0    0x10100
    5001738000330162          1          0          0          0    0x10200
```

4. To query an MPIO storage device and display information about it, enter the following command:

```
lsmpio -ql hdisk48
```

The system displays an output similar to the following output:

```
Device:  hdisk48
          Vendor Id:  IBM
         Product Id:  2107900
           Revision:  .160
           Capacity:  10G
      Volume Serial:  600507630AFFC16B0000000000001505  (Page 83 NAA)
```

Note: The output that is displayed is derived from the standard inquiry data and the device identification vital product data (VPD). If the device represents a Peer-to-Peer Remote Copy (PPRC) pair (the **san_rep_device** attribute has a value of *yes*), the display includes the volume serial number for each logical unit number (LUN) in the PPRC pair and the vendor-specific ID that is shared by the two LUNs of the PPRC pair, as shown in the following output:

```
Device:  hdisk33
          Vendor Id:  IBM
         Product Id:  2107900
           Revision:  .160
           Capacity:  10G
      Volume Serial:  600507630AFFC16B0000000000000113  (Page 83 NAA)
      Volume Serial:  600507630AFFC16B000000000000031F  (Page 83 NAA)
      Vendor LUN Id:  3735544C37373130313133005022AD6A
```

5. To display local and remote port error counts, enter the following command:

```
lsmpio -are
```

The output might be similar to the following sample:

```
Adapter Driver: fscsi3 -> AIX PCM
    Adapter WWPN:  21000024ff6aee7d
    Link State:    Up
    Connectivity Errors:
        Last 10 Minutes:      74
        Last 60 Minutes:     222
        Last 24 Hours:     12345

                              Connectivity Errors
                            Last 10   Last 60   Last 24
    Remote Ports            Minutes   Minutes   Hours
    5001738000330171            0         0         0
    5001738000330173            0         0         0
    500a098286a7d4ca            2         9        45
    500a098196a7d4ca           72       213     12300
```

lsnamsv Command

Purpose

Shows name service information stored in the database.

Syntax

lsnamsv { **-C** | **-S"***AttributeList ...***"** } [**-Z**]

Description

The **lsnamsv** high-level command shows customized, TCP/IP-based name service information from the **/etc/resolv.conf** file only. No information from the name server database is shown. The command can extract all customized name service information or selected name service attribute information from the configuration database.

You can use the System Management Interface Tool (SMIT) **smit lsnamerslv** fast path to run this command.

Flags

Item	Description
-C	Extracts all customized name service configuration information.
-S "*AttributeList ...***"**	Specifies a selected set of attributes to be extracted from the system configuration database. Attributes can be the following: ***domain*** Domain name ***nameserver*** Internet address of name server in dotted decimal format
-Z	Specifies that the output be in colon format. This flag is used when the **lsnamsv** command is invoked from the SMIT usability interface.

Examples

1. To list all customized name service configuration information in dotted decimal format, enter the following command:

   ```
   lsnamsv -C
   ```

2. To list selected attributes, enter the following command:

   ```
   lsnamsv -S "domain nameserver"
   ```

   The **-S** flag indicates that the quoted list that follows contains a list of attributes to display.

lsnfsexp Command

Purpose

Displays the characteristics of directories that are exported with the Network File System (NFS).

Syntax

/usr/sbin/lsnfsexp [**-c** | **-l**] [*Directory*] [**-V** *Exported Version*] [**-f** *Exports_file*]

Description

The **lsnfsexp** command displays the characteristics of NFS-exported directories. The *Directory* parameter specifies the directory to be displayed. If no directory is specified, all directories exported with NFS will be displayed.

Flags

Item	Description
-c	Specifies that the output should be in colon format.
-l	(Lowercase L) Specifies that the output should be in list format. This flag is the default.
Directory	Specifies the directory to be displayed. If no directory is specified, all directories exported with NFS will be displayed.
-f *Exports_file*	Specifies the full path name of the export file to use if other than **/etc/exports**.
-V *Exported Version*	Specifies the version of the directory to be displayed. Valid version numbers are 2, 3 and 4.

Examples

1. To list all of the directories currently exported with NFS in the colon format, enter:

   ```
   lsnfsexp -c
   ```

2. To list all of the directories currently exported with NFS in the colon format and use a specified path name other than **/etc/exports** enter:

   ```
   lsnfsexp -c -f /etc/exports.other
   ```

3. To list the entry for the **/common/documents** directory that is exported as version 4, enter the following command:

   ```
   lsnfsexp /common/documents -V 4
   ```

File

Item	Description
/etc/exports	Lists the directories the server can export.

lsnfsmnt Command

Purpose

Displays the characteristics of NFS mountable file systems.

Syntax

/usr/sbin/lsnfsmnt [**-c** | **-l** | **-p**] [*FileSystem*]

Description

The **lsnfsmnt** command displays the current characteristics of NFS mountable file systems. The *FileSystem* parameter specifies the file system to be displayed in the output. If no file system is specified, all of the file systems that are NFS mountable will be displayed.

Flags

Item	Description
-c	Specifies that the output should be in colon format.
-l	(Lowercase L) Specifies that the output should be in list format. This flag is the default.
-p	Specifies that the output should be in pipe format.

Examples

To list all of the NFS mounted file systems in the colon format, enter:

```
lsnfsmnt -c
```

Files

Item	Description
/etc/filesystems	Centralizes file system characteristics.

lsnim Command

Purpose

Displays information about the Network Installation Management (NIM) environment.

Syntax

To Display a List of Supported NIM Classes, Subclasses, or Types

lsnim { **-p**| **-P**} [**-c***Class*| **-S**]

To Display Predefined NIM Information

lsnim { **-p**| **-P**} [**-c***Class*| **-s***Subclass*| **-t***Type*] [**-l** | [**-o**] | **-O**] [**-Z**]

OR

lsnim { **-p**| **-P**}[**-a** *Attribute*] . . . [**-Z**]

To Display Attributes Required for an Operation

lsnim -t*Type* **-q***Operation*

To Display Information about All Customized NIM Objects

lsnim [**-c***Class*| **-s***Subclass*| **-t***Type*] [**-l** | [**-o**] | **-O**] [**-Z**]

OR

lsnim [**-a***Attribute*] . . . [**-Z**]

To Display Information about a Specific NIM Object

lsnim[**-l** | **-O**| **-a** *Attribute* . . .][**-Z**] *ObjectName*

OR

lsnim[**-q***Operation*] *ObjectName*

To Display Information about Resources Available to a Specific NIM Machine

lsnim -L [**-s***Subclass*| **-t***Type*]*ObjectName*

To Display Information about NIM Groups

lsnim -g | **-m** [**-a** *Attribute* | **-c***Class* | **-L** | **-l** | **-s***Subclass* | **-t***Type*] *GroupObjectName*

Description

The **lsnim** command displays information about the NIM environment. This information is divided into two basic categories: predefined and customized.

Predefined information consists of values that are preset by NIM and cannot be modified by the user. Examples of predefined information include:

- The types of objects supported by NIM
- The classes and subclasses into which NIM organizes objects
- The operations that can be performed on NIM objects
- The attributes that can be entered by the user

In general, NIM uses this information to make decisions during operations. Predefined information can be displayed by using the **-p** or **-P** flag. The **-p** flag displays default values while the **-P** flag displays help information.

Customized information consists of values that you enter or modify. This information represents the physical environment in which NIM operates. Related pieces of customized information are grouped together to form *objects*, which are organized in the NIM database by object type and class. Some examples of object types include `diskless`, `paging`, and `standalone`. Two examples of object classes are `machines` and `network`.

For example, a standalone workstation that is part of the NIM environment is represented by a unique object. This object is classified by NIM as a `standalonemachines` object, where `standalone` represents the object type and `machines` represents the object class. Entering the **lsnim** command on the command line without any flags displays information on all customized objects.

You can also use the **lsnim** command to display relationships between customized objects. Choose an object to *anchor* on (specified by the *Objectname* parameter) and then select the desired relationship with the **-c**, **-s**, or **-t** flag. The information displayed then depends upon the type and class of the anchored object. For example, if you select an object of type **spot**, the type of relationships that can be displayed are:

- Machines that use the Shared Product Object Tree (SPOT) resource.
- Networks that can access the SPOT resource.

When not displaying relationships, the **lsnim** command provides flags that can be used to filter the output that it would normally display. The **-a**, **-c**, **-O**, **-s**, or **-t** flag can be used to restrict the amount of information which is displayed.

Flags

Item	Description
-a *Attribute*	Filters displayed information based on the specified attribute name. The possible attributes are: Operation subclass type class
-c *Class*	Specifies a NIM object class. When this flag is used without the *Objectname* parameter, it filters the displayed information so only information about objects in that class is displayed.
-g	Displays long listing of group object with state information for individual members.
-l	Displays detailed information.
-L	Displays information about resources that can be accessed by a client machine.
-m	Applies other flags specified to group members.
-o	Is used by the SMIT interface of a NIM environment.
-O	Lists the operations NIM supports.
-p	Displays predefined information using default values.
-P	Displays help information for predefined data.
-q *Operation*	Lists the attributes required for the specified operation.
-S	Displays a list of NIM subclasses.
-s *Subclass*	Specifies a NIM subclass. When this flag is used without the *ObjectName* parameter, it filters the displayed information so only information about objects in that subclass is displayed.
-t *Type*	Specifies a NIM object type. When this flag is used without the *Objectname* parameter, it filters the displayed information so only information about objects of that type is displayed.
-Z	Displays information in colon-separated format.

Security

Access Control: You must have root authority to run the **lsnim** command.

Examples

1. To display a list of NIM object classes, enter:

   ```
   lsnim -p
   ```

2. To display a list of NIM subclasses, enter:

   ```
   lsnim -p -S
   ```

3. To display the list of NIM object types for the `machines` object class, enter:

   ```
   lsnim -p -c machines
   ```

4. To display help information about NIM object types for the `machines` object class, enter:

```
lsnim -P -c machines
```

5. To display detailed information about the NIM attributes named `lpp_source` and `Rstate`, enter:

```
lsnim -p -a lpp_source -a Rstate
```

6. To display the operations which can be performed on the `paging` object type, enter:

```
lsnim -p -t paging -O
```

7. To display the information required to perform a `bos_inst` operation on an object of the `standalone` object type, enter:

```
lsnim -t standalone -q bos_inst
```

8. To display information about all customized objects of the `diskless` object type, enter:

```
lsnim -t diskless
```

9. To display all customized objects in the `networks` object class, enter:

```
lsnim -c networks
```

10. To display detailed information about a NIM object named `altoid`, enter:

```
lsnim -l altoid
```

11. To display the relationship between an object named `altoid` and all NIM resources, enter:

```
lsnim -c resources altoid
```

12. To display a list of operations that can be applied to `altoid`, enter:

```
lsnim -O altoid
```

13. To display a list of resources available to `altoid`, enter:

```
lsnim -L altoid
```

14. To display the members of the machine group `MacGrp1` with state and group exclusion status, enter:

```
lsnim -g MacGrp1
```

15. To display basic information about the members of the resource group `ResGrp1`, enter:

```
lsnim -m ResGrp1
```

16. To display a long listing of members of the machine group `MacGrp1`, with any hidden NIM internal information, enter:

```
lsnim -m -Fl MacGrp1
```

17. To display all members of machine group `MacGrp1` which has a spot allocated, enter:

```
lsnim  -ma spot MacGrp1
```

Files

Item	Description
/etc/niminfo	Contains variables used by NIM.

lsnlspath Command

Purpose

Shows the value of the secure NLSPATH system configuration variable.

Syntax

lsnlspath

Description

The **lsnlspath** command outputs the current value of the secure NLSPATH variable.

lsparent Command

Purpose

Displays the possible parent devices that accept a specified connection type or device.

Syntax

lsparent { **-C** | **-P** } { **-k** *ChildConnectionKey* | **-l** *ChildName* } [**-f** *File*] [**-F***Format*] [**-h**] [**-H**]

Description

The **lsparent** command lists devices from the Device Configuration database that can accept a given device as a child device, specified by the **-l** *ChildName* flag, or a given type of child device connection, specified by the **-k** *ChildConnectionKey* flag.

You can display the default output one of the following ways.

- Use the **-C** flag to display the default output information for a device from the Customized Devices object class, which is name, state, location, and description.
- Use the **-P** flag to display the default output information for a device from the Predefined Devices object class, which is class, type, subclass, and description.

To override these two default outputs, you can use the **-F** *Format* flag to display the output as designated by a user-formatted string. The *Format* parameter is a quoted list of column names separated and possibly terminated by nonalphanumeric characters.

You can supply the flags either on the command line or from the specified *File* parameter using the **-f** flag.

Flags

Item	Description
-C	Lists information about a device that is in the Customized Devices object class. The information displayed can be from both the Customized and Predefined Devices object classes. This flag cannot be used with the **-P** flag.
-f *File*	Reads the necessary flags from the *File* variable.

Item	Description
-F *Format*	Displays the output in a user-specified format, where the *Format* variable is a quoted list of column names from the Predefined Devices object class or the Customized Devices object class separated and possibly terminated by non-alphanumeric characters. If white space is used as the separator, the **lsparent** command displays the output in aligned columns. In addition to the column names in the two object classes, the special name *description* can be used to display a text description of the device.
-H	Displays headers above the column output.
-h	Displays the command usage message.
-k *ChildConnectionKey*	Specifies the connection key that identifies the device subclass name of the child device. This flag cannot be used with the **-l** flag.
-l *ChildName*	Specifies the logical name of a possible child device. This flag cannot be used with the **-k** flag.
-P	Lists information about a device that is in the Predefined Devices object class. The information displayed can be from both the Customized and Predefined Devices object classes. This flag cannot be used with the **-C** flag.

Examples

1. To list possible parent devices in the Customized Devices object class that accept an RS-232 device, type the following:

   ```
   lsparent -C -k rs232
   ```

   The system displays a message similar to the following:

   ```
   sa0 Available 01-S1 Standard I/O Serial Port
   sa1 Available 01-S2 Standard I/O Serial Port
   sa2 Available 10-68 IBM 8-Port EIA-232/RS-422A (PCI) Adapter
   sa3 Available 10-70 IBM 8-Port EIA-232/RS-422A (PCI) Adapter
   sa4 Available 10-78 IBM 8-Port EIA-232/RS-422A (PCI) Adapter
   sa5 Available 20-58 IBM 8-Port EIA-232/RS-422A (PCI) Adapter
   ```

2. To list possible types of parent devices in the Predefined Devices object class that accept an RS-232 device, type the following:

   ```
   lsparent -P -k rs232
   ```

   The system displays a message similar to the following:

   ```
   adapter      pnp501   isa_sio  Standard I/O Serial Port
   adapter      4f111100 pci      IBM 8-Port EIA-232/RS-422A (PCI) Adapter
   concentrator 16c232   sync_pci 16-Port RAN EIA-232 for 128-Port Adapter
   concentrator 16e232   sync_pci 16-Port Enhanced RAN EIA-232 for 128-Port Adapter
   ```

3. To list possible parent devices in the Customized Devices object class that accept the `rmt0` tape device as a child device, type the following:

   ```
   lsparent -C -l rmt0
   ```

   The system displays a message similar to the following:

   ```
   scsi2 Available 20-60 Wide/Ultra-2 SCSI I/O Controller
   scsi3 Available 20-61 Wide/Ultra-2 SCSI I/O Controller
   scsi1 Available 10-88 Wide/Ultra-2 SCSI I/O Controller
   scsi0 Available 10-60 Wide/Fast-20 SCSI I/O Controller
   ```

4. To list possible types of parent devices in the Predefined Devices object class that accept the `rmt0` tape device as a child device, type the following:

```
lsparent -P -l rmt0
```

The system displays a message similar to the following:

```
adapter sym896 pci Wide/Ultra-2 SCSI I/O Controller
adapter sym895 pci Wide/Ultra-2 SCSI I/O Controller
adapter sym875 pci Wide/Fast-20 SCSI I/O Controller
```

Files

Item	Description
/usr/sbin/lsparent	Contains the **lsparent** command.

lspath Command

Purpose

Displays information about paths to an MultiPath I/O (MPIO) capable device.

Syntax

lspath [**-F** *Format* | **–t**] [**-H**] [**-l** *Name*] [**-p** *Parent*] [**-s** *Status*] [**-w** *Connection*] [**-i** *PathID*]

lspath -A -l *Name* **-p** *Parent* [**-w** *Connection*] [**-i** *PathID*] {**-D** [**-O**] | **-E** [**-O**] | **-F** *Format* [**-Z** *character*] } [**-a** *Attribute*] ...[**-f** *File*] [**-H**]

lspath -A -l *Name* **-p** *Parent* [**-w** *Connection*] [**-i** *PathID*]**-R -a** *Attribute* [**-f** *File*] [**-H**]

lspath -h

Description

The **lspath** command displays one of two types of information about paths to an MPIO capable device. It either displays the operational status for one or more paths to a single device, or it displays one or more attributes for a single path to a single MPIO capable device. The first syntax shown above displays the operational status for one or more paths to a particular MPIO capable device. In this instance, the **lspath** command is similar to the **lsdev** command. The second syntax (keyed by the presence of the **-A** flag) displays one or more attributes for a single path to a particular MPIO capable device. In this instance, the lspath command is similar to the **lsattr** command. In fact, all of the flags for the **lsattr** command are supported on the **lspath** command when displaying path attributes.

Displaying Path Status with the lspath Command

When displaying path status, the set of paths to display is obtained by searching the device configuration database for paths that match the following criteria:

- The target device name matches the device specified with the **-l** flag. If the **-l** flag is not present, then the target device is not used in the criteria.
- The parent device name matches the device specified with the **-p** flag. If the **-p** flag is not present, then parent is not used in the criteria.
- The connection matches the connection specified with the **-w** flag. If the **-w** flag is not present, then connection is not used in the criteria.
- The path status matches status specified with the **-s** flag. If the **-s** flag is not present, the path status is not used in the criteria.

If none of the **-l**, **-p**, **-w**, **-s** flags are specified, then all paths known to the system are displayed.

By default, this command will display the information in columnar form. When no flags are specified that qualify the paths to display, the format of the output is:

```
status device  parent
```

The default display format can be overridden by using the **-F** *Format* flag. The **-F** *Format* flag displays the output in a user-specified format where the *Format* parameter is a quoted list of field names separated by, and possibly ended by, non-alphanumeric characters or white space. The field names are the fields defined in the **CuPath** class or one of the column heading defined above.

Note: The column names above are not translated into other languages (either when output as column headings or when input as part of the *Format* of the **-F** flag).

Possible values that can appear for the status column are:

enabled
Indicates that the path is configured and operational. It will be considered when paths are selected for IO.

Note: The AIX MPIO does not fail the last path. Even though all paths to the storage are lost, the last path displays the status as **enabled**. In such a case, an IO error indicates the actual loss of the last path to the disk.

disabled
Indicates that the path is configured, but not currently operational. It has been manually disabled and will not be considered when paths are selected for IO.

failed
Indicates that the path is configured, but it has had IO failures that have rendered it unusable. It will not be considered when paths are selected for IO.

defined
Indicates that the path has not been configured into the device driver.

missing
Indicates that the path was defined in a previous boot, but it was not detected in the most recent boot of the system.

detected
Indicates that the path was detected in the most recent boot of the system, but for some reason it was not configured. A path should only have this status during boot and so this status should never appear as a result of the **lspath** command.

Displaying Path Attributes with the lspath Command

When displaying attributes for a path, the path must be fully qualified. Multiple attributes for a path can be displayed, but attributes belonging to multiple paths cannot be displayed in a single invocation of the **lspath** command. Therefore, in addition to the **-A** flag, the **-l**, **-p**, or**-w** flags are required to uniquely identify a single path. For example:

- if only one path exists to a device, the **-l** flag is required
- if only one path between a device and a specific parent, the **-l** and **-p** flags are required
- if there are multiple paths between a device and a specific parent, the **-l**, **-p**, and **-w** flags are required

Furthermore, the **-s** flag is not allowed.

The same rules used by the **lsattr** command for displaying device attributes applies to the **lspath** command for displaying path attributes.

By default, this command will display the information in columnar form. The format of the output is the same as the **lsattr** command:

```
attribute   value    description         user_settable
```

All fields are shown by default. The default display format can be overridden by using the **-F** *Format* flag. The **-F** *Format* flag displays the output in a user-specified format where the *Format* parameter is a

quoted list of column names separated by, and possibly ended by, non-alphanumeric characters or white space. The column names allowed are the field names from the **CuPathAt**, **PdPathAt**, and **PdAtXtd** object classes plus the columns listed above.

Note: The column names above are not translated into other languages (either when output as column headings or when input as part of the *Format* of the **-F** flag).

Flags

Item	Description
-a *Attribute*	Identifies the specific attribute to list. The '*Attribute*' is the name of a path specific attribute. When this flag is provided, only the identified attribute is displayed. Multiple instances of this flag may be used to list multiple attributes. If this flag is not specified at all, all attributes associated with the identified path will be listed.
-A	Indicates that attributes for a specific path are to be displayed. When the **-A** flag is present, the **-s** *Status* flag is not allowed. However, the **-l** *Name*, **-p** *Parent*, and **-w** *Connection* flags must be present to fully qualify the path.
-D	Displays the attribute names, default values, descriptions, and user-settable flag values for a specific path when not used with the **-O** flag. The **-D** flag displays only the attribute name and default value in colon format when used with the **-O** flag. This flag is only valid when displaying path attributes and it cannot be used with the **-E**, **-F**, or **-R** flag.
-E	Displays the attribute names, current values, descriptions, and user-settable flag values for a specific path when not used with the **-O** flag. The **-E** flag displays only the attribute name and current value in colon format when used with the **-O** flag. This flag is only valid when displaying path attributes and it cannot be used with the **-D**, **-F**, or **-R** flag.
-f *File*	Reads the needed flags from the *File* parameter.
-F *Format*	Displays the output in a user-specified format, where the *Format* parameter is a quoted list of column names separated by non-alphanumeric characters or white space. Using white space as the separator, the lspath command displays the output in aligned columns. Valid column names depends upon the type of information requested. For path display, column names from the **CuPath** object class can be specified. For path attribute display (the **-A** flag is specified), column names from the **PdPathAt** and **CuPathAt** object classes can be specified. In addition to the column names, there are two special purpose names that can be used. The name description can be used to obtain a display of attribute descriptions and user-settable can be used to obtain an indication as to whether or not an attribute can be changed. This flag cannot be used with the **-E**, **-D**, **-O** or **-R** flag.
-h	Displays the command usage message.
-H	Displays headers above the column output. To use the **-H** flag with the **-O** flag is meaningless, the **-O** flag prevails. To use the **-H** flag with the **-R** flag is meaningless; the **-R** flag prevails.
-i *PathID*	Indicates the path ID associated with the path to be displayed.
-l *Name*	Specifies the logical device name of the target device whose path information is to be displayed. This flag is optional for displaying path status, but is required for displaying path attributes.
-O	Displays all attribute names separated by colons and, on the second line, displays all the corresponding attribute values separated by colons. The attribute values are current values when the **-E** flag is also specified and default values when the **-D** flag is specified. This flag is only valid when displaying path attributes and it cannot be used with the **-F** and **-R** flags.

Item	Description
-p *Parent*	Indicates the logical device name of the parent device, whose paths are to be displayed. This flag is optional for displaying path status, but it is required for displaying path attributes.
-R	Displays the legal values for an attribute name. The **-R** flag cannot be used with the **-D**, **-E**, **-F** and **-O** flags. The **-R** flag displays the list attribute values in a vertical column as follows: `Value1` `Value2` `.` `.` `ValueN` The **-R** flag displays the range attribute values as x...n(+i) where x is the start of the range, n is the end of the range, and i is the increment.
-s *Status*	The **-s** *Status* flag indicates the status to use in qualifying the paths to be displayed. When displaying path information, the allowable values for this flag are: **enabled** Display paths that are **enabled** for MPIO path selection. **disabled** Display paths that are **disabled** from MPIO path selection. **failed** Display paths that are **failed** due to IO errors. **available** Display paths whose **path_status** is **PATH_AVAILABLE** (that is, paths that are configured in the system, includes **enabled**, **disabled**, and **failed** paths). **defined** Display paths whose **path_status** is **PATH_DEFINED**. **missing** Display paths whose **path_status** is **PATH_MISSING**. You can use alternative forms of the values. Strings beginning with **0**, **d**, or **D** are treated in the same way as **defined**. The only exception is that strings starting with **di**, **Di**, **dI**, or **DI** are treated in the same way as **disabled**. Strings beginning with **1**, **a**, or **A** are treated in the same way as **available**. Strings beginning with **3**, **m**, or **M** are treated in the same way as **missing**. Strings beginning with **e** or **E** are treated in the same way as **enabled**. Strings beginning with **f** or **F** are treated in the same way as **failed**.
-t	Displays the path ID in addition to the current default output. The **-t** flag cannot be used with the **-F** or the **-A** flags.
-w *Connection*	Indicates the connection information to use in qualifying the paths to be displayed. This flag is optional for displaying path status, but is required for displaying path attributes.
-Z *Character*	The **-Z** *Character* flag is meant to be used for programs that need to deal with ODM fields that may have embedded new line characters. The **-Z** *Character* flag is used to change the record separator character for each record (line) of output generated. The new record separator is the *'Character'* argument to this flag. The **-Z** *Character* flag is only relevant when the **-A** and the **-F** *Format* flags are specified. The **-Z** *Character* flag cannot be used with the **-D**, **-E**, **-O**, or the **-R** flags.

Security

Privilege Control: All users can execute this command.

Auditing Events: None.

Examples

Examples of displaying path status:

1. To display the status of all paths to hdisk1 with column headers, enter the following command:

   ```
   lspath -H -l hdisk1
   ```

   The system will display a message similar to the following:

   ```
   status    device   parent
   enabled   hdisk1   scsi0
   disabled  hdisk1   scsi1
   missing   hdisk1   scsi2
   ```

2. To display, without column headers, the set of paths whose operational status is disabled, enter the following command:

   ```
   lspath -s disabled
   ```

   The system will display a message similar to the following:

   ```
   disabled  hdisk1   scsi1
   disabled  hdisk2   scsi1
   disabled  hdisk23  scsi8
   disabled  hdisk25  scsi8
   ```

3. To display the set of paths whose operational status is `failed`, enter the following command:

   ```
   lspath -s failed
   ```

   The system will display a message similar to the following:

   ```
   failed  hdisk1   scsi1
   failed  hdisk2   scsi1
   failed  hdisk23  scsi8
   failed  hdisk25  scsi8
   ```

4. To display in a user-specified format, without column headers, the set of paths to hdisk1 whose path status is available enter the following command:

   ```
   lspath -l hdisk1 -s available -F"connection:parent:path_status:status"
   ```

   The system will display a message similar to the following:

   ```
   5,0:scsi0:available:enabled
   6,0:scsi1:available:disabled
   ```

   Note that this output shows both the path status and the operational status of the device. The path status simply indicates whether the path is configured or not. The operational status indicates how the path is being used with respect to path selection processing in the device driver. Only paths with a path status of `available` also have an operational status. If a path is not currently configured into the device driver, it does not have an operational status.

Examples of displaying path attributes:

1. If the target device is a SCSI disk, to display all attributes for the path to parent scsi0 at connection 5,0, enter the following command:

   ```
   lspath -AHE -l hdisk10 -p scsi0 -w "5,0"
   ```

   The system will display a message similar to the following:

   ```
   attribute  value  description                      user_settable
   weight     1      Order of path failover selection  true
   ```

lspriv Command

Purpose

Displays the privileges available on the system.

Syntax

lspriv [**-v**]

Description

The **lspriv** command displays privileges available to the system. If run within a workload partition (WPAR), the **lspriv** command displays only the privileges available to the partition. If the **-v** flag is specified, the **lspriv** command also displays privilege descriptions.

Flags

Item	Description
-v	Specifies to display the privilege description for each privilege.

Security

Any user can run the **lspriv** command on the system.

Examples

1. To display all privileges available on the system, use the following command:

   ```
   lspriv
   ```

   The system displays an output similar to the following example:

   ```
   PV_ROOT
   PV_AU_
   PV_AU_ADD
   PV_AU_ADMIN
   PV_AU_READ
   ...
   ```

2. To display all privileges available on the system and their textual description, use the following command:

   ```
   lspriv -v
   ```

   The system displays an output similar to the following example:

   ```
   PV_ROOT Allows a process to pass any non-SU privilege check.
   PV_AU_ Equivalent to all Auditing privileges (PV_AU_*) combined.
   PV_AU_ADD Allows a process to record/add an audit record.
   PV_AU_ADMIN Allows a process to configure and query the audit system.
   PV_AU_READ Allows a process to read a file marked as an audit file.
   ...
   ```

lsprtsv Command

Purpose

Shows print service information stored in the database.

Syntax

lsprtsv { **-c** | **-p** } [**-h**] [**-q***QEntry ...*] [**-Z**]

Description

The **lsprtsv** high-level command shows predefined and customized TCP/IP-based print service information. Use the **lsprtsv** command to extract customized or predefined print service information.

The **lsprtsv** command can show the following information:

- A list of host names that have access rights to the print server
- Logical print queue information only

Flags

Item	Description
-c	Extracts customized configuration information.
-h	Shows a list of host names that can use the print server.
-p	Extracts predefined configuration information.
-q*QEntry...*	Shows the logical print queues specified and their attributes available on a host. The *QEntry* variable specifies the names of the queues to display.
-Z	Specifies that the output be produced in colon format. This flag is used if the **lsprtsv** command is invoked from the SMIT usability interface.

Examples

1. To show all host names who have access rights to a print server, enter:

   ```
   $ lsprtsv  -c  -h
   ```

2. To show which logical printers are available on a given client machine, enter:

   ```
   lsprtsv  -c  -q sahara
   ```

lsps Command

Purpose

Displays the characteristics of a paging space.

Syntax

lsps { **-s** | [**-c** | **-l**] { **-a** | **-t** { *lv* | *nfs* | *ps_helper*} | *PagingSpace* } }

Description

The **lsps** command displays the characteristics of a paging space. The **lsps** command displays characteristics such as the paging-space name, physical-volume name, volume-group name, size, percentage of the paging space used, whether the space is active or inactive, and whether the paging space is set to automatic. The *PagingSpace* parameter specifies the paging space whose characteristics are to be shown.

For NFS paging spaces, the physical-volume name and volume-group name will be replaced by the host name of the NFS server and the path name of the file that is used for paging.

If the **-t** flag is specified, the argument will be assumed to be a third-party helper executable. If the helper executable is present in the /sbin/helpers/pagespace path then it will be spawned passing all the arguments and with the **-l** flag to specify the **lsps** command. The helper executable must take care of displaying the characteristics of the page space. If the helper program doesn't exist in /sbin/helpers/pagespace directory, the **lsps** command will display the usage error. The helper executable must exit with a 0 if successful and a non-zero if it fails.

You can use the System Management Interface Tool (SMIT) **smit lsps** fast path to run this command.

Flags

Item	Description
-a	Specifies that the characteristics of all paging spaces are to be given. The size is given in megabytes.
-c	Specifies that the output should be in colon format. The colon format gives the paging space size in logical partitions.
-l	Specifies that the output should be in list format.
-s	Specifies that the summary characteristics of all paging spaces are to be given. This information consists of the total paging space in megabytes and the percentage of paging space currently assigned (used). If the **-s** flag is specified, all other flags are ignored. **Note:** There is a paging space limit of 64 GB per device. **Note:** Setting the environment variable **PSALLOC=early** causes the use of early paging space algorithm. In this case, the value the **-s** flag specifies is different from the value returned for a single paging space or when using the **-a** flag for all the paging spaces. The value the **-s** flag displays is the percentage of paging space allocated (reserved), whether the paging space has been assigned (used) or not. Therefore, the percentage reported by the **-s** flag is usually larger than that reported by the **-a** flag when **PSALLOC** is set to early.
-t	Specifies the characteristics of the paging space. One of the following variables is required: ***lv*** Specifies that the characteristics of only logical volume paging spaces are to be given. ***nfs*** Specifies that the characteristics of only NFS paging spaces are to be given. The heading of the output will be changed to display the host name of the NFS server and the path name of the file that resides on the server that is being used for NFS paging. ***ps_helper*** Name of the helper program for a third party device.

Examples

1. To list the characteristics of all paging spaces, enter:

   ```
   lsps  -a
   ```

   This displays the characteristics for all paging spaces and provides a listing similar to the following listing:

   ```
   Page Space   PhysicalVolume  Volume Group    Size   %Used  Active  Auto  Type  Chksum
   hd6           hdisk0          rootvg         512MB    1     yes     yes   lv     8
   ```

2. To display the characteristics of paging space myps using the helper program foo enter the following commad:

   ```
   lsps -t foo myps
   ```

This displays the characteristics for all paging spaces and provides a listing similar to the following listing:

```
Page Space      Physical Volume   Volume Group    Size %Used Active  Auto  Type
myps             mydisk            myvg            512MB     1    yes   yes    lv
```

Files

Item	Description
/etc/swapspaces	Specifies the paging space devices and their attributes.

lspv Command

Purpose

Displays information about a physical volume within a volume group.

Syntax

lspv

OR

lspv [**-L**] [**-P**] [**-l** | **-p** | **-M**] [**-n** *descriptorphysicalvolume*] [**-v** *volumegroupid*] *physicalvolume*

Description

The **lspv** command displays the information about the physical volume if the specific physical volume name is specified. If you do not add flags to the **lspv** command, by default all the available physical volumes are printed along with the following information:

- Physical disk name.
- Physical volume identifiers (PVIDs).
- The volume group, if any, that the physical volume belongs to or the label, if any, locked with the **lkdev** command.
- The state of the volume group.

  Active
  When the volume group is varied on.

  Concurrent
  When the volume group is varied on in the concurrent mode.

  Locked
  When the physical volume is locked with the **lkdev** command.

Note: If the **lspv** command cannot find the information for a field in the Device Configuration Database, it will insert a question mark (?) in the value field. As an example, if there is no information for the **PP RANGE** field, the following value might be displayed:

```
PP RANGE: ?
```

Note: The **lspv** command, without any flags, can display the General Parallel File System (GPFS) volume groups that are located on the disks. However, the **lspv** command must initially be run with a root authority so that the command has permissions to query the GPFS nodes for information. After the GPFS volume group names are cached locally, non-root users running the **lspv** command can see the GPFS volume group names.

The **lspv** command attempts to obtain as much information as possible from the description area when it is given a logical volume identifier.

When the *physicalvolume* parameter is used, the following characteristics of the specified physical volume are displayed:

Item	Description
`Physical volume`	The name of the physical volume.
`Volume group`	The name of volume group. Volume group names must be unique systemwide names and can be from 1 to 15 characters long.
`PV Identifier`	The physical volume identifier for this physical disk.
`VG Identifier`	The volume group identifier of which this physical disk is a member.
`PVstate`	The state of the physical volume. If the volume group that contains the physical volume is varied on with the **varyonvg** command, the state is `active`, `missing`, or `removed`. If the physical volume is varied off with the **varyoffvg** command, the state is `varied off`.
`Allocatable`	The allocation permission for this physical volume.
`Logical volumes`	The number of logical volumes using the physical volume.
`Stale PPs`	The number of physical partitions on the physical volume that are not current.
`VG descriptors`	The number of volume group descriptors on the physical volume.
`PP size`	The size of physical partitions on the volume.
`Total PPs`	The total number of physical partitions on the physical volume.
`Free PPs`	The number of free physical partitions on the physical volume.
`Used PPs`	The number of used physical partitions on the physical volume.
`Max Request`	The max transfer size of the physical volume.
`Free distribution`	The number of free partitions available in each intra-physical volume section.
`Used distribution`	The number of used partitions in each intra-physical volume section.
`Mirror Pool`	The mirror pool that the physical volume has been assigned to.

You can use the System Management Interface Tool (SMIT) **smit lspv** fast path to run this command.

Flags

Item	Description
-L	Specifies no waiting to obtain a lock on the Volume group. **Note:** If the volume group is being changed, using the **-L** flag gives unreliable date.

Item	Description
-l	Lists the following fields for each logical volume on the physical volume: **LVname** Name of the logical volume to which the physical partitions are allocated. **LPs** The number of logical partitions within the logical volume that are contained on this physical volume. **PPs** The number of physical partitions within the logical volume that are contained on this physical volume. **Distribution** The number of physical partitions, belonging to the logical volume, that are allocated within each of the following sections of the physical volume: outer edge, outer middle, center, inner middle and inner edge of the physical volume. **Mount Point** File system mount point for the logical volume, if applicable.
-M	Lists the following fields for each logical volume on the physical volume: `PVname:PPnum [LVname: LPnum [:Copynum] [PPstate]]` Where: **PVname** Name of the physical volume as specified by the system. **PPnum** Physical partition number. **LVname** Name of the logical volume to which the physical partitions are allocated. Logical volume names must be system-wide unique names, and can range from 1 to 64 characters. **LPnum** Logical partition number. Logical partition numbers can range from 1 to 64,000. **Copynum** Mirror number. **PPstate** Only the physical partitions on the physical volume that are not current are shown as stale.
-n *descriptorphysicalvolume*	Accesses information from the variable descriptor area specified by the *descriptorphysicalvolume* variable. The information may not be current, since the information accessed with the -n flag has not been validated for the logical volumes. If you do not use the -n flag, the descriptor area from the physical volume that holds the validated information is accessed, and therefore the information displayed is current. The volume group need not be active when you use this flag.

Item	Description
-p	Lists the following fields for each physical partition on the physical volume: **Range** A range of consecutive physical partitions contained on a single region of the physical volume. **State** The current state of the physical partitions: `free`, `used`, `stale`, or `vgda`. **Note:** If a *volume group* is converted to a big vg format, it may be necessary to use some data partitions for *volume group* descriptor area. These partitions will be marked `vgda`. **Region** The intra-physical volume region in which the partitions are located. **LVname** The name of the logical volume to which the physical partitions are allocated. **Type** The type of the logical volume to which the partitions are allocated. **Mount point** File system mount point for the logical volume, if applicable.
-P	Lists the mirror pool that each physical volume belongs to.
-u	Lists all the physical volumes in the system along with the following information: • Physical disk name. • Physical volume identifiers (PVIDs). • The volume group (if any), or label (if any), that the physical volume belongs to and that is locked with the **lkdev** command. • The state of the volume group. **Active** When the volume group is varied on. **Concurrent** When the volume group is varied on in the concurrent mode. **Locked** When the physical volume is locked with the **lkdev** command. • Unique device identifier (UDID). • Universally Unique Identifier (UUID).
-v *volumegroupid*	Accesses information based on the *volumegroupid* variable. This flag is needed only when the lspv command does not function due to incorrect information in the Device Configuration Database. The *volumegroupid* variable is the hexadecimal representation of the volume group identifier, which is generated by the **mkvg** command.

Examples

1. To display the status and characteristics of physical volume `hdisk3`, enter the following command:

```
lspv hdisk3
```

2. To display the status and characteristics of physical volume `hdisk5` by physical partition number, enter the following command:

```
lspv  -p hdisk5
```

Files

Item	Description
/usr/sbin	Contains the **lspv** command.

lspprc Command

Purpose

Displays information about PPRC (Peer-to-Peer Remote Copy) disks.

Syntax

lspprc -A *[-o]*

lspprc -c *pprc_disk*

lspprc *[-h]*

lspprc -p *pprc_disk*

lspprc -v *pprc_disk*

Description

The **lspprc** command displays information that is related to PPRC disk like vital product data (VPD) information of the individual LUNs that are part of the PPRC disk, path group information of a PPRC disk, replication path information of a PPRC disk, and a list of all PPRC disks available on the system.

Flags

Item	Description
-A *[-o]*	Displays the information of all PPRC disks in the system, such as PPRC state and path groups IDs. The optional **-o** flag briefly opens all potential PPRC disks before displaying the status, ensuring that the status is current. Use of the **-o** flag might increase the length of time that is required to run the command, depending on the number of disks that are attached to the system. Example: `hdisk#  PPRC  Primary  Secondary  Primary Storage  Secondary Storage` `  state  path group ID  path group ID  WWNN  WWNN` `hdisk4  Active  0(s)  1  500507630affc16b 500507630affc16b` `hdisk5  Active  0(s)  1  500507630affc16b 500507630affc16b` `hdisk6  Active  0(s)  1  500507630affc16b 500507630affc16b` **Note:** For explanation of the path group IDs and the selected path group that is identified as *s*, see the description of the **-p** flag.

Item	Description
-c	Displays information about the replication path connection, which is related to a PPRC disk. The output displays information about the replication paths between the two storage subsystems that contain the LUNs in the PPRC pair. The output displays the worldwide node names, Subsystem IDs (SSID), Logical Subsystems (LSS), and ports for the endpoints of the paths and the current path state. The contents of the output depends on the current state of the PPRC disk. • If the disk is not part of a PPRC pair, the output shows all paths that originate from the LSS that the disk resides on, to any other LSS. • If the disk is part of a PPRC pair, but has only one path group, the output displays paths that originate from the LSS on which the disk resides and lead to the LSS on which the partner disk resides. • If the disk is part of a PPRC pair and has two path groups, the output displays all paths between the two LSSs on which the members of the PPRC pair reside. Example output of the replication paths: `lspprc -c hdisk33` `Displays all paths between LSS 01 and LSS 03` `Source Target` `WWNN SSID LSS Port WWNN SSID LSS Port State` `==========` `500507630AFFC16B FF03 03 0301 500507630AFFC16B FF01 01 0302 Up` `500507630AFFC16B FF03 03 0302 500507630AFFC16B FF01 01 0303 Up` `500507630AFFC16B FF01 01 0300 500507630AFFC16B FF03 03 0302 Up`
-p	Displays path group information that is part of the specified PPRC disk. Example output of path group information: `lspprc -p hdisk55` `Path WWNN LSS VOL Path` `group id group status` `==========` `0(s) 500507630affc16b 0xf 0x1c` `PRIMARY` `1 5005076303ffd2ea 0xc1 0x0 SECONDARY` `path path id path parent connection` `group id status` `==========` `0 0 Available fscsi0 500507630a08016b,400f401c00000000` `0 1 Available fscsi0 500507630a08416b,400f401c00000000` `1 2 Available fscsi1 50050763030812ea,40c1400000000000` `1 3 Available fscsi1 50050763030852ea,40c1400000000000` When you use PPRC, the paths are grouped based on which LUN is accessed by the path, in the PPRC pair. The path group ID indicates how the paths are grouped, with all paths that have the same path group ID accessing the same LUN in the PPRC pair. A path group ID of -1 indicates that there are no paths that are configured from this initiator to the indicated LUN in the PPRC pair. At any time, only one of the two path groups is selected for I/O operations to the hdisk. The selected path group is identified in the output by "(s)".

Item	Description
-v	Displays the VPD information of individual LUNs that are part of given PPRC disk. Example output of VPD information:

```
# lspprc -v hdisk0

Hyperswap lun unique identifier.............35203735544c3737313
037303000502a14ae07210790003IBMfcp

hdisk0  Primary         MPIO IBM 2107 FC Disk

        Manufacturer..................IBM
        Machine Type and Model........2107900
        ROS Level and ID..............2E313630
        Serial Number.................75TL7710
        Device Specific.(Z7)..........0700
        Device Specific.(Z0)..........000005329F101002
        Device Specific.(Z1)..........700
        Device Specific.(Z2)..........075
        Unique Device Identifier......200B75TL771070007210790003IBMfcp
        Logical Subsystem ID..........0x07
        Volume Identifier.............0x00
        Subsystem Identifier(SS ID)...0xFF07
        Control Unit Sequence Number..00000TL771
        Storage Subsystem WWNN........500507630affc16b
        Logical Unit Number ID........4007400000000000

hdisk0  Secondary       MPIO IBM 2107 FC Disk

        Manufacturer..................IBM
        Machine Type and Model........2107900
        ROS Level and ID..............2E313630
        Serial Number.................75TL7710
        Device Specific.(Z7)..........0900
        Device Specific.(Z0)..........000005329F101002
        Device Specific.(Z1)..........900
        Device Specific.(Z2)..........075
        Unique Device Identifier......200B75TL771090007210790003IBMfcp
        Logical Subsystem ID..........0x09
        Volume Identifier.............0x00
        Subsystem Identifier(SS ID)...0xFF09
        Control Unit Sequence Number..00000TL771
        Storage Subsystem WWNN........500507630affc16b
        Logical Unit Number ID........4009400000000000
```

Files

Item	Description
/usr/bin/lspprc	Contains the **lspprc** command.

lsque Command

Purpose

Displays the queue stanza name.

Syntax

lsque [**-c**] **-q***Name*

Description

The **lsque** command uses the **printf** subroutine to display the name of the queue stanza and associated attributes from the **/etc/qconfig** file.

Flags

Item	Description
-c	Causes colon output format for use by SMIT.
-q *Name*	Specifies the *Name* of the queue stanza that is sent to standard output.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the name of queue stanza `lp0`, enter:

   ```
   lsque  -qlp0
   ```

   A list similar to the following is displayed:

   ```
   lp0:
      device = lpd0
      host = neptune
      rq = nlp0
   ```

2. To display the name of queue stanza `lp0` in colon format, enter:

   ```
   lsque  -c  -q lp0
   ```

   A list similar to the following is displayed:

   ```
   device:discipline:up:acctfile:host:s_statfilter:l_statfilter:rq
   lpd0:fcfs:true:false:neptune:::nlp0
   ```

Files

Item	Description
/usr/bin/lsque	Contains the **lsque** command.
/etc/qconfig	Contains the configuration file.

lsquedev Command

Purpose

Displays the device stanza name.

Syntax

lsquedev [**-c**] **-q***Name* **-d** *Name*

Description

The **lsquedev** command displays the name of the queue stanza and associated attributes from the **/etc/qconfig** file.

Flags

Item	Description
-c	Specifies colon output format for use by SMIT.
-d *Name*	Specifies the *Name* variable of the device stanza that is displayed.
-q *Name*	Specifies the *Name* variable of the queue containing the device stanza that is displayed.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display device stanza `dlp0` on the `lp0` queue, type:

   ```
   lsquedev  -q lp0  -d dlp0
   ```

   A listing similar to the following is displayed:

   ```
   dlp0:
   FILE = /dev/lp0
   BACKEND = /usr/lib/lpd/piobe
   ```

2. To display device stanza `dlp0` on the `lp0` queue in colon format, type:

   ```
   lsquedev  -c  -qlp0  -d dlp0
   ```

   A listing similar to the following is displayed:

   ```
   file:access:feed:header:trailer:backend:align
   dlp0:/dev/lp0:read:never:never:never:/usr/lib/lpd/piobe:TRUE
   ```

Files

Item	Description
/usr/bin/lsquedev	Contains the **lsquedev** command.
/etc/qconfig	Contains the configuration file.

lsresource Command

Purpose

Displays bus resources for available devices in the system and recommends attribute values for bus resource resolution.

Syntax

lsresource [**-a** | **-r**] [**-d**] **-l** *Name*

Description

The **lsresource** command writes a list of assigned bus resources to standard out, or determines if the bus resources for devices resolve.

The **lsresource** command lets you display the currently assigned values for the bus resource attributes for the device specified by the given device logical name (**-l** *Name*). Use the **-a** flag to display the currently assigned values for all bus resource attributes for all devices under the same parent bus as the specified device.

Use the **-r** flag to determine if the bus resources for the specified device are resolvable. In this case, the **lsresource** command checks all devices under the same parent bus as the specified device, including defined but not available devices, to see if their bus resource attributes are resolvable. The **lsresource** command produces no output if all attributes resolve. The **lsresource** command provides information depending on the type of conflict detected if any device's bus resources are unresolvable. In some cases, the **lsresource** command can provide you with information that leads to the resolution of the conflict.

The **lsresource** command identifies the device name, attribute name, and a suggested value for the attribute if a conflict results from an attribute that only a user can change. Setting the attribute to the suggested value should resolve the conflict. This may aid in the configuration of devices with attributes that can only a user can change. Such devices include adapter cards which use jumpers or switches on the card to select values.

In some cases, a conflict may be due to an attribute which the system can normally adjust at boot time but is prevented from doing so at run time because the device is in the Available state. In these situations, the **lsresource** command will indicate that the configuration will be resolved by rebooting the system.

It is possible that multiple user changeable attributes will be identified when unresolvable conflicts occur. These may be for the device specified by the given device logical name (**-l** *Name*) or for other devices in the system. All of the identified attributes will need to be changed to resolve the conflict. It may even be the case where user changeable attributes are identified and a reboot is indicated. In this case, all of the identified attributes will need to be changed and the system rebooted to resolve the conflicts.

Finally, **lsresource** may determine that the set of devices currently defined in the devices configuration database can not be resolved regardless of attributes being changed or the system rebooted. In this case, a list of the devices which could not be resolved is written to standard out. If the problem has resulted from a new device just being defined, that device should be removed, or the devices listed by lsresource should be removed. If the problem is not resolved by removing devices, there could be additional problems on the next reboot. This is because the order in which devices are resolved at boot time may differ from the order they are resolved by lsresource, resulting in a different set of unresolvable devices at boot time. If the set of unresolvable devices at boot time should now include a device needed for booting, problems such as no console being configured or the system failing to boot could occur.

The following applies when lsresource is used to list currently assigned bus resource values (the **-r** flag is not specified).

The **TYPE** field in the output listing contains the following symbols:

Item	Description
B	Bus Memory Address Values
M	Bus Memory Address Values
O	I/O Address Values
I	Bus Interrupt Levels
N	Non-sharable Bus Interrupt Levels
A	DMA Arbitration Level

The **S** column denotes shared attributes. These are attributes which are required to be set to the same value. They are grouped by the number specified in the column. All attributes with a 1 by them must be set to the same value, all attributes with a 2 by them must be set to the same value, and so on. In some cases, two or more interrupt attributes may be set to the same value but have no numbers in the **S** column indicating that they are shared. This is because the values are not required to be the same but just happen to be set to the same value because they could not be assigned their own unique values.

The **G** column denotes attributes in a group. These are a set of attributes whose values depend on each other. If one is changed to the next possible value, the rest of the attributes in the group must also be changed to the next possible value. Their groupings are indicated by the number specified in the column. All attributes with a 1 by them are in the same group, all attributes with a 2 by them are same group, and so on.

On some models, the interrupt value displayed may be followed by a value enclosed in parenthesis. This is not part of the interrupt value but serves to identify the interrupt controller to which the interrupt is associated. The identifier consists of a letter followed by a number, such as A0. The letter indicates the type of interrupt controller and the number distinguishes between multiple instances of that type of controller. There are two types of interrupt controllers that may be identified:

Item	Description
A	Indicates an AT interrupt controller.
B	Indicates a non-AT interrupt controller.

Flags

Item	Description
-a	Specifies that all allocated bus resource attributes for all devices connected to the same top parent bus as the device specified with the **-l** flag are to be displayed. This flag cannot be used with the **-r** flag.
-d	Specifies that the attribute text descriptions are to be included in the output.
-l *Name*	(Lowercase L) Specifies the logical name of the device attributes to display.
-r	Specifies to attempt to resolve all bus resources of all devices connected to the same top parent bus as the device specified with the **-l** flag. This will include all devices that are in the DEFINED state. The **lsresource** command will display any conflicts and advise the user on changeable values. No changes to the ODM database are made. This flag cannot be used with the **-a** flag.

Security

Access Control: Any User

Auditing Events: N/A

Examples

1. To list bus attributes for the token ring device, enter:

   ```
   lsresource -l tok0
   ```

   The system displays a message similar to the following:

   ```
   TYPE   DEVICE         ATTRIBUTE       S G CURRENT VALUE
   ----- --------------  --------------- - - ----------------------
   M     tok0            dma_bus_mem         0x003b2000 - 0x003f1fff
   O     tok0            bus_io_addr         0x000086a0 - 0x000086af
   N     tok0            bus_intr_lvl                3
   A     tok0            dma_lvl                     7
   ```

2. To list bus attributes for all devices, enter:

   ```
   lsresource -a -l tok0
   ```

   The system displays a message similar to the following:

```
TYPE  DEVICE          ATTRIBUTE          S G   CURRENT VALUE
----- --------------  ---------------    - -    ----------------------
M      bus0            bus_iocc_mem             0x00fffff0 - 0x00ffffff
M      gda0            vram_start         1     0x00400000 - 0x007fffff
M      gda0            bus_mem_start            0x000c0000 - 0x000c1fff
M      gda0            dma1_start               0x00800000 - 0x009fffff
M      gda0            dma2_start               0x00a00000 - 0x00bfffff
M      gda0            dma3_start               0x00c00000 - 0x00dfffff
M      gda0            dma4_start               0x01000000 - 0x011fffff
M      scsi0           bus_mem_addr             0x000e0000 - 0x000e0fff
M      scsi0           dma_bus_mem              0x00100000 - 0x00301fff
M      tok0            dma_bus_mem              0x003b2000 - 0x003f1fff
O      da0             bus_io_addr              0x00000060 - 0x0000006f
O      siokta0         bus_io_addr              0x00000050 - 0x00000051
O      sioma0          bus_io_addr              0x00000048 - 0x00000049
O      ppa0            bus_io_addr              0x00000078 - 0x0000007a
O      gda0            bus_addr_start     1     0x00002110 - 0x0000211f
O      tok0            bus_io_addr              0x000086a0 - 0x000086af
I      siokta0         bus_intr_lvl                         1   (A0)
I      sioma0          bus_intr_lvl                         1   (A0)
I      ppa0            bus_intr_lvl                        13   (A0)
I      gda0            int_level                            9   (A0)
I      scsi0           bus_intr_lvl                        14   (A0)
N      fda0            bus_intr_lvl                         6   (A0)
N      tok0            bus_intr_lvl                         3   (A0)
A      fda0            dma_lvl                              0
A      gda0            dma_channel                          3
A      scsi0           dma_lvl                              4
A      tok0            dma_lvl                              7
```

3. To report the outcome of a resolution of device attributes, enter:

```
lsresource -r - d -l tok0
```

Depending on the outcome of the resolution, different messages may be displayed. The output below signifies to a user that the resolution can be successful if changes are made, i.e., the attributes are changed to the suggested values.

```
lsresource: The attribute(s) for some device(s) in the system could
not be resolved.  To resolve conflicts, attribute(s) need to be
modified.  A suggested value for each attribute is provided.

DEVICE          ATTRIBUTE        CURRENT   SUGGESTED  DESCRIPTION
--------------  ---------------  -------   ---------  -----------
ent1            bus_intr_lvl     11        5          Bus interrupt level
ent1            bus_mem_addr     0xc0000   0xc4000    Bus memory address
ent1            bus_io_addr      0x300     0x320      Bus I/O address
ent2            bus_intr_lvl     11        7          Bus interrupt level
ent2            bus_mem_addr     0xc0000   0xc8000    Bus memory address
```

Files

Item	Description
/usr/sbin/lsresource	Contains the **lsresource** command.

lsresponse Command

Purpose

Lists information about one or more responses.

Syntax

`lsresponse` [`-a`] [`-C` | `-l` | `-t` | `-d` | `-D` *delimiter*] [`-A`] [`-q`] [`-U`] [`-x`] [**`-b`**] [`-h`] [`-TV`] [*response1*[`,`*response2*`,`...] `:`*node_name*]

Description

The `lsresponse` command lists the following information about defined responses:

Field	Description
ResponseName	The name of the response.
Node	The location of the response.
Action	The name of an action.
DaysOfWeek	The days of the week when the action can be run. `DaysOfWeek` and `TimeOfDay` together define the interval when the action can be run. The values for the days can be separated by plus signs (+) or displayed as a range of days separated by a hyphen (-). Multiple `DaysOfWeek` values are separated by commas (,). The number of `DaysOfWeek` values must match the number of `TimeOfDay` values. The values for each day follow: **1** Sunday **2** Monday **3** Tuesday **4** Wednesday **5** Thursday **6** Friday **7** Saturday
TimeOfDay	The time range when `Action` can be run, consisting of the start time followed by the end time separated by a hyphen. `DaysOfWeek` and `TimeOfDay` together define the interval when the action can be run. The time is in 24-hour format (HHMM), where the first two digits represent the hour and the last two digits represent the minutes. Multiple `TimeOfDay` values are separated by commas (,). The number of `DaysOfWeek` values must match the number of `TimeOfDay` values.
ActionScript	The script or command to run for the action.
ReturnCode	The expected return code for `ActionScript`.
CheckReturnCode	Indicates whether the actual return code for `ActionScript` is compared to its expected return code. The values are: y (yes) and n (no).
EventType	The type of event that causes the action to be run: event, rearm event, or both.
StandardOut	Indicates whether standard output is directed to the audit log. The values are: y (yes) and n (no).
EnvironmentVars	Indicates any environment variables that will be set before the action is run.
UndefRes	Indicates whether the action is to be run if a monitored resource becomes undefined. The values are: y (yes) and n (no).

Field	Description
Locked	Indicates whether the resource is locked or unlocked.
EventBatching	Indicates whether the response action supports event batching.

To get a list of all response names, run the `lsresponse` command alone without any response names specified. A list of all response names is returned. The default format in this case is tabular.

Specifying a node name after the response names limits the display to the responses defined on that node. List all of the responses on a node by specifying a colon (`:`) followed by the node name. The node name is a node within the management scope determined by the CT_MANAGEMENT_SCOPE environment variable. The management scope determines the list of nodes from which the responses are listed. For local scope, only responses on the local node are listed. Otherwise, the responses from all nodes within the domain are listed.

To see all the information about all response names, specify the `-A` flag with the `lsresponse` command. The `-A` flag causes all information about a response to be listed when no response names are specified. When all of the information about all responses is listed, the long format is the default.

When more than one response is specified, the response information is listed in the order in which the responses are entered.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

Flags

-a
Specifies that this command applies to all nodes in the cluster. The cluster scope is determined by the CT_MANAGEMENT_SCOPE environment variable. If it is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management and peer domain exist, `lsresponse -a` with CT_MANAGEMENT_SCOPE not set will list the management domain. In this case, to list the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-A
Displays all of the attributes of the response.

-b
Displays only the responses that support event batching.

-C
Displays the **mkresponse** command that can be used to create the response and one of its actions. If more than one response is specified, each **mkresponse** command appears on a separate line. This flag is ignored when no responses are specified. This flag overrides the **-l** flag.

-d
Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the –D flag if you wish to change the default delimiter.

-D *delimiter*
Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify something other than the default, colon (:). For example, when the data to be displayed contains colons, use this flag to specify another delimiter of one or more characters.

-l
Displays the response information on separate lines (long form).

-q
Does not return an error when `response` does not exist.

-t
Displays the response information in separate columns (table form).

-U
Indicates whether the resource is locked.

-x
Suppresses headers when printing.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

***response1*[,*response2*,...]**
This parameter can be a response name or a substring of a response name. You can specify more than one response name. When it is a substring, any defined response name that contains the substring is listed.

node_name
Specifies the node where the response is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

Security

The user needs read permission for the `IBM.EventResponse` resource class to run `lsresponse`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *Administering RSCT* guide for details on the ACL file and how to modify it.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To list all of the responses, run this command:

```
lsresponse
```

The output will look like this:

```
ResponseName
"E-mail root anytime"
"E-mail root first shift"
"Critical notifications"
"Generate SNMP trap"
```

2. To see which resources are locked, run this command:

```
lsresponse -U
```

The output will look like this:

```
ResponseName                    Node      Locked
"Broadcast event on-shift"      "nodeA"   "No"
"E-mail root off-shift"         "nodeA"   "No"
"E-mail root anytime"           "nodeA"   "No"
"Log event anytime"             "nodeA"   "No"
"Informational notifications"   "nodeA"   "No"
"Warning notifications"         "nodeA"   "No"
```

```
"Critical notifications"       "nodeA"   "No"
"Generate SNMP trap"           "nodeA"   "No"
```

3. To list general information about the response "Critical notifications", run this command:

```
lsresponse "Critical notifications"
```

The output will look like this:

```
ResponseName     = "Critical notifications"
Node             = "nodeA"
Action           = "Log Critical Event"
DaysOfWeek       = 1+2+7
TimeOfDay        = 0000-2400
ActionScript     = "/opt/rsct/bin/logevent /tmp/
criticalEvents"
ReturnCode       = 0
CheckReturnCode  = "y"
EventType        = "b"
StandardOut      = "y"
EnvironmentVars  = '"Env1=5","Env=10"'
UndefRes         = "n"

ResponseName     = "Critical notifications"
Node             = "nodeA"
Action           = "E-mail root"
DaysOfWeek       = 6+2,6+2,6+5
TimeOfDay        = 1700-2400,0000-0800,0000-2400
ActionScript     = "/opt/rsct/bin/notifyevent root"
ReturnCode       = 0
CheckReturnCode  = "y"
EventType        = "b"
StandardOut      = "y"
EnvironmentVars  = ""
UndefRes         = "n"
```

4. To list the command that would create the response "Critical notifications" along with one of its actions, run this command:

```
lsresponse -C "Critical notifications"
```

The output will look like this:

```
mkresponse -n "Log Critical Event" -d 1+2+7 -t 0000-2400 \
-s "usr/sbin/rsct/bin/logevent /tmp/criticalEvents"  \
-e b -r 0 "Critical notifications"
```

5. To list all responses that have the string `E-mail` in their names, run this command:

```
lsresponse "E-mail"
```

The output will look like this:

```
ResponseName = "E-mail root anytime"
Action       = "E-mail root"
⋮
ResponseName = "E-mail root first shift"
Action       = "E-mail root"
```

Location

/opt/rsct/bin/lsresponse

lsrole Command

Purpose

Displays role attributes.

Syntax

lsrole [**-R** *load_module*] [**-c** | **-f** | **-C**] [**-a** *List*] { **ALL** | *Name* [,*Name*] ... }

Description

The **lsrole** command displays the role attributes. You can use this command to list all attributes of all the roles or all the attributes of specific roles. Since there is no default parameter, you must enter the **ALL** keyword to see the attributes of all the roles. By default, the **lsrole** command displays all role attributes. To view selected attributes, use the **-a** *List* flag. If one or more attributes cannot be read, the **lsrole** command lists as much information as possible.

By default, the **lsrole** command lists each role's attributes on one line. It displays attribute information as *Attribute***=***Value* definitions, each separated by a blank space. To list the role attributes in stanza format, use the **-f** flag. To list the information as colon-separated records, use the **-c** flag.

You can use the System Management Interface Tool (SMIT) **smit lsrole** fast path to run this command.

If the system is configured to use multiple domains for the role database, the roles, as specified by the *Name* parameter, are searched from the domains in the order specified by the **secorder** attribute of the roles stanza in the **/etc/nscontrol.conf** file. If duplicate entries exist in multiple domains, only the first entry instance is listed. Use the **-R** flag to list the roles from a specific domain.

The **lsrole** command only lists the role definitions available in the roles database. If the system is operating in enhanced Role Based Access Control (RBAC) mode, the information in the roles database might differ from what is used for security considerations on the system in the kernel security tables (KST). To view the state of the roles database in the KST, use the **lskst** command.

Flags

Item	Description
-a *List*	Lists the attributes to display. The *List* variable can include any attribute that is defined in the **chrole** command. Specify more than one attributes with a blank space between attribute names. If an empty list is specified, only the role names are displayed. In addition to the attributes defined in the **chrole** command, the following attributes can also be listed with the **-a** flag: **all_auths** Traverses the role hierarchy of the specified roles and gathers all the authorizations. The **all_auths** attribute differs from the **authorizations** attribute because the **lsrole** command only lists the explicit authorizations of the specified roles for that attribute. **users** Displays the users that are granted the specified roles. **description** Displays the text description of the role as indicated by the **dfltmsg**, **msgcat**, **msgset** and **msgnum** attributes for the role.
-c	Displays the role attributes in colon-separated records, as follows: `# role:  attribute1:  attribute2:  ...` `  Role:  value1:      value2:      ...`

Item	Description
-C	Displays the role attributes in colon-separated records that are easier to parse than the output of the **-c** flag: `#role:attribute1:attribute2: ...` `role:value1:value2: ...` `role2:value1:value2: ...` The output is preceded by a comment line that has details about the attribute represented in each colon-separated field. If you specified the **-a** flag, the order of the attributes matches the order specified in the **-a** flag. If a role does not have a value for a given attribute, the field is still displayed but is empty. The last field in each entry is ended by a newline character rather than a colon.
-f	Displays the output in stanzas, with each stanza identified by a role name. Each *Attribute*=*Value* pair is listed on a separate line: `Role:` `attribute1=value` `attribute2=value` `attribute3=value`
-R *load_module*	Specifies the loadable module to list roles from.

Security

The **lsrole** command is a privileged command. You must assume a role that has the following authorization to run the command successfully.

Item	Description
aix.security.role.list	Required to run the command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files Accessed:

Mode	File
r	/etc/security/roles

Examples

1. To display the role `rolelist` and groups of the role `ManageAllUsers` in a colon format, use the following command:

   ```
   lsrole -c -a rolelist groups ManageAllUsers
   ```

   Information similar to the following appears:

   ```
   # role: rolelist:groups
    ManageAllUsers: ManagerBasicUser:security
   ```

2. To list all attributes of the `ManageAllUsers` role from LDAP, use the following command:

   ```
   lsrole -R LDAP ManageAllUsers
   ```

   All the attribute information appears, with each attribute separated by a blank space.

Files

Item	Description
/etc/security/roles	Contains the attributes of roles.

lsrpdomain Command

Purpose

Displays peer domain information for the node.

Syntax

`lsrpdomain` [`-o` | `-O`] [`-l` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] [*peer_domain*]

Description

The `lsrpdomain` command displays information about the peer domains that the node where the command runs belongs to. Use the command's flags and parameters to specify which information you want to display and how you want to display it. When you specify the name of a peer domain, the command displays information about that peer domain only. The `-o` and `-O` flags also limit the information this command displays. The `-o` flag displays information only about the online peer domain. The `-O` flag displays information only about peer domains that are offline.

By default, the `lsrpdomain` command displays information in table format (`-t`).

Some of the peer domain information that is displayed follows:

Field	Description
Name	The name of the peer domain.
RSCTActiveVersion	The version of RSCT that is active in the peer domain.
MixedVersions	Indicates whether more than one version of RSCT is active in the peer domain.
TSPort	The topology services port number.
GSPort	The group services port number.
OpState	The current state of the peer domain.

Flags

-o
Displays information about the node's online peer domain.

-O
Displays information about peer domains that are offline for the node.

-l
Displays the information on separate lines (long format).

-t
Displays the information in separate columns (table format). This is the default.

-d
Displays the information using delimiters. The default delimiter is a colon (`:`). Use the `-D` flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default colon (`:`) — when the information you want to display contains colons, for example. You can use this flag to specify a delimiter of one or more characters.

-x
Excludes the header (suppresses header printing).

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

peer_domain
Specifies the name of the peer domain about which you want to display information. You can specify a peer domain name or a substring of a peer domain name for this parameter. If you specify a substring, the command displays information about any defined peer domain with a name that contains the substring.

Security

The user of the `lsrpdomain` command needs read permission for the `IBM.PeerDomain` resource class on the node on which the command runs. By default, `root` on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

6
The peer domain definition does not exist.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on the node for which the peer domain information is requested.

Implementation Specifics

This command is part of the **rsct.basic.rte** fileset for the AIX® operating system.

Standard Input

When the `-f "-"` or `-F "-"` flag is specified, this command reads one or more node names from standard input.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To display general information about the peer domains to which nodeA belongs, run this command on nodeA:

   ```
   lsrpdomain
   ```

   The output will look like this:

   ```
   Name            OpState         RSCTActiveVersion       MixedVersions   TSPort  GSPort
   ApplDomain      Online          2.5.0.0                  No              12347   12348
   ```

2. To display general information about the peer domains to which nodeA belongs, with the default delimiter (but without the heading), run this command on nodeA:

   ```
   lsrpdomain -xd
   ```

   The output will look like this:

   ```
   ApplDomain:Online:2.5.0.0:No:12347:12348:
   ```

3. To display general information about the peer domains to which nodeA belongs, in long format, run this command on nodeA:

   ```
   lsrpdomain -l
   ```

   The output will look like this:

   ```
   Name              = ApplDomain
   OpState           = Online
   RSCTActiveVersion = 2.5.0.0
   MixedVersions     = No
   ```

```
TSPort           = 12347
GSPort           = 12348
```

Location

/opt/rsct/bin/lsrpdomain

lsrpnode Command

Purpose

Displays information about one or more of the nodes that are defined in the online peer domain.

Syntax

lsrpnode [[**-i**] [**-l** | **-t** | **-d** | **-D** *delimiter*] **-o** | **-O** | **-L**] **-P**] **-Q**] [**-B**] [**-x**] [**-h**] [**-TV**] [*node_name*]

`lsrpnode -p` *peer_domain* [`-l` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`]

Description

The **lsrpnode** command displays information about one or more of the nodes that are defined in the online peer domain. Use the command's flags and parameters to specify which information you want to display and how you want to display it. When you specify a node name, the command displays information about that node only.

The **-o**, **-O**, and **-L** flags also limit the information this command displays. The **-o** flag displays information about nodes that are online. The **-O** flag displays information about nodes that are offline. The **-L** flag displays information about the local node, which is the node the command runs on.

The **-P** flag displays additional node configuration information related to group services group leader selection. The **-Q** flag displays additional node configuration information related to quorum decisions. The **-B** flag displays additional node configuration information related to the tiebreaker mechanism.

By default, the `lsrpnode` command displays information in table format (`-t`).

Some of the node information that is displayed follows:

Field	Description
Name	The name of the node in the peer domain.
OpState	The operational state of the node.
RSCTVersion	The version of RSCT that is active in the node.

The following fields are displayed when you specify the `-i` flag:

Field	Description
NodeNum	The node number used by topology services and group services. This number is unique within the cluster.
NodeID	The unique node identifier.

Along with other fields (depending on the flags specified), this field is displayed when you specify the **-P** flag:

Field	Description
Preferred	Indicates whether the node is a group services group leader candidate.

Along with other fields (depending on the flags specified), this field is displayed when you specify the **-Q** flag:

Field	Description
Quorum	Indicates whether the node participates in quorum decisions.

Along with other fields (depending on the flags specified), this field is displayed when you specify the **-B** flag:

Field	Description
Tiebreaker	Indicates whether the node has access to the peer domain's tiebreaker mechanism.

See the *Administering RSCT* guide for information about group services group leader selection, quorum decisions, and the tiebreaker mechanism.

Flags

-d
Displays the information using delimiters. The default delimiter is a colon (`:`). Use the `-D` flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default colon (`:`) — when the information you want to display contains colons, for example. You can use this flag to specify a delimiter of one or more characters.

-i
Displays the node number and node ID for the node. The node number is used by topology services and group services and is unique within the cluster. The node ID is the unique node identifier.

-l
Displays the information on separate lines (long format).

-L
Displays information about the local node only, which is the node that the command runs on.

-o
Displays information about the nodes that are online in the peer domain.

-O
Displays information about the nodes that are offline in the peer domain.

-p *peer_domain*
Displays information about nodes defined in an *offline* peer domain that the local node belongs to. (By default, the `lsrpnode` command displays information about the nodes that are defined in the domain where you are currently *online*.) However, this information might not reflect changes that are made to the domain after the local node is taken offline, because an offline node might not have the latest configuration.

The `-p` flag ignores the `CT_CONTACT` environment variable. You must have root access to use the **-p** flag.

-P
Indicates whether the node is a group services group leader candidate. **yes** is displayed if the node can be a group services group leader. **no** is displayed if the node cannot be a group services group leader. See the *Administering RSCT* for more information about group services group leader selection.

-Q
Indicates whether the node participates in quorum decisions. yes is displayed if the node participates in quorum decisions. no is displayed if the node does not participate in quorum decisions. See the *Administering RSCT* for more information on quorum decisions.

-B
Indicates whether the node has access to the peer domain's tiebreaker mechanism. yes is displayed if the node has access to the peer domain's tiebreaker mechanism. no is displayed if the node does not have access to the peer domain's tiebreaker mechanism. See the *Administering RSCT* for more information on the tiebreaker mechanism.

-t
Displays the information in separate columns (table format). This is the default format.

-x
Excludes the header (suppresses header printing).

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

node_name
Specifies the name of the node about which you want to display information. You can specify a node name or a substring of a node name for this parameter. If you specify a substring, the command displays information about any defined node with a name that contains the substring.

Security

The user of the `lsrpnode` command needs read permission for the `IBM.PeerNode` resource class on the node this command runs on. By default, `root` on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is online in the peer domain.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the `-f "-"` or `-F "-"` flag is specified, this command reads one or more node names from standard input.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To display general information about the nodes in the online peer domain that nodeA belongs to, run this command on nodeA:

   ```
   lsrpnode
   ```

   The output will look like this:

   ```
   Name    OpState   RSCTVersion
   nodeA   Online    3.1.4.0
   nodeB   Online    3.1.4.0
   nodeC   Offline   3.1.4.0
   ```

2. To display general information about the nodes in the online peer domain that nodeA belongs to, with the default delimiter (but without the heading), run this command on nodeA:

   ```
   lsrpnode -xd
   ```

   The output will look like this:

   ```
   nodeA:Online:3.1.4.0:
   nodeB:Online:3.1.4.0:
   nodeC:Offline:3.1.4.0:
   ```

3. To display general information about the nodes in the online peer domain that nodeA belongs to, in long format, run this command on nodeA:

   ```
   lsrpnode -l
   ```

   The output will look like this:

   ```
   Name        = nodeA
   OpState     = Online
   RSCTVersion = 3.1.4.0
   ```

```
Name        = nodeB
OpState     = Online
RSCTVersion = 3.1.4.0

Name        = nodeC
OpState     = Offline
RSCTVersion = 3.1.4.0
```

4. To display general information about the nodes in the online peer domain that nodeA belongs to, including the node number and node ID, run this command on nodeA:

```
lsrpnode -i
```

The output will look like this:

```
Name    OpState   RSCTVersion   NodeNum   NodeID
nodeA   Online    3.1.4.0       2         40a514bed9d82412
nodeB   Online    3.1.4.0       1         47fe57098f4ec4d9
```

5. To display general information about the nodes in the online peer domain to which **nodeA** belongs, including the preferred group services group leader information, run this command on **nodeA**:

```
lsrpnode -P
```

The output will look like this:

```
Name    OpState   RSCTVersion   Preferred
nodeA   Online    3.1.4.0       yes
nodeB   Online    3.1.4.0       no
```

6. To display general information about the nodes in the online peer domain to which **nodeA** belongs, including the quorum information, run this command on **nodeA**:

```
lsrpnode -Q
```

The output will look like this:

```
Name    OpState   RSCTVersion   Quorum
nodeA   Online    3.1.4.0       no
nodeB   Online    3.1.4.0       yes
nodeC   Online    3.1.4.0       yes
```

7. To display general information about the nodes in the online peer domain to which **nodeA** belongs, including quorum and tiebreaker information, run this command on **nodeA**:

```
lsrpnode -QB
```

The output will look like this:

```
Name    OpState   RSCTVersion   Quorum        Tiebreaker
nodeA   Online    3.1.4.0       no             no
nodeB   Online    3.1.4.0       yes            yes
nodeC   Online    3.1.4.0       yes            yes
```

Location

/opt/rsct/bin/lsrpnode

lsrset Command

Purpose

Displays system rset contents.

Syntax

lsrset [**-X**] [-f] [-v | -o] [[**-S**] -r *rsetname* | -n *namespace* | -a]

or

lsrset [**-X**] [-P] [-v | -o] -p *pid*

Description

The **lsrset** command displays information contained in rsets stored in the system registry or rsets attached to a process.

Flags

Item	Description
-f	Displays rset owner, group, and mode data.
-v	Verbose mode. Displays resources contained in the rset, rset owner, group and mode data.
-o	Displays only the online resources contained in the rset. The default is to display all resources.
-p *pid*	Displays the effective rset attached to this process.
-r *rsetname*	Displays the rset with this name in the system registry. The name consists of a namespace and an *rsname* separated by a "/" (slash). Both the *namespace* and *rsname* may contain up to 255 characters. See the **rs_registername()** service for additional information about character set limits of rset names.
-n *namespace*	Displays all rsets in this *namespace* in the system registry.
-a	Displays all rsets in the system registry.
-P	Displays the partition rset attached to the specified process.
-S	Displays the resources contained in this rset if it were to be scheduled with the `-S` hint with either the `execrset` or the `attachrset` command. The rset does not need to be an exclusive rset. This is to be contrasted with the `attachrset` and `execrset` commands, which require exclusive rsets to be specified with the `-S` flag.
-X	Prints all available characters of each user and group name instead of truncating to the first 8 characters.

Examples

1. To display all resources for all rsets in the system registry, type:

```
lsrset -v -a
```

2. To display a summary of the effective rset attached to pid 28026, type:

```
lsrset -p 28026
```

3. To display the online resources in the effective rset attached to pid 28026, type:

```
lsrset -o -p 28026
```

4. To display all the resources in the effective rset attached to pid 28026, type:

```
lsrset -v -p 28026
```

5. To display online resources for all rsets in the system registry, type:

```
lsrset -a -o
```

6. To display all resources for all rsets in the system registry with expanded user and group name, type:

```
lsrset  -X -v -a
```

Files

Item	Description
/etc/bin/lsrset	Contains the **lsrset** command

lsrsrc Command

Purpose

Displays attributes and values for a resource or a resource class.

Syntax

To display the attributes and values for a *resource:*

`lsrsrc` [`-s` *"selection_string"*] [`-a` | `-N` { *node_file* | `"-"` }] [`-A` `p` | `d` | `b`] [`-p` *property*] [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] [*resource_class*] [*attr*...]

`lsrsrc -r` [`-s` *"selection_string"*] [`-a` | `-N` { *node_file* | `"-"` }] [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] [*resource_class*]

To display the attributes and values for a *resource class:*

`lsrsrc -c` [`-A` `p` | `d` | `b`] [`-p` *property*] [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-a`] [`-h`] [`-TV`] *resource_class* [*attr*...]

`lsrsrc -C` *domain_name...* [`-A` `p` | `d` | `b`] [`-p` *property*] [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] *resource_class* [*attr*...]

To display a list of all of the resource classes:

`lsrsrc`

Description

The `lsrsrc` command displays the persistent and dynamic attributes and their values for a resource or a resource class.

Instead of specifying multiple node names in *selection_string*, you can use the `-N` *node_file* flag to indicate that the node names are in a file. Use `-N "-"` to read the node names from standard input.

When one or more attribute names are specified, these names and their values are displayed in the order specified, provided that each of the specified attribute names is valid. When no attribute names are specified:

- the `-A p | d | b` flag controls whether persistent attributes or dynamic attributes or both — and their values — are displayed.
- only attributes that are defined as `public` are displayed. Use the `-p` flag to override this default.

For best performance, specify either the `-A p` flag or only persistent attributes as parameters.

Specify the `-r` flag to display only the resource handles associated with the resources for the specified resource class.

To display a list of the attributes and values for a resource class, specify the `-c` flag.

By default, the resource attributes and values are displayed in long format. Use the `-t`, `-d`, or `-D` flag to display the resources in table format or delimiter-formatted output.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM `nodegrp` command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference.*

The `lsrsrc` command does not list any attributes that have a datatype defined as `ct_none` (`Quantum`, for example). RMC does not return attribute values for attributes that are defined as `Quantum`. To list attribute definitions, use the `lsrsrcdef` command.

Flags

-a
Specifies that this command applies to all nodes in the cluster. The cluster scope is determined by the CT_MANAGEMENT_SCOPE environment variable. If it is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management and peer domain exist, `lsrsrc -a` with CT_MANAGEMENT_SCOPE not set will list the management domain. In this case, to list the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-A p | d | b
Specifies an attribute type. By default only persistent attributes are displayed. This flag can be used only when no attribute names are specified on the command line.

p
Displays only persistent attributes.

d
Displays only dynamic attributes.

b
Displays both persistent and dynamic attributes.

For best performance, specify the `-A p` flag.

-c
Displays the attributes for the resource class. This flag overrides the `-r` flag.

-C *domain_name...*
Displays the class attributes of a globalized resource class on one or more RSCT peer domains that are defined on the management server. Globalized classes are used in peer domains and management domains for resource classes that contain information about the domain. To display class attributes of a globalized resource class on all peer domains defined on the management server, use the `-c` flag with the `-a` flag instead of `-C`. The command returns the name of the peer domain in the form of an attribute `ActivePeerDomain`. This is not an actual attribute, but is presented as such to indicate which peer domain is being displayed.

-d
Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the `-D` flag if you want to change the default delimiter.

-D *delimiter*
Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify something other than the default colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

-i
Generates a template of *resource_data_input_file* that can then, after appropriate editing, be used as input to the `mkrsrc` command. The output is displayed in long (stanza) format. All required and optional attributes that can be used to define a resource are displayed. The attribute data type is displayed as the value in the *attr=value* pairs. It is suggested that when you use this flag, the output of the `lsrsrc` command be directed to a file. This flag overrides the `-s` and `-A d` flags.

-l

Specifies long formatted output. Each attribute is displayed on a separate line. This is the default display format. If the `lsrsrc` command is issued with the `-l` flag, but without a resource class name, the `-l` flag is ignored when the command returns the list of defined resource class names.

-N { *node_file* | "-" }

Specifies that node names are read from a file or from standard input. Use **-N** *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*.
- A number sign (#) in column 1 indicates that the line is a comment.
- Any blank characters to the left of a node name are ignored.
- Any characters to the right of a node name are ignored.

Use **-N** "-" to read the node names from standard input.

The **CT_MANAGEMENT_SCOPE** environment variable determines the scope of the cluster. If **CT_MANAGEMENT_SCOPE** is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and **CT_MANAGEMENT_SCOPE** is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set **CT_MANAGEMENT_SCOPE** to 2.

-p *property*

Displays attributes with the specified *property*. By default, only public attributes are displayed. To display all of the attributes regardless of the property, use the `-p 0` flag. Use this flag in conjunction with the `-A` flag when no attributes are specified on the command line.

Persistent attribute properties:

0x0001
`read_only`

0x0002
`reqd_for_define` (required)

0x0004
`inval_for_define` (not valid)

0x0008
`option_for_define` (optional)

0x0010
`selectable`

0x0020
`public`

Dynamic attribute properties:

0x0020
`public`

A decimal or hexadecimal value can be specified for the property. To display attributes and their values for all attributes that have one or more properties, "OR" the properties of interest together and then specify the "OR"ed value with the `-p` flag. For example, to display attributes and their values for all persistent attributes that are either `reqd_for_define` or `option_for_define`, enter:

```
lsrsrc -p 0x0a
```

-r

Displays the resource handles for the resources that match the specified selection string or all resources when no selection string is specified.

-s *"selection_string"*
Specifies a selection string. All selection strings must be enclosed within either double or single quotation marks. If the selection string contains double quotation marks, enclose the entire selection string in single quotation marks. For example:

```
-s 'Name == "testing"'
```

```
-s 'Name ?= "test"'
```

Only persistent attributes may be listed in a selection string. For information on how to specify selection strings, see the *RSCT: Administration Guide* .

-t
Specifies table format. Each attribute is displayed in a separate column, with one resource per line.

-x
Suppresses header printing.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software-service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

resource_class
Specifies the name of the resource class with the resources that you want to display.

attr...
Specifies one or more attribute names. Both persistent and dynamic attribute names can be specified to control which attributes are displayed and their order. Zero or more attributes can be specified. Attributes must be separated by spaces.

Security

The user needs read permission for the *resource_class* specified in `lsrsrc` to run `lsrsrc`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for information about the ACL file and how to modify it.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To list the names of all of the resource classes, enter:

```
lsrsrc
```

The output will look like this:

```
class_name
"IBM.Association"
"IBM.Condition"
'IBM.EventResponse"
"IBM.Host"
"IBM.Ethernet"
"IBM.TokenRing"
...
```

2. To list the persistent attributes for resource IBM.Host that have 4 processors, enter:

```
lsrsrc -s "NumProcessors == 4" -A p -p 0 IBM.Host
```

   The output will look like this:

```
Resource Persistent Attributes for: IBM.Host
resource 1:
        Name           = "c175n05.ppd.pok.ibm.com"
        ResourceHandle = "0x4008 0x0001 0x00000000 0x0069684c 0x0d7f55d5
0x0c32fde3"
        Variety        = 1
        NodeList       = {1}
        NumProcessors  = 4
        RealMemSize    = 1073696768
```

3. To list the public dynamic attributes for resource IBM.Host on node 1, enter:

```
lsrsrc -s 'Name == "c175n05.ppd.pok.ibm.com"' -A d IBM.Host
```

   The output will look like this:

```
Resource Dynamic Attributes for: IBM.Host
resource 1:
        ProcRunQueue        = 1.03347987093142
        ProcSwapQueue       = 1.00548852941929
        TotalPgSpSize       = 65536
        TotalPgSpFree       = 65131
        PctTotalPgSpUsed    = 0.61798095703125
        PctTotalPgSpFree    = 99.3820190429688
        PctTotalTimeIdle    = 0
        PctTotalTimeWait    = 51.5244382399734
        PctTotalTimeUser    = 12.8246006482343
        PctTotalTimeKernel  = 35.6509611117922
        PctRealMemFree      = 66
        PctRealMemPinned    = 4
        RealMemFramesFree   = 173361
        VMPgInRate          = 0
        VMPgOutRate         = 0
        VMPgFaultRate       = 0
         ...
```

4. To list the Name, Variety, and ProcessorType attributes for the IBM.Processor resource on all the online nodes, enter:

```
lsrsrc IBM.Processor Name Variety ProcessorType
```

   The output will look like this:

```
Resource Persistent Attributes for: IBM.Processor
resource 1:
        Name          = "proc3"
        Variety       = 1
        ProcessorType = "PowerPC_604"
resource 2:
        Name          = "proc2"
        Variety       = 1
        ProcessorType = "PowerPC_604"
resource 3:
        Name          = "proc1"
        Variety       = 1
        ProcessorType = "PowerPC_604"
resource 4:
        Name          = "proc0"
        Variety       = 1
        ProcessorType = "PowerPC_604"
```

5. To list both the persistent and dynamic attributes for the resource class IBM.Condition, enter:

```
lsrsrc -c -A b -p 0 IBM.Condition
```

   The output will look like this:

```
Resource Class Persistent and Dynamic Attributes for: IBM.Condition
resource 1:
        ResourceType = 0
        Variety      =
0
```

6. To list the nodes in the cluster that have at least four processors, using the `/tmp/common/node_file` file:

```
# common node file
#
node1.ibm.com      main node
node2.ibm.com      main node
node4.ibm.com      backup node
node6.ibm.com      backup node
#
```

as input, enter:

```
lsrsrc -s "NumProcessors >= 4" -N /tmp/common/node_file -t IBM.Host \
Name NumProcessors
```

The output will look like this:

```
Resource Persistent Attributes for IBM.Host
Name              NumProcessors
"node1.ibm.com"   4
"node2.ibm.com"   4
```

Location

/opt/rsct/bin/lsrsrc

lsrsrcassoc Command

Purpose

Retrieves a list of resources that are associated with a class using an association provider.

Syntax

`lsrsrcassoc` [`-s` "*source_selection_string*"] [`-c` *association_class*] [`-d` *association_endpoint_class*] [`-S` "*destination_selection_string*"] [`-o` *role*] [`-R` *result_role*] [`-h`] [`-TV`] *source_class_name* [*property_list*...]

Description

You can use the `lsrsrcassoc` command to learn about the relationships among CIM resources.

This command is an interface into the association query mechanism of the Common Information Model (CIM) resource manager. Association providers that are registered with the CIM resource manager are called to retrieve association data. Before using `lsrsrcassoc`, it might be helpful to run the `lsassocmap` command to find out which association classes are known to the resource monitoring and control (RMC) subsystem.

You must specify a source class name with the `lsrsrcassoc` command. With no flags specified, `lsrsrcassoc` retrieves all resources associated with every resource of this class. Flags can be used to filter which associated resources are displayed.

The command output is similar to that of `lsrsrc`. Resources associated with a source resource are displayed with their class name and one attribute per line to facilitate searching and filtering the output.

Parameters

source_class_name
Specifies the source class in the association.

property_list
Specifies one or more property names. Only these properties (or attributes, in RMC terminology) of associated resources are displayed. If you do not specify this parameter, all property names are displayed.

Flags

-s *source_selection_string*
Specifies that only resources of the source class that match the selection string are used in the search for associated resources.

-S *destination_selection_string*
Specifies that only resources of the associated classes that match this selection string are displayed.

-c *association_class*
Limits the association search to only those resources tied to the source class through *association_class*.

-d *association_endpoint*
Limits the search of associated resources to just the members of this class.

-o *role*
The CIM association interface defines the *role* parameter as the name of the property referring to the class on the source side of the association. Typical values for this parameter are "GroupComponent" or "PartComponent", though the specific name must come from the association class definition.

-R *result_role*
Used like the `-o` flag, except this is the name of the property that refers to the destination side of the association.

-h
Writes the command usage statement to standard output.

-T
Writes the command trace messages to standard error. For your software service organization use only.

-V
Writes the command verbose messages to standard output.

Standard output

When the `-h` flag is specified, this command usage statement is written to standard output. When the `-V` flag is specified, this command verbose messages are written to standard output.

Standard error

When the `-T` flag is specified, this command trace messages are written to standard error.

Exit status

0
The command ran successfully.

1
An error occurred with the command-line interface (CLI) script.

2
An incorrect flag was specified on the command line.

3
An incorrect parameter was specified on the command line.

4
The source endpoint class was not found.

5
The destination endpoint class was not found.

6
The association class was not found.

Implementation specifics

This command is part of the `rsct.exp.cimrm` fileset, in the `rsct.exp` package on the AIX Expansion Pack and Reliable Scalable Cluster Technology (RSCT) package for the Linux operating system.

Location

Item	Description
`/opt/rsct/bin/lsrsrcassoc`	

Examples

To view instances of `cimv2.IBMAIX_UnixProcess` (for AIX) and `cimv2.Linux_UnixProcess` (for Linux) that are associated with `cimv2.IBMAIX_OperatingSystem` and `cimv2.Linux_OperatingSystem` respectively on the specified node, enter:

For AIX:

```
lsrsrcassoc -c cimv2.IBMAIX_OSProcess -s 'Name=~"c175nf14"' -S \
'Name=~"emacs"' cimv2.IBMAIX_OperatingSystem Handle Parameters
```

For Linux:

```
lsrsrcassoc -c
cimv2.Linux_OSProcess -s 'Name=~"c175nf14"' -S \
'Name=~"emacs"' cimv2.Linux_OperatingSystem Handle Parameters
```

In these examples:

- `-c cimv2.IBMAIX_OSProcess` and `-c cimv2.Linux_OSProcess` are the association classes whose provider is used.
- `-s 'Name=~"c175nf14"'` is the selection string against the `cimv2.IBMAIX_OperatingSystem` and `cimv2.Linux_OperatingSystem` instances (we only want objects associated with the OS instance representing the node `c175nf14`).
- `-S 'Name=~"emacs"'` is the selection string against `cimv2.IBMAIX_UnixProcess` and `cimv2.Linux_UnixProcess` objects; only those with `Name` attributes that contain the pattern `emacs` are returned.
- `cimv2.IBMAIX_OperatingSystem` and `cimv2.Linux_OperatingSystem`, which are the "source object" parameter, are one of the classes in the association.
- `Handle Parameters` are properties that the provider is asked to return. `Handle` is the PID of the process; `Parameters` is a list of arguments to the process.

The following output is displayed:

```
Resource Persistent Attributes for cimv2.IBMAIX_UnixProcess (or cimv2.Linux_UnixProcess)
resource 1:
Handle = "2781"
Parameters = {"emacs", "-u", "foo.C"}
resource 2:
Handle = "2782"
Parameters = {"emacs", "bar.C"}
```

```
resource 3:
Handle = "2783"
Parameters = {"emacs","foo_bar.C"}
resource 4:
Handle = "2784"
Parameters = {"emacs","bar_foo.C"}
resource 5:
Handle = "2785"
Parameters = {"emacs","CIMRC.C"}
resource 6:
Handle = "26994"
Parameters = {"emacs","lsassocmap.pl"}
```

lsrsrcdef Command

Purpose

Displays definition information for a resource or a resource class.

Syntax

For a *resource*...

To display the definition:

`lsrsrcdef` [`-p` *property*] [`-e`] [`-s`] [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] *resource_class* [*attr*...]

To display the persistent attribute definitions:

`lsrsrcdef -A p` [`-p` *property*] [`-e`] [`-s`] [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] *resource_class* [*attr*...]

To display the dynamic attribute definitions:

`lsrsrcdef -A d` [`-p` *property*] [`-e`] [`-s`] [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] *resource_class* [*attr*...]

For a *resource class*...

To display the definition:

`lsrsrcdef -c` [`-p` *property*] [`-e`] [`-s`] [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] *resource_class* [*attr*...]

To display the persistent attribute definitions:

`lsrsrcdef -c -A p` [`-p` *property*] [`-e`] [`-s`] [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] *resource_class* [*attr*...]

To display the dynamic attribute definitions:

`lsrsrcdef -c -A d` [`-p` *property*] [`-e`] [`-s`] [`-l` | `-i` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] *resource_class* [*attr*...]

To display a list of all of the resource class names:

`lsrsrcdef`

Description

The `lsrsrcdef` command displays the definition of a resource or a resource class or the persistent or dynamic attribute definitions of a resource or a resource class. By default:

- if no *attr* parameters are specified on the command line, this command displays the definitions for `public` attributes. To override this default, use the `-p` flag or specify the name of the attribute you want to display.

- this command does not display attribute descriptions. To display attribute definitions and descriptions, specify the `-e` flag.

Flags

-A p | d
Specifies the attribute type. You can display either persistent or dynamic attribute definitions. Use this flag with the `-c` flag to display the persistent or dynamic attribute definitions of a resource class.

p
Displays only persistent attributes

d
Displays only dynamic attributes

-c
Displays the definition of a resource class definition. To display the persistent attribute definitions for a resource class, specify this flag with the `-A p` flag. To display the dynamic attribute definitions for a resource class, specify this flag with the `-A d` flag.

-d
Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the `-D` flag to change the default delimiter.

-D *delimiter*
Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify something other than the default colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

-e
Specifies expanded format. By default, the descriptions of the definitons are not displayed. Specify this flag to display the definitions and the descriptions.

-i
Generates a template of *resource_data_input_file* that can then, after appropriate editing, be used as input to the `mkrsrc` command. The output is displayed in long (stanza) format. All required and optional attributes that can be used to define a resource are displayed. The attribute data type is displayed as the value in the *attr=value* pairs. It is suggested that when you use this flag, the output of the `lsrsrcdef` command be directed to a file. This flag overrides the `-s` and `-A d` flags.

-l
Specifies "long" format — one entry per line. This is the default display format. If the `lsrsrcdef -l` command is issued without a resource class name, this flag is ignored when the command returns the list of defined resource class names.

-p *property*
Displays attribute definitions for attributes with the specified *property*. By default, only the definitions for `public` attributes are displayed. To display all attribute definitions regardless of the property, use the `-p 0` flag.

Persistent attribute properties:

0x0001
`read_only`

0x0002
`reqd_for_define` (required)

0x0004
`inval_for_define` (not valid)

0x0008
`option_for_define` (optional)

0x0010
`selectable`

0x0020
public

Dynamic attribute properties:

0x0020
public

A decimal or hexadecimal value can be specified for the property. To request the attribute definitions for all attributes that have one or more properties, "OR" the properties of interest together and then specify the "OR"ed value with the `-p` flag. For example, to request the attribute definitions for all persistent attributes that are either `reqd_for_define` or `option_for_define`, enter:

```
lsrsrcdef -p 0x0a
```

-s
Displays the structured data definition. Specify this flag for the structured data definition to be expanded so that each element definition of the structured data attributes is displayed.

-t
Specifies table format. Each attribute is displayed in a separate column, with one resource per line.

-x
Suppresses header printing.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software-service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

resource_class
Specifies the name of the resource class with the attribute definitions you want to display.

attr
If a *resource_class* parameter is specified, zero or more attribute names can be specified. If no *attr* parameter is specified, the definition for all of the attributes for the resource are displayed. Specify individual attribute names to control which attributes are displayed and their order. Specify only persistent attribute names when the `-A p` flag is used. Specify only dynamic attribute names when the `-A d` flag is used. Attributes must be separated by spaces.

Security

The user needs write permission for the *resource_class* specified in `lsrsrcdef` to run `lsrsrcdef`. Permissions are specified in the access control list (ACL) file on the contacted system. See *RSCT: Administration Guide* for information about the ACL file and how to modify it.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To display the names of all of the resource classes defined on the system, enter:

```
lsrsrcdef
```

The output will look like this:

```
class_name
"IBM.ATMDevice"
```

```
"IBM.Association"
"IBM.AuditLog"
"IBM.AuditLogTemplate"
"IBM.Condition"
"IBM.EthernetDevice"
"IBM.EventResponse"
...
```

2. To display the resource class definitions for resource IBM.Host, enter:

```
lsrsrcdef -c IBM.Host
```

The output will look like this:

```
Resource Class Definition for: IBM.Host
resource class 1:
        class_name         = "IBM.Host"
        class_id           = 8
        properties         = {"has_rsrc_insts","mtype_subdivided"}
        display_name       = ""
        description        = ""
        locator            = "NodeList"
        class_pattr_count  = 1
        class_dattr_count  = 3
        class_action_count = 0
        pattr_count        = 6
        dattr_count        = 47
        action_count       = 0
        error_count        = 0
        rsrc_mgr_count     = 1
rsrc_mgrs 1:
        mgr_name  = "IBM.HostRM"
        first_key = 1
        last_key  = 1
```

3. To display the resource class persistent attribute definitions for resource IBM.Host, enter:

```
lsrsrcdef -c -A p -p 0 IBM.Host
```

The output will look like this:

```
Resource Class Persistent Attribute Definitions for: IBM.Host
attribute 1:
        program_name              = "Variety"
        display_name              = ""
        group_name                = ""
        properties                = {"read_only","inval_for_define"}
        description               = ""
        attribute_id              = 0
        group_id                  = 255
        data_type                 = "uint32"
        variety_list              = {{1..1}}
        variety_count             = 1
        default_value             = 0
```

4. To display the resource persistent attribute definitions and descriptions for resource IBM.Host, enter:

```
lsrsrcdef -A p -p 0 -e IBM.Host
```

The output will look like this:

```
Resource Persistent Attribute Definitions for: IBM.Host
attribute 1:
        program_name       = "Name"
        display_name       = "Name"
        group_name         = "General"
        properties         = {"reqd_for_define","public","selectable"}
        description        = "Identifies the current name of the host
                              as returned by command."
        attribute_id       = 0
        group_id           = 0
        data_type          = "char_ptr"
        variety_list       = {{1..1}}
        variety_count      = 1
        default_value      = ""
attribute 2:
```

```
        program_name       = "ResourceHandle"
        display_name       = "Resource Handle"
        group_name         = "Internal"
        properties         = {"read_only","inval_for_define","selectable"}
        description        = "A globally unique handle that identifies the host.
                              Every resource is assigned a resource handle,
                              which is used internally for identifying and
                              locating each resource.  The resource handle
                              is fixed in size and avoids the problems of
                              name space collisions across different types
                              of resources."
        attribute_id       = 1
        group_id           = 255
        data_type          = "rsrc_handle_ptr"
        variety_list       = {{1..1}}
        variety_count      = 1
        default_value      = "0x0000 0x0000 0x00000000 0x00000000 0x00000000 0x00000000"
attribute 3:
        program_name       = "Variety"
        display_name       = "Variety"
        group_name         = "Internal"
    ...
```

5. To display the public dynamic attributes for resource IBM.Host, enter:

```
lsrsrcdef -A d IBM.Host
```

The output will look like this:

```
Resource Dynamic Attribute Definitions for: IBM.Host
attribute 1:
        program_name           = "ProcRunQueue"
        display_name           = ""
        group_name             = ""
        properties             = {"public"}
        description            = ""
        attribute_id           = 1
        group_id               = 1
        data_type              = "float64"
        variable_type          = 0
        variety_list           = {{1..1}}
        variety_count          = 1
        init_value             = 0
        min_value              = 0
        max_value              = 100
        expression             = "(ProcRunQueue - ProcRunQueue@P) >= (ProcRunQueue@P * 0.5)"
        expression_description = ""
        rearm_expression       = "ProcRunQueue < 50"
        rearm_description      = ""
        PTX_name               = ""
attribute 2:
...
```

Location

/opt/rsct/bin/lsrsrcdef

lssavevg Command

Purpose

Lists or restores the contents of a volume group backup on a specified media.

Syntax

lssavevg [**-b** *blocks*] [**-f** *device*] [**-a**] [**-c**] [**-l**] [**-n**] [**-r**] [**-s**] [**-d** *path*] [**-B**] [**-D**] [**-L**] [**-V**] [*file_list*]

Description

The **lssavevg** command lists the contents of a volume group backup from tape, file, CD-ROM, or other source and can be used to restore files from a valid backup source. The **lssavevg** command also works for multi-volume backups such as multiple CDs, DVDs, USB disks, or tapes.

The **lssavevg -r** and **restorevgfiles** commands perform identical operations and must be considered interchangeable.

Flags

Item	Description
-a	Verifies the physical block size of the tape backup, as specified by the **-b** *block* flag. You might need to alter the block size if necessary to read the backup. The **-a** flag is valid only when a tape backup is used.
-b *blocks*	Specifies the number of 512-byte blocks to read in a single input operation, as defined by the *blocks* parameter. If the *blocks* parameter is not specified, the number of blocks read defaults to 100.
-B	Prints the volume group backup log to **stdout**. This flag displays the past 256 backups (roughly). The log is in alog format and is kept in **/var/adm/ras/vgbackuplog**. Each line of the log is a semicolon-separated list of the file or device name, the command that is used to make backup, date, shrink size, full size of the backup, and recommended maintenance or technology level (if any). **Note:** The shrink size is the size of the data on all file systems. The full size is total size of each file system (unused + data).
-c	Produces colon-separated output. This flag works only with the **-l** and **-L** flags.
-d *path*	Specifies the directory path to which the files are restored, as defined by the *path* parameter. If the **-d** parameter is not used, the current working directory is used. This can be a problem if the current working directory is root. We recommend writing to a temporary folder instead of to root.
-D	Produces debug output.
-f *device*	Specifies the type of device containing the backup (file, tape, CD-ROM, or other source) as defined by the *device* parameter. When **-f** is not specified, *device* will default to **/dev/rmt0**.
-l	Displays useful information about a volume group backup. This flag requires the **-f** *device* flag. This flag causes **lssavevg** to display information such as volume group, date and time backup was made, uname output from backed up system, oslevel, recommended maintenance or technology level, backup size in megabytes, and backup shrink size in megabytes. The shrink size is the size of the data on all file systems. The full size is the total size of each file system (unused + data). The **-l** flag also displays the logical volume and file system information of the backed up volume group, equivalent to running "**lsvg -l** *vgname*".
-L	Displays lpp fileset information about a **mksysb** backup only. This flag requires the **-f** *device* flag and displays the equivalent information to that produced by invoking "**lslpp -l**" on the running backed up system. This flag does not produce output about any volume group backup other than that produced by **mksysb**.
-n	Does not restore ACLs, PCLs, or extended attributes

Item	Description
-r	Specifies to restore the backup files, as defined by the *file-list* parameter. If the *file-list* parameter is not specified, then all files in the backup are restored. If the **-r** flag is not used, then executing the **lssavevg** command lists only the files in the specified backup.
-s	Specifies that the backup source is a user volume group and not rootvg.
-V	Verifies a tape backup. This flag requires the **-f** *device* flag and works for tape devices only. The **-V** flag causes **lssavevg** to verify the readability of the header of each file on the volume group backup and print any errors that occur to **stderr**.

Parameters

Item	Description
file_list	Identifies the list of files to be restored. This parameter is used only when the **-r** flag is specified. The full path of the files relative to the current directory must be specified in the space-separated list. All files in the specified directory are restored unless otherwise directed. If you are restoring all files in a directory, we recommend writing to a temporary folder instead of to root.

Examples

1. To list the contents of the system backup that is on the default device **/dev/rmt0**, enter the following command:

   ```
   lssavevg
   ```

2. To list the contents of the system backup that is on device **/dev/cd1**, enter the following command:

   ```
   lssavevg -f /dev/cd1
   ```

3. To list the contents of the system backup that is on device **/dev/cd1**, which is a user volume group that is not rootvg, enter the following command:

   ```
   lssavevg -f /dev/cd1 -s
   ```

4. To restore **/etc/filesystems** from the system backup that is on device **/dev/cd1**, enter the following command:

   ```
   lssavevg -f /dev/cd1 -r ./etc/filesystems
   ```

5. To restore all files in the **/myfs/test** directory of the non-rootvg backup, which is on device **/dev/cd1**, and write the restored files to **/data/myfiles**, enter the following command:

   ```
   lssavevg -f /dev/cd1 -r -s -d /data/myfiles ./myfs/test
   ```

6. To display colon-separated lpp information about a **mksysb** backup tape that is on device **/dev/rmt0**, enter the following command:

   ```
   lssavevg -Lc -f /dev/rmt0
   ```

7. To display the volume group backup log to **stdout**, enter the following command:

   ```
   lssavevg -B
   ```

8. To list volume group and general backup data about a backup that is on **/tmp/mybackup**, enter the following command:

```
lssavevg -l -f /tmp/mybackup
```

9. To verify the readability of each header on a volume group backup tape in **/dev/rmt0**, enter the following command:

```
lssavevg -V -f /dev/rmt0
```

10. To list the contents of the system backup that is on device **/dev/usbms0,** enter the following command:

```
lssavevg –f /dev/usbms0
```

Files

Item	Description
/usr/bin/lssavevg	Contains the **lssavevg** command

lssavewpar Command

Purpose

Lists the contents of a workload partition backup on a specified media.

Syntax

lssavewpar [**-b** *blocks*] [**-f** *device*] [**-a**] [**-c**] [**-D**] [**-l** | **-L** | **-M** | **-N**] [**-V**]

Description

The **lssavewpar** command lists the contents of a workload partition backup from tape, file, CD, USB flash drive, or DVD.

Flags

Item	Description
-a	Verifies the physical block size of the tape backup, as specified by the **-b** flag. You might need to alter the block size if necessary to read the backup. The **-a** flag is valid only when a tape backup is used.
-b *blocks*	Specifies the number of 512-byte blocks to read in a single input operation, as defined by the *blocks* parameter. If the *blocks* parameter is not specified, the number of blocks to read is 100, which is the default. The **-b** flag is valid only when a tape backup is used.
-c	Produces colon-separated output. Specify the **-c** flag only with the **-l** , **-L**, **-M**, and **-N** flags.
-D	Produces the output of debugging.
-f *device*	Specifies the device that contains the backup (file, tape, CD, or other sources) as defined by the *device* parameter. If you do not specify the **-f** flag, the default device name is **/dev/rmt0**.

Item	Description
-l	Displays information about a workload partition backup. Displays information including the volume group, the date and time that the backup was made, the uname output from the backed up system, the operating system level, the maintenance or technology level, the backup size in megabytes, and the backup-shrink size in megabytes. The shrink size is the size of the data on all file systems. The full size is the total size of each file system (unused and data). The **-l** flag also displays the logical volume and file system information of the backed up workload partition. You must specify the **-f** flag when you specify the **-l** flag. The **-l** flag is mutually exclusive with the **-L**, **-M**, and **-N** flags.
-L	Displays lpp file-set information about a workload partition backup only. When you specify the **-L** flag, you must also specify the **-f** device flag. The **-L** flag is mutually exclusive with the **-l**, **-M**, and **-N** flags.
-M	Lists information about any writable namefs-mounted file systems included in the backup. The information is displayed in the following form: `Mount_Device Blocks Blocks_Free Blocks_Used Mount_Point` The `Blocks` and `Blocks_Free` fields describe the number of 512-byte blocks and the free 512-byte blocks that are present in the mounted file system. The `Blocks_Used` describes the number of 512-byte blocks used in the portion of the mounted file system mounted from the WPAR. The **-M** flag is mutually exclusive with the **-l**, **-L**, and **-N** flags.
-N	Lists information about any NFS-mounted file systems included in the backup. The information is of the following form: `RemoteHost HostFilesystem Blocks Blocks_Free Blocks_Used` The `Blocks` and `Blocks_Free` describe the number of 512-byte blocks and the free 512-byte blocks in the remote file system. The `Blocks_Used` describes the number of 512-byte blocks used in the portion of the remote file system mounted from the WPAR. The **-N** flag is mutually exclusive with the **-l**, **-L**, and **-M** flags.
-V	Verifies a tape backup. You must specify the **-f** flag with the **-V** flag. The flag is valid only for tape devices. The **-V** flag verifies the readability of the header of each file on the volume group backup and prints any errors that occur to the **stderr** file.

Examples

1. To list the contents of the workload partition backup that is located on the default device **/dev/rmt0**, use the following command:

   ```
   lssavewpar
   ```

2. To list the contents of the system backup that is located on device **/dev/cd1**, use the following command:

   ```
   lssavewpar -f /dev/cd1
   ```

3. To display colon-separated lpp information about a workload partition backup tape that is located on **/dev/rmt0**, use the following command:

```
lssavewpar -Lc -f /dev/rmt0
```

4. To list volume group and general backup data about a backup located at `/tmp/mybackup`, use the following command:

```
lssavewpar -l -f /tmp/mybackup
```

5. To verify the readability of each header on a workload partition backup tape in **/dev/rmt0**, use the following command:

```
lssavewpar -V -f /dev/rmt0
```

6. To list the contents of the system backup located on device **/dev/usbms0,** use the following command:

```
lssavewpar –f /dev/usbms0
```

lssec Command

Purpose

Lists attributes in the security stanza files.

Syntax

lssec [**-c**] [**-f** *File*] [**-s** *Stanza*] [**-a** *Attribute* ...]

Description

The **lssec** command lists attributes stored in the security configuration stanza files. The following security configuration files contain attributes that you can specify with the *Attribute* parameter:

- **/etc/security/environ**
- **/etc/security/group**
- **/etc/security/audit/hosts**
- **/etc/security/lastlog**
- **/etc/security/limits**
- **/etc/security/login.cfg**
- **/usr/lib/security/mkuser.default**
- **/etc/nscontrol.conf**
- **/etc/security/passwd**
- **/etc/security/portlog**
- **/etc/security/pwdalg.cfg**
- **/etc/security/roles**
- **/etc/security/smitacl.user**
- **/etc/security/smitacl.group**
- **/etc/security/user**
- **/etc/security/user.roles**
- **/etc/security/rtc/rtcd_policy.conf**

When listing attributes in the **/etc/security/environ**, **/etc/security/lastlog**, **/etc/security/limits**, **/etc/security/passwd**, and **/etc/security/user** files, the stanza name specified by the *Stanza* parameter must be either a valid user name or `default`. When listing attributes in the **/etc/security/group** file, the stanza name specified by the *Stanza* parameter must be either a valid group name or `default`. When listing attributes in the **/usr/lib/security/mkuser.default** file, the *Stanza* parameter must be either `admin` or `user`. When listing attributes in the **/etc/security/portlog** file, the *Stanza* parameter must be a valid port name. When listing attributes in the **/etc/security/login.cfg** file, the *Stanza* parameter must be either a valid port name, a method name, or the **usw** attribute.

You cannot list the **password** attribute of the **/etc/security/passwd** file with the **lssec** command.

Only the root user or a user with PasswdAdmin authorization can list the lastupdate and flags attributes for administrative users.

Flags

Item	Description
-c	Specifies that the output should be in colon-separated format.
-f *File*	Specifies the name of the stanza file to list.
-s *Stanza*	Specifies the name of the stanza to list.
-a *Attribute*	Specifies the attribute to list.

Security

Access Control: This command grants execute access only to the root user and the security group. The command has the trusted computing base attribute and runs the **setuid** subroutine for the root user to access the security databases.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

To get the full functionality of the command, besides the **accessauths**, the role should also have the **aix.security** authorization.

On a Trusted AIX system, only users with authorization aix.mls.clear.read can list clearance attributes of other users. Only users with authorization aix.mls.tty.read can list port attributes.

Files Accessed:

Mode	File
r	**/etc/security/environ**
r	**/etc/security/group**
r	**/etc/security/audit/hosts**
r	**/etc/security/lastlog**
r	**/etc/security/limits**
r	**/etc/security/login.cfg**
r	**/usr/lib/security/mkuser.default**
r	**/etc/nscontrol.conf**
r	**/etc/security/passwd**
r	**/etc/security/portlog**
r	**/etc/security/pwdalg.cfg**

Mode	File
r	/etc/security/roles
r	/etc/security/smitacl.user
r	/etc/security/smitacl.group
r	/etc/security/user
r	/etc/security/user.roles
r	/etc/security/domains
rw	/etc/security/rtc/rtcd_policy.conf

Examples

1. To list the number of unsuccessful login attempts by the root user since the last successful login of the root user, enter:

   ```
   lssec -f /etc/security/lastlog -s root -a unsuccessful_login_count
   ```

   The system displays the result as follows:

   ```
   root unsuccessful_login_count=15
   ```

2. To list the times that logins are allowed on the **/dev/tty2** port, enter:

   ```
   lssec -f /etc/security/login.cfg -s /dev/tty2 -a logintimes
   ```

   The system displays the result as follows:

   ```
   /dev/tty0 logintimes=!january1,!july4,!december25
   ```

3. To list the default setting for the **tpath** attribute and the **ttys** attribute in colon format,
4. enter:

   ```
   lssec -c -f /etc/security/user -s default -a tpath -a ttys
   ```

   The system displays the result as follows:

   ```
   #name:tpath:ttys
   default:nosak:ALL
   ```

Files

Item	Description
/usr/bin/lssec	Specifies the path to the **lssec** command.
/etc/security/environ	Contains the environment attributes of users.
/etc/security/group	Contains extended attributes of groups.
/etc/security/audit/hosts	Contains host and processor IDs.
/etc/security/lastlog	Defines the last login attributes for users.
/etc/security/limits	Defines resource quotas and limits for each user.
/etc/security/login.cfg	Contains port configuration information.
/usr/lib/security/mkuser.default	Contains the defaults values for new users.
/etc/nscontrol.conf	Contains configuration information of some name services.

Item	Description
/etc/security/passwd	Contains password information.
/etc/security/portlog	Contains unsuccessful login attempt information for each port.
/etc/security/pwdalg.cfg	Contains configuration information for loadable password algorithms (LPA).
/etc/security/roles	Contains a list of valid roles.
/etc/security/smitacl.user	Contains user ACL definitions.
/etc/security/smitacl.group	Contains group ACL definitions.
/etc/security/user	Contains the extended attributes of users.
/etc/security/user.roles	Contains a list of roles for each user.
/etc/security/enc/LabelEncodings	Contains label definitions for the Trusted AIX system.
/etc/security/domains	Contains the valid domain definitions for the system.
/etc/security/rtc/rtcd_policy.conf	Contains configuration information for the **rtcd** daemon

lssecattr Command

Purpose

Displays the security attributes of a command, a device, a privileged file, a process or, a domain-assigned object.

Syntax

lssecattr [**-R load_module**] { **-c** | **-d** | **-p** [**-h**] [**-A**]| **-f** | **-o** } [**-C** | **-F**] [**-a** *List*] { **ALL** | *Name* [,*Name*] ... }

Description

The **lssecattr** command lists the security attributes of one or more commands, devices, or processes. The command interprets the *Name* parameter as either a command, a device, a privileged file, a process, or a domain-assigned object based on whether the **-c** (command), **-d** (device), **-f** (privileged file), **-p** (process), or **-o** (domain-assigned object) flag is specified. If the **-c** flag is specified, the *Name* parameter must include the full path to the commands. If the **-d** flag is specified, the *Name* parameter must include the full path to the devices. If the **-f** flag is specified, the *Name* parameter must include the full path to the file. If the **-p** flag is specified, the *Name* parameter must be the numeric process identifier (PID) of an active process on the system. If the **-o** flag is specified, the *Name* parameter must be the full path if it is a file or device and for port or port ranges it must be prefixed with TCP_ or UDP_. Use the **ALL** keyword to list the security attributes for all commands, devices, files, or processes. By default, the **lssecattr** command displays all of the security attributes for the specified object. To view the selected attributes, use the **-a** *List* flag.

If the system is configured to use databases from multiple domains, the privileged commands, privileged devices, and privileged files, as specified by the *Name* parameter, are searched from the domains in the order specified by the **secorder** attribute of the corresponding database stanza in the **/etc/nscontrol.conf** file. If duplicate entries exist in multiple domains, only the first entry instance is listed. Use the **-R** flag to list the objects from a specific domain.

By default, the **lssecattr** command lists the security attributes on one line. It displays the attribute information as the definitions of `Attribute=Value`, each separated by a blank space. To list the attributes in stanza format, use the **-F** flag. To list the attributes as colon-separated records, use the **-C** flag.

Flags

Item	Description
-a *List*	Lists the attributes to display. The *List* variable requires a blank space between attributes to list multiple attributes. If you specify an empty list, only the object names are displayed. The attributes that can be listed in the *List* variable are dependent on which one of the **-c**, **-d**, and **-p** flags is specified. For a list of the valid attribute names for each flag, see the **setsecattr** command.
-A	Display the list of authorizations used by a specified process. This flag can only be used with the **-p** flag.
-c	The *Name* parameter specifies the full paths to one or more commands on the system that have entries in the **/etc/security/privcmds** privileged command database.
-C	Displays the privileged security attributes in colon-separated records as follows: `#name:attribute1:attribute2: ...` `name:value1:value2: ...` `name:value1:value2: ...` The output is preceded by a comment line that has details about the attribute represented in each colon-separated field. If the **-a** flag is specified, the order of the attributes matches the order specified in the **-a** flag. If an object does not have a value for a given attribute, the field is still output but is empty. The last field in each entry is terminated by a newline character rather than a colon.
-d	The *Name* parameter specifies the full paths to one or more devices on the system that have entries in the **/etc/security/privdevs** privileged device database.
-f	The *Name* parameter specifies the full paths to one or more files on the system that have entries in the **/etc/security/privfiles** privileged files database.
-F	Displays the output in stanza format, with each stanza identified by an object name. Each pair of `Attribute=Value` is listed on a separate line: `Name:` `  attribute1=value` `  attribute2=value` `  attribute3=value`
-h	Displays the full hierarchy of privileges for the process. By default, only the highest level of privilege is listed.
-o	The *Name* parameter specifies one of the following entries in the **/etc/security/domobjs** domain-assigned object database. • the full paths to one or more devices/files on the system • the port or port ranges prefixed with TCP_ or UDP_ • the network interfaces
-p	The *Name* parameter specifies the numeric process identifiers (PID) of one or more active processes on the system. The **-p** flag cannot be listed with the **-R** flag as they are mutually exclusive.

Item	Description
-R *load_module*	Specifies the loadable module to query the *Name* entry from.

Parameters

Item	Description
ALL	For all commands, devices or processes.
Name	The object to modify. The *Name* parameter is interpreted according to which one of the **-c**, **-d**, **-p**, and **-o** flags is specified.

Security

The **lssecattr** command is a privileged command. It is owned by the root user and the security group, with mode set to 755. You must assume a role with at least one of the following authorizations to run the command successfully.

Item	Description
aix.security.cmd.list	Required to list the attributes of a command with the **-c** flag.
aix.security.device.list	Required to list the attributes of a device with the **-d** flag.
aix.security.file.list	Required to list the attributes of a file with the **-f** flag.
aix.security.proc.list	Required to list the attributes of a process with the **-p** flag.
aix.security.dobject.list	Required to list the attributes of a domain-assigned object with the **-o** flag.

File Accessed

Item	Description
File	**Mode**
/etc/security/privcmds	r
/etc/security/privdevs	r
/etc/security/privfiles	r
/etc/security/domobjs	r

Examples

1. To display the access authorization and the innate privileges of the **/usr/sbin/mount** command, enter the following command:

   ```
   lssecattr -c -a accessauths innateprivs /usr/sbin/mount
   ```

2. To display all the security attributes of the **/dev/mydev** device, enter the following command:

   ```
   lssecattr -d /dev/mydev
   ```

3. To display all the security attributes of the **/dev/mydev** device in LDAP, enter the following command:

   ```
   lssecattr -R LDAP -d /dev/mydev
   ```

4. To display the privileges for the effective and used privilege sets of two processes in a colon format, enter the following command:

```
lssecattr -p -C -a eprivs uprivs 38483,57382
```

5. To display the read authorization list of the **/etc/security/user** file, enter the following command:

```
lssecattr -f -a readauths /etc/security/user
```

6. To display the used authorizations for a process in a stanza format, enter the following command:

```
lssecattr -F -p -A 34890
```

7. To display all the domain attributes of the **/dev/dev1** device, enter the following command:

```
lssecattr -o /dev/dev1
```

8. To display all the domain attributes of the network interface **en0** device, enter the following command:

```
lssecattr -o en0
```

lssecmode Command

Purpose

Displays the current or pending security mode configuration and key types in a formatted output.

Syntax

lssecmode [**-p**] [**-d** | **-D** *delim*] [**-x**] [**-T**] [**-V**] [**-h**]

Description

The **lssecmode** command displays information about the current or pending security mode configuration and key types. This information consists of the compliance mode, public or private key type, and default symmetric key types.

Note: If no flag is specified, the current security configuration mode and key types are displayed.

Flags

Item	Description
-d	Displays the delimiter-formatted output. The default delimiter is a colon (:). You can use the **-D** flag to change the default delimiter.
-D *delim*	Specifies the delimiter to be used in the formatted output. By default, the colon (:) character is used as the delimiter in the output. You can use this flag to format the output with another delimiter that can contain one or more characters. For example, if the output data already contains colons, the default delimiter might result in confusion.
-h	Displays the command usage.
-p	Displays the pending security mode configuration and key types.
-T	Writes the command trace messages to standard output.
-V	Writes the command verbose messages to standard output.
-x	Specifies that the header information must not be displayed.

Exit status

0
Successful.

1
Missing argument error.

2
Invalid option error.

3
API error.

Examples

1. To display the current security configuration mode and key types, enter the following command:

```
# lssecmode
```

An output similar to the following example is displayed:

```
Current Security Mode Configuration
Compliance Mode : none
Asymmetric Key Type : rsa512
Symmetric Key Type : default
```

2. To display the pending security configuration mode and key types, enter the following command:

```
# lssecmode -p
```

If there is no pending security configuration mode and if the staging file is not present, the following output is displayed:

```
2650-384 There are no pending configuration available
```

3. To format the output with a delimiter **::**, enter the following command:

```
# lssecmode -D "::"
```

An output similar to the following example is displayed:

```
Current Security Mode Configuration
Compliance Mode :: none
Asymmetric Key Type :: rsa512
Symmetric Key Type :: default
```

4. To display the pending security configuration mode and key types with command verbose messages and without header information, enter the following command:

```
# lssecmode -p -x -V -D "::" -T
```

An output similar to the following example is displayed:

```
Invoked with parameters: -p -x -V -D :: -T
Invoking lssecmode to get the pending security mode and key types....
No header information required..
Compliance Mode :: none
Asymmetric Key Type :: rsa512
Symmetric Key Type :: default
Checking lssecmode log file size and backup if necessary....
No log file exist. No backup is needed
```

Location

/opt/rsct/bin/lssecmode
Contains the **lssecmode** command.

lssensor Command

Purpose

Displays information about sensors and microsensors that are defined to the resource monitoring and control (RMC) subsystem.

Syntax

lssensor [**-m**] [**-a** | **-n** *host1*[**,***host2*...] | **-N** { *node_file* "–" }] [**-l** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-v** | **-V**] [**-A** | *sensor_name1* [*sensor_name2*...]]

Description

The `lssensor` command displays the attributes of one or more sensors. If you do not specify any *name* parameters, the `lssensor` command lists the names of all of the sensors. Use the `-A` flag to list all of the sensors and all of their attributes and values. Use the **-m** flag to display information about microsensors.

The **lssensor** command displays values for attributes that you can set using a sensor command or a microsensor module, if the attributes are monitored. If the attributes are not monitored, **lssensor** does not display their values. A sensor command is a command or script that the sensor resource manager runs to set and update a sensor's attribute values. A microsensor module is a loadable module that the microsensor resource manager runs to set and update a microsensor's attribute values.

Use the **-l**, **-t**, **-d**, or **-D** flags to display the output in long format, table format, or delimiter format. The **-x** flag omits headings when any of these flags are used.

The **lssensor** command runs on any node. If you want **lssensor** to run on all of the nodes in a domain, use the **-a** flag. If you want **lssensor** to run on a subset of nodes in a domain, use the **-n** flag. Instead of specifying multiple node names using the **-n** flag, you can use the **-N** *node_file* flag to indicate that the node names are in a file. Use **-N "–"** to read the node names from standard input.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

The `lssensor` command lists the following information about defined sensors:

Field	Description
Name	The name of the sensor.
Command	The command that is run to update the sensor attributes
ConfigChanged	Information about changes to access or to persistent attributes
ControlFlags	Indicates whether any special handling is required for this sensor
Description	This field is not used
ErrorExitValue	Indicates how the exit value is interpreted by the sensor resource manager
ErrorMessage	This field is not used
ExitValue	The exit code from the command that is running
Float32	The type `float32` attribute for this sensor resource
Float64	The type `float64` attribute for this sensor resource
Int32	The type `int32` attribute for this sensor resource

Field	Description
Int64	The type int64 attribute for this sensor resource
MonitorStatus	This attribute is set to 1 when certain sensor attributes are being monitored
NodeNameList	The name of the node where the sensor resource is defined
RefreshInterval	The interval (in seconds) during which the sensor attribute values are updated when the sensor command is run
SavedData	An output string from the sensor command
SD	Contains all dynamic resource attributes except ConfigChanged, Quantum, and ExitValue as its elements
String	The type string attribute for this sensor resource
TimeCommandRun	Indicates the date and time that the sensor command was run
Uint32	The type uint32 attribute for this sensor resource
Uint64	The type uint64 attribute for this sensor resource
UserName	The user ID that is used when run the sensor command is run

The **lssensor** command displays the following information about defined microsensors:

Field	Description
Name	The name of the microsensor.
ActivePeerDomain	The peer domain for which information is being displayed.
Arguments	The arguments for this microsensor resource.
ConfigChanged	Information about changes to persistent attributes or to access.
CustomDynamicAttributes	The custom dynamic attributes for this microsensor resource.
Description	Information about the microsensor and what it monitors.
Float32	The type float32 attribute for this microsensor resource.
Float32Array	The type float32 array attribute for this microsensor resource.
Float64	The type float64 attribute for this microsensor resource.
Float64Array	The type float64 array attribute for this microsensor resource.
Int32	The type int32 attribute for this microsensor resource.
Int32Array	The type int32 array attribute for this microsensor resource.
Int64	The type int64 attribute for this microsensor resource.
Int64Array	The type int64 array attribute for this microsensor resource.
LastQueryRC	The return code from the microsensor module from the last time the microsensor was called for an attribute of the microsensor resource.
LastQueryTime	The time of LastQueryRC.
ModuleName	The path name to the loadable microsensor module.
MonitorStatus	This attribute is set to 1 when any of the other microsensor attributes is being monitored.
NodeNameList	The name of the node where this microsensor is defined.

Field	Description
RefreshInterval	The interval (in seconds) during which the microsensor attribute values are updated when the microsensor callback is called.
String	The type string attribute for this microsensor resource.
StringArray	The type string array attribute for this microsensor resource.
UInt32	The type uint32 attribute for this microsensor resource.
UInt32Array	The type uint32 array attribute for this microsensor resource.
UInt64	The type uint64 attribute for this microsensor resource.
UInt64Array	The type uint64 array attribute for this microsensor resource.

Flags

-a
Lists sensors that match the specified name on all nodes in the domain.
The CT_MANAGEMENT_SCOPE environment variable determines the cluster scope. If CT_MANAGEMENT_SCOPE is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management domain and a peer domain exist, `lssensor` -a with CT_MANAGEMENT_SCOPE not set will run in the management domain. In this case, to run in the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-A
Displays all of the sensors with their attributes and values.

–d
Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the **-D** flag if you want to change the default delimiter.

–D *delimiter*
Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify something other than the default colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

–l
Specifies that the information be displayed in "long" format. Each attribute is displayed on a separate line.

-m
Specifies that information about microsensors will be displayed.

-n *host1*[,*host2*...]
Specifies the node from which the sensor should be listed. By default, the sensor is listed from the local node. This flag is only appropriate in a management domain or a peer domain.

-N {*node_file* | "-"}
Specifies that node names are read from a file or from standard input. Use **-N** *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (#) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use **-N "–"** in a management domain or a peer domain to read the node names from standard input.

–t
Specifies table format. Each attribute is displayed in a separate column, with one sensor resource per line.

-x
Suppresses header printing when **-l**, **-t**, **-d**, or **-D** is specified.

-h
Writes the command's usage statement to standard output.

-v | **-V**
Writes the command's verbose messages to standard output.

Parameters

***sensor_name1* [*sensor_name2*...]**
Specifies the names of one or more sensors to display.

Security

To display sensor information using this command, you need read permission for the **IBM.Sensor** resource class. To display microsensor information using this command, you need read permission for the **IBM.MicroSensor** resource class. Permissions are specified in the access control list (ACL) file on the contacted system. See the *Administering RSCT* guide for details on the ACL file and how to modify it.

Exit Status

0
The command has run successfully.

1
An incorrect combination of flags and parameters has been entered.

6
No sensor resources were found.

n
Based on other errors that can be returned by the RMC subsystem.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If this environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled.

The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Examples

1. To list the names of all of the sensors, enter:

```
lssensor
```

The output will look like this:

```
sensor1
sensor2
sensor3
```

2. To list the names and attributes of all sensors, enter:

```
lssensor -A
```

The output will look like this:

```
 Name = sensor1
 ActivePeerDomain =
 Command = /usr/local/bin/sensorcmd1
 ConfigChanged = 0
 ControlFlags = 1
 Description =
 ErrorExitValue = 1
 ExitValue = 0
 Float32 = 1.06381e+06
 Float64 = 1.06381e+06
 Int32 = 1063814
 Int64 = 1063814
 NodeNameList = {somenode.pok.ibm.com}
 RefreshInterval = 60
 SavedData = Last SavedData
 SD = [string from sensor1,1063814,1063814,1063814,1063814,1.06381e+06,1.06381e+06]
 String = string from sensor1
 Uint32 = 1063814
 Uint64 = 1063814
 UserName = root
-------------------
 Name = CFMRootModTime
 ActivePeerDomain =
 Command = /opt/csm/csmbin/mtime/cfmroot
 ConfigChanged = 0
 ControlFlags = 0
 Description =
 ErrorExitValue = 1
 ExitValue = 0
 Float32 = 0
 Float64 = 0
 Int32 = 0
 Int64 = 0
 NodeNameList = {somenode.pok.ibm.com}
 RefreshInterval = 60
 SavedData =
 SD = [,0,0,0,0,0,0]
 String =
 Uint32 = 0
 Uint64 = 0
 UserName = root
```

```
-------------------
 Name = ErrorLogSensor
 ActivePeerDomain =
 Command = /opt/csm/csmbin/monerrorlog
 ConfigChanged = 0
 ControlFlags = 0
 Description =
 ErrorExitValue = 1
 ExitValue = 0
 Float32 = 0
 Float64 = 0
 Int32 = 0
 Int64 = 0
 NodeNameList = {somenode.pok.ibm.com}
 RefreshInterval = 60
 SavedData =
 SD = [,0,0,0,0,0,0]
 String =
 Uint32 = 0
 Uint64 = 0
 UserName = root
-------------------
.
.
.
```

3. To list the attributes of sensor2, enter:

```
lssensor sensor2
```

   The output will look like this:

```
Name = sensor2
Command = /usr/local/bin/sensorcmd2
ConfigChanged = 0
ControlFlags = 0
Description =
ErrorExitValue = 1
ExitValue = 127
Float32 = 0
Float64 = 0
Int32 = 0
Int64 = 0
NodeNameList = {somenode.pok.ibm.com}
RefreshInterval = 60
SavedData =
SD = [,0,0,0,0,0,0]
String =
Uint32 = 0
Uint64 = 0
UserName = root
```

4. To list all of the sensors' information using delimited output, enter:

```
lssensor -dA
```

   The output will look like this:

```
Displaying sensor information:
Name:ActivePeerDomain:Command:ConfigChanged:ControlFlags:Description:ErrorExitValue:ErrorMessage:ExitValue
:
Float32:Float64:Int32:Int64:MonitorStatus:NodeNameList:RefreshInterval:SD:SavedData:
String:TimeCommandRun:Uint32:Uint64:UserName:
JoeExample:JoeDomain:cat /etc/motd:0:0::1::0:
::::0:{node1.myhost.com}:60:[,0,0,0,0,0,0]::
:Fri Feb  6 19:00:00 2009:::root:
JoeSample:JoeDomain:/opt/rsct/install/bin/ctversion:0:0::1::0:
::::0:{node1.myhost.com}:60:[,0,0,0,0,0,0]::
:Fri Feb  6 19:00:00 2009:::root:
JoeSens:JoeDomain:/tmp/sensor/numusers:0:1::1::0:
::::0:{node1.myhost.com}:0:[,2,0,0,0,0,0]::
:Tue Mar  3 10:27:19 2009:::root:
```

5. To list the names of all of the sensors on the nodes that are listed in the **/u/joe/common_nodes** file, enter:

```
lssensor -N /u/joe/common_nodes
```

where /u/joe/common_nodes contains:

```
# common node file
#
node1.myhost.com    main node
node2.myhost.com    backup node
```

The output will look like this:

```
sensor1
sensor2
sensor3
```

6. To list the names of all of the microsensors, enter:

```
lssensor -m
```

The output will look like this:

```
IBM.MSensor1
IBM.MSensor2
IBM.MSensor3
```

7. To list the attributes of the microsensor **IBM.MSensor2**, enter:

```
lssensor -m IBM.MSensor2
```

The output will look like this:

```
Name = IBM.MSensor2
ActivePeerDomain =
Arguments = all
ConfigChanged = 0
CustomDynamicAttributes = {[CDA1,19,1,3,0,1],[CDA2,20,2,2,0,1],[CDA3,21,3,2,0,1]
Description =
Float32 =
Float32Array =
Float64 =
Float64Array =
Int32 = 52
Int32Array = {36, 45, 2, 73}
Int64 =
Int64Array =
LastQueryRC = 0
LastQueryTime = Tue Mar 31 18:00:00 2009
ModuleName = /usr/slib/msensors/sensor2
MonitorStatus = 0
NodeNameList = {node2.gumby.com}
RefreshInterval = 600
String =
StringArray =
UInt32 =
UInt32Array =
UInt64 =
UInt64Array =
```

Location

/opt/rsct/bin/lssensor

lsslot Command

Purpose

Displays dynamically reconfigurable slots, such as hot plug slots, and their characteristics.

Syntax

lsslot **-c** *ConnectorType* [**-a** | **-o** | **-l** *DeviceName* | **-s** *Slot*] [**-F** *Delimiter*]

Description

The **lsslot** command displays all the specified hot plug slots and their characteristics. Hot plug slots are the plug-in points for connecting entities that can be added and removed from the system without turning the system power off or rebooting the operating system. The **-c** flag is required. It specifies the type of hot plug connector, for example, pci for hot pluggable PCI adapters. You can display only the empty, that is, available, hot plug slots with the **-a** flag, the occupied slots with the **-o** flag, or a specific slot by using the **-s** flag. The **-l** flag can be used to locate the slot associated with specified *DeviceName*, as listed by the **lsdev** command.

The **lsslot** command is used to list the connectors which are connection points for either physical entities like PCI adapters or logical entities like logical slots or logical host-Ethernet adapter ports. The command can list the following types of connectors:

- pci: a physical connector
- slot: a logical connector
- phb: a logical connector
- port: a logical connector

The **-a** and the **-o** flags will be ignored for the logical connectors. The **lsslot** command in the case of the logical connectors displays the logical entities that are currently assigned to the partition, depending upon the connector type specified. When there are multiple slots under a PHB, a logical slot entity can be associated with a logical slot connector. Otherwise, it can be associated with a logical PHB connector. You can run the **lsslot -c** *slot* command and the **lsslot -c** *phb* command to view all logical slot entities.

The output of the **lsslot** command is dependent on the *ConnectorType* and the platform on which the command is executed. The characteristics of a slot may include the following:

- Slot name or identification
- Connector type or slot description, for example, a PCI hot plug slot
- Connected device name(s), for example, scsi0, ent0

When the PHBs are listed using the **lsslot** command, the Device(s) Connected column will display the ODM name of the PHB followed by the ODM names of the devices corresponding to the logical slots underneath the PHB, with all the ODM devices associated with each logical slot displayed on each separate line under the ODM name of the PHB. In case there is no ODM name for the PHB, a blank line will be displayed.

Flags

Item	Description
-a	Displays available hot plug slots and their characteristics. Available slots are those slots that do not have a hot plug device connected. This flag is ignored for connector types of slot and phb.
-c *ConnectorType*	Displays the slots of the specified *ConnectorType*. *ConnectorType* identifies the type of connector. For example, the *ConnectorType* for a hot plug PCI slot is pci, for logical slots, it is slot and for PHBs, it is phb. This flag is required.
-F *Delimiter*	Specifies a single character to delimit the output. The heading is not displayed and the columns are delimited by the *Delimiter* character.
-l *DeviceName*	Displays the characteristics of the slot to which *DeviceName* is associated. The *DeviceName* is the logical device name of the device connected to the slot, as listed by the **lsdev** command.
-o	Displays the characteristics of the occupied slots. Occupied slots have a hot plug device connected. This flag is ignored for connector types of slot and phb.

Item	Description
-s *Slot*	Displays characteristics for the specified *Slot*. The format of *Slot* is platform/connector_type dependent.

Examples

1. To list the available PCI hot plug slots, enter:

   ```
   lsslot -c pci -a
   ```

   The system displays a message similar to the following:

Item	Description	
Slot name	**Description**	**Device(s) Connected**
U0.4-P1-I1	PCI 64 bit, 66MHz, 3.3 volt slot	empty
U0.4-P1-I2	PCI 64 bit, 66MHz, 3.3 volt slot	empty
U0.4-P1-I3	PCI 64 bit, 66MHz, 3.3 volt slot	empty

2. To list the PCI hot plug slot associated with a scsi adapter named scsi1, enter:

   ```
   lsslot -c pci -l scsi1
   ```

   The system displays a message similar to the following:

Item	Description	
Slot name	**Description**	**Device(s) Connected**
U0.4-P1-I1	PCI 64 bit, 33MHz, 5 volt slot	scsi1

3. To list all the PCI hot plug slots, enter:

   ```
   lsslot -c pci
   ```

   The system displays a message similar to the following:

Item	Description	
Slot name	**Description**	**Device(s) Connected**
U0.4-P1-I1	PCI 64 bit, 33MHz, 3.3 volt slot	empty
U0.4-P1-I2	PCI 64 bit, 33MHz, 3.3 volt slot	scsi0
U0.4-P1-I3	PCI 64 bit, 33MHz, 3.3 volt slot	unknown
U0.4-P1-I5	PCI 64 bit, 33MHz, 3.3 volt slot	empty

   Slots that have *unknown* in the Device(s) Connected column have a device connected to the slot, but the device isn't in the ODM customized device (CuDv) database. This can be due to the device having been newly added but not configured yet, deleted with the **rmdev -d** command, or the system may not be installed with the software packages associated with the device.

4. To list all the PCI Host Bridges that are assigned to the partition, enter:

   ```
   lsslot -c phb
   ```

   This displays output similar to the following:

   ```
   PHB Name         Description                 Device(s) Connected
   PHB 1            Logical PCI Host Bridge     pci0
                                                pci2 scsi1
   PHB 2            Logical PCI Host Bridge     pci1
                                                pci3
                                                pci4 scsi2
   ```

5. In case the PCI Host Bridge is assigned to the partition but has no ODM data, column will show blank as shown in this example. For example, when you enter:

```
lsslot -c phb
```

The output will look similar to the following:

```
PHB Name         Description              Device(s) Connected
PHB 4            Logical PCI Host Bridge
PHB 5            Logical PCI Host Bridge
```

6. To list all the logical host-Ethernet adapter-port devices that are assigned to the partition, enter:

```
lsslot  -c port
```

Item	Description	
LHEA port name	**Description**	**Device(s) Connected**
Port 1	Logical HEA Port	ent4
Port 2	Logical HEA Port	ent7

When the logical host-Ethernet adapter port is assigned to the partition but has no ODM data, the column shows Unknown as shown in the example:

```
lsslot  -c port
```

Item	Description	
LHEA port name	**Description**	**Device(s) Connected**
Port 4	Logical HEA Port	Unknown
Port 5	Logical HEA Port	Unknown

Files

Item	Description
/usr/sbin/lsslot	Contains the **lsslot** command.

lssmbcred Command

Purpose

Lists the credential entries that are stored in the /etc/smbcred file for the specified server and username.

Syntax

```
lssmbcred [-s server_name] [-u user_name]
```

Description

When you run the **lssmbcred** command without any flags, the **lssmbcred** command lists all of the credential entries that have password for the username in the /etc/smbcred file. If you specify the **-s** or **-u** flag, the **lssmbcred** command displays credentials that match the specified server name or the specified username.

Flags

-s *server_name*
Specifies the remote host, which is the Server Message Block (SMB) server for which the credentials are displayed if a matching credential entry is found in the `/etc/smbcred` file.

-u *user_name*
Specifies the username for which the credentials are displayed if a matching credential entry is found in the `/etc/smbcred` file.

Exit status

0
The command completed successfully.

>0
An error occurred.

Example

To list all the credential entries on the `xxx.in.ibm.com` server, enter the following command:

```
lssmbcred -s xxx.in.ibm.com
```

An output similar to the following example is displayed:

```
server: xxx.in.ibm.com user: user1
server: xxx.in.ibm.com user: user2
```

Location

`/usr/sbin/lssmbcred`

Files

/etc/smbcred
Stores the credentials of the SMB client file system.

lssrad Command

Purpose

Displays the system SRADID (Scheduler Resource Allocation Domain Identifier) hierarchy and topology.

Syntax

lssrad [**-v**] { **-s** *SRADID* | **-a** }

Description

The **lssrad** command displays information related to SRADIDs, such as the processor and memory associated with the SRAD (Scheduler Resource Allocation Domain) and REF1 system detail level, where REF1 is the first hardware provided reference point that identifies sets of resources that are near each other. This command also displays the SRADID hierarchy and topology.

Flags

Item	Description
-a	Displays all SRADs in the system.
-s *SRADID*	Displays the specified SRADID.
-v	Displays resources in the SRAD, along with the REF1 System Detail Level that the SRAD belongs to, in verbose mode.

Examples

1. To display the list of all SRADs in the system, enter:

```
# lssrad -a
```

2. To verify that a specific SRAD exists, enter:

```
# lssrad -s 0
SRAD
0
```

```
# lssrad -s 5
SRAD 5:  No such SRAD
```

3. To display the topology of a specific SRAD, enter:

```
# lssrad -v -s 5
```

4. To display the SRADID hierarchy and topology, enter:

```
# lssrad -v -a
```

Files

Item	Description
/usr/sbin/lssrad	Contains the **lssrad** command

lssrc Command

Purpose

Gets the status of a subsystem, a group of subsystems, or a subserver.

Syntax

To Get All Status

lssrc [**-h** *Host*] **-a**

To Get Group Status

lssrc [**-h** *Host*] **-g** *GroupName*

To Get Subsystem Status

lssrc [**-h** *Host*] [**-l**] **-s** *Subsystem*

To Get Status by PID

lssrc [**-h** *Host*] [**-l**] **-p** *SubsystemPID*

To Get Subserver Status

lssrc [**-h** *Host*] [**-l**] **-t** *Type* [**-p** *SubsystemPID*] [**-o** *Object*] [**-P** *SubserverPID*]

To Get Subsystem Status in SMIT Format

lssrc -S [**-s** *Subsystem* | **-d**]

To Get Subserver Status in SMIT Format

lssrc -T [**-t** *Type*]

To Get Notify in SMIT Format

lssrc -N [**-n** *NotifyName*]

Description

The **lssrc** command sends a request to the System Resource Controller to get status on a subsystem, a group of subsystems, or all subsystems. The **lssrc** command sends a subsystem request packet to the daemon to be forwarded to the subsystem for a subserver status or a long subsystem status.

You can choose whether to request a short or long status for a subserver. When the **-l** flag is absent, the status request is assumed to be a short status. A short status of a subsystem, group of subsystems, or all subsystems is handled by the System Resource Controller.

When the **-l** flag is present for a subsystem, a status request is taken to the subsystem and the subsystem sends the status back. The **-l** flag is supported only for those subsystems not using signals as their communication method. For either a long or short status of a subserver, the subsystem is sent a status request packet, and the subsystem sends the status back.

The **lssrc** command output can sometimes show two entries for a particular daemon. One instance will be active and another instance will be inoperative. This can happen if the subsystem is modified (using the **mkssys** command or **chssys** command) without stopping the subsystem. The original subsystem will remain active and the modified instance will be inoperative until the subsystem is stopped and started again.

Flags

Item	Description
-a	Lists the current status of all defined subsystem.
-d	Specifies that the default record is printed.
-g *GroupName*	Specifies a group of subsystems to get status for. The command is unsuccessful if the *GroupName* variable is not contained in the subsystem object class.
-h *Host*	Specifies the foreign host on which this status action is requested. The local user must be running as "root". The remote system must be configured to accept remote System Resource Controller requests. That is, the **srcmstr** daemon (see **/etc/inittab**) must be started with the **-r** flag and the **/etc/hosts.equiv** or **.rhosts** file must be configured to allow remote requests.
-l	Requests that a subsystem send its current status in long form. Long status requires that a status request be sent to the subsystem; it is the responsibility of the subsystem to return the status.
-n *NotifyName*	Specifies the name of a notify method.
-N	Specifies that the Object Data Manager (ODM) records are output in SMIT format for the notify object class.
-o *Object*	Specifies that a subserver *Object* variable is passed to the subsystem as a character string.

Item	Description
-p *SubsystemPID*	Specifies a particular instance of the *SubsystemPID* variable to get status for, or a particular instance of the subsystem to which the status subserver request is to be taken.
-P *SubserverPID*	Specifies that a *SubserverPID* variable is to be passed to the subsystem as a character string.
-s *Subsystem*	Specifies a subsystem to get status for. The *Subsystem* variable can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the *Subsystem* variable is not contained in the subsystem object class.
-S	Specifies that the ODM records are output in SMIT format for the subsystem object class.
-t *Type*	Requests that a subsystem send the current status of a subserver. The command is unsuccessful if the subserver *Type* variable is not contained in the subserver object class.
-T	Specifies that the ODM records are output in SMIT format for the subserver object class.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Exit status

0
The command ran successfully.

1
The command failed.

Examples

1. To get the status of all subsystems on the local machine, enter:

   ```
   lssrc -a
   ```

   This gets the status of all subsystems known on the local machine.
2. To get the status of all subsystems on a foreign host, enter:

   ```
   lssrc -h zork -a
   ```

   This gets the status of all subsystems known on the `zork` machine.
3. To get the status of the srctest subsystem, enter:

   ```
   lssrc -s srctest
   ```

   This gets the status of all instances of the `srctest` subsystem on the local machine.
4. To get the status of the subsystem by PID, enter:

   ```
   lssrc -p 1234
   ```

   This gets the status of the subsystem with the subsystem PID of 1234 on the local machine.
5. To get the status of the `tcpip` subsystem group, enter:

```
lssrc  -g tcpip
```

This gets the status of all instances of subsystems in the `tcpip` group on the local machine.

6. To get the status of the `tester` subserver, enter:

```
lssrc  -t tester  -p 1234
```

This gets the status of `tester` subserver that belongs to the `srctest` subsystem with the subsystem PID of 1234 on the local machine.

7. To get the status of the subsystem by PID, enter:

```
lssrc  -l  -p 1234
```

This gets the long status of the subsystem with the PID of 1234.

Files

Item	Description
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration Object Class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration Object Class.
/etc/objrepos/SRCnotify	Specifies the SRC Notify Configuration Object Class.
/etc/services	Defines the sockets and protocols used for Internet services.
/dev/SRC	Specifies the AF_UNIX socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.

lsts Command

Purpose

Lists thin server information. This command can be run on a thin server.

Syntax

lsts [[**-l{1|2|3}**] . . .] [**-v**] [*ThinServer*]

Description

The `lsts` command lists information pertaining to a thin server. The level of information to be listed depends on the numeric value specified by the `-l` flag, with a level ranging from 1 - 3 (3 being the most detailed). If a level is not specified, a default of level 1 information is displayed. This command can be run on both a NIM master or a thin server. When run on a NIM master and no argument is provided, the `lsts` command lists all thin servers in the environment controlled by the caller of the `lsts` command.

Flags

Item	Description
`-l{1|2|3}`	Specifies the level of output. **1** This level displays very limited information related to a thin server. The information listed shows only a brief summary of the thin server, such as the common image it is using. **2** This level displays more than just basic information related to a thin server. The level includes information pertaining to the software content of the thin server. **3** This level displays more in-depth information related to a thin server. The level includes information pertaining to the installation log of the thin server.
`-v`	Enables verbose debug output when the `lsts` command runs.

Parameters

Item	Description
Thinserver	Specifies the thin server where the command lists information about the client.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

Access Control: You must have root authority to run the `lsts` command.

Examples

1. To list brief status information for a thin server named `lobo`, enter:

   ```
   lsts lobo
   ```

   Information similar to the following is displayed:

   ```
   Lobo:
           class          = machines
           type           = diskless
           platform       = chrp
           netboot_kernel = 64
           if1            = master_net jsblade04 0 ent1
           cable_type1    = bnc
           Cstate         = diskless or dataless boot is enabled
           prev_state     = in the process of booting
           Mstate         = currently running
           boot           = boot
           dump           = dump_res
           paging         = paging_res
           root           = root_res
           spot           = 530spot_res
           cpuid          = 00012A80D000
   ```

```
        control        = master
        Cstate_result  = success
```

2. To list software content for a thin server named lobo, enter:

```
lsts -l2 lobo
```

Software content similar to the following is displayed from the common image:

```
Fileset                   Level  State  Type  Description
(Uninstaller)
----------------------------------------------------------
bos.64bit                5.2.0.75  C     F    Base Operating System 64 bit Runtime
bos.diag.com             5.2.0.75  C     F    Common Hardware Diagnostics
bos.diag.rte             5.2.0.75  C     F    Hardware Diagnostics
.
.
.
```

3. To list both software content and status information for a thin server named lobo, enter:

```
lsts -l1 -l2 lobo
```

Location

/usr/sbin/lsts

Files

Item	Description
/etc/niminfo	Contains variables used by NIM.

lstun Command

Purpose

Lists tunnel definition(s).

Syntax

lstun [**-v 4**|**6**] [**-t** *tid_list*] [**-p manual**] [**-a**]

Description

Use the **lstun** command to list the tunnel definition(s) and their current status. This command can either list the tunnels in the tunnel database or in the active system.

Flags

Item	Description
-v	This flag specifies the IP version. For listing IP version 4 tunnel only, use the value of **4**. For listing IP version 6 tunnel only, use the value of **6**. If this flag is not used, both the version 4 and version 6 tunnels will be listed.
-t	Only list the tunnel definition and its current status for the tunnel whose tunnel ID is in **tid_list**. If this flag is not used, all the tunnel definitions and their current status will be listed.

Item	Description
-p	Selects the type of the tunnel to be listed. Using the **-p** flag with the value of **manual** lists **manual** tunnels only. The **-p** flag is for listing tunnel definitions in the tunnel database only and thus is mutually exclusive with the **-a** flag.
-a	Lists the tunnels active in the IP Security subsystem.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

lstxattr Command

Purpose

Lists the security attributes of a file, directory, process, or interprocess communication (IPC).

Syntax

lstxattr { **-f** | **-m** | **-p** | **-q** | **-s** } [**-C** | **-F**] [**-a** *List*] { *Name* [,*Name*] ... }

Description

The **lstxattr** command lists Trusted AIX security attributes of the file, process, shared memory, message queue or semaphore that is specified by the *Name* parameter. The command interprets the *Name* parameter as either a file, process, shared memory, message queue or semaphore based on whether the **-f** (file), **-p** (process), **-m** (shared memory), **-q** (message queue) or **-s** (semaphore) flag is specified.

By default, the **lstxattr** command displays all the security attributes for the specified object. To view the selected attributes, use the **-a** *List* flag.

By default, the security attributes are listed on one line. The command displays the attribute information as *Attribute = Value* definitions, each separated by a blank space. To list the attributes in stanza format, use the **-F** flag. To list the attributes as colon-separated records, use the **-C** flag.

Flags

Item	Description
-a *List*	Lists the attributes to display. The *List* variable requires a blank space between attributes to list multiple attributes. If you specify an empty list, the command displays only the object names. The attributes that can be listed in the *List* variable are dependent on which one of the **-f**, **-p**, **-m**, **-q** or **-s** flags that you specified. Use the following file security attributes for the **-f** flag: **sl** Sensitivity Label. If specified for a non-regular file, the command lists both the maximum and minimum sensitivity labels. **maxsl** Maximum Sensitivity Label. If specified for regular files, the command lists the **sl** value. **minsl** Minimum Sensitivity Label. If specified for regular files, the command lists the **sl** value. **tl** Integrity Label. **secflags** Trusted AIX file security flags. Use the following process security attributes for the **-p** flag: **effsl** Effective Sensitivity Label. **maxcl** Maximum Sensitivity Clearance Label. **mincl** Minimum Sensitivity Clearance Label. **efftl** Effective Integrity Label. **maxtl** Maximum Integrity Label. **mintl** Minimum Integrity Label. Use the following security attributes for the **-q, -m,** and **-s** flags: **sl** Sensitivity Label. **tl** Integrity Label.
-C	Displays the privileged security attributes in colon-separated records in the following way: `#name:attribute1:attribute2: ...` `name:value1:value2: ...` `name:value1:value2: ...` The output is preceded by a comment line that lists details about the attribute represented in each colon-separated field. If you specify the **-a** flag, the order of the attributes matches the order specified in the **-a** flag. If an object does not have a value for a given attribute, the field is still displayed but is empty. The last field in each entry is ended by a newline character rather than a colon.

Item	Description
-f	Lists the security attributes of a file. The *Name* parameter specifies the path to this file on the system.
-F	Displays the output in stanza format, with each stanza identified by a object name. Each *Attribute = Value* pair is listed on a separate line: `Name:` `attribute1=value` `attribute2=value` `attribute3=value`
-m	Lists the security attributes of a shared memory. The *Name* parameter specifies the numeric shared memory identifier on the system.
-p	Lists the security attributes of a process. The *Name* parameter specifies the numeric process identifier (PID) of an active process on the system.
-q	Lists the security attributes of a message queue. The *Name* parameter specifies the numeric message queue identifier on the system.
-s	Lists the security attributes of a semaphore. The *Name* parameter specifies the numeric semaphore identifier on the system.

Parameters

Item	Description
Name	The object to list. The *Name* parameter is interpreted according to which one of the **-f**, **-p**, **-m**, **-q** or **-s** flags that you specified.

Security

The **lstxattr** command is a privileged command. It is owned by the root user and the security group, with the mode set to 755.

Restriction: The binary labels of the objects are interpreted as human-readable format and depend on the values in the **/etc/security/enc/LabelEncodings** file. If the conversion fails, you must have the following authorizations:

- **aix.mls.stat** authorizations for listing the binary labels of files and IPC objects
- **aix.mls.proc** authorizations for listing the binary labels of processes

Files Accessed:

Item	Description
Mode	**File**
r	**/etc/security/enc/LabelEncodings**

Examples

1. To list all the attributes of the **regfile** file, enter the following command:

   ```
   lstxattr –f regfile
   ```

2. To list the maximum sensitivity, minimum sensitivity and integrity labels of the **dirname** directory, enter the following command:

   ```
   lstxattr –f -a maxsl minsl tl dirname
   ```

3. To list the labels of a message-queue IPC object with "0" as the message queue ID, enter the following command:

```
lstxattr -q -a sl tl 0
```

4. To list the labels of a shared-memory IPC object with "3145728" as the shared memory ID, enter the following command:

```
lstxattr -m -a sl tl 3145728
```

lsuser Command

Purpose

Displays user account attributes.

Syntax

lsuser [**-R** *load_module*] [**-c** | **-C** | **-f**] [**-a** *List*] { **ALL** | *Name* [,*Name*] ... }

Description

The **lsuser** command displays the user account attributes. You can use this command to list all attributes of all the system users or all the attributes of specific users. Since there is no default parameter, you must enter the **ALL** keyword to see the attributes of all the users. By default, the **lsuser** command displays all user attributes. To view selected attributes, use the **-a** *List* flag. If one or more attributes cannot be read, the **lsuser** command lists as much information as possible, but does not display empty attributes.

Note: If the *domainlessgroups* attribute is set in the /etc/secvars.cfg file, the **lsuser** command lists the merged group from the LDAP module and the LOCAL module, if present.

By default, the **lsuser** command lists each user's attributes on one line. It displays attribute information as *Attribute*=*Value* definitions, each separated by a blank space. To list the user attributes in stanza format, use the **-f** flag. To list the information as colon-separated records, use the **-c** or **-C** flag.

You can use the System Management Interface Tool (SMIT) **smit lsusers** fast path to run this command.

Flags

Item	Description
-a *List*	Lists the attributes to display. The *List* variable can include any attribute that is defined in the **chuser** command and requires a blank space between attributes. If you specify an empty list, only the user names are displayed.
-c	Displays the user attributes in colon-separated records, as follows: `# name:  attribute1:  attribute2:  ...` `  User:  value1:      value2:      ...` If a value contains a : symbol, then in the output : symbol is prefixed with the #! symbols.

Item	Description
-C	Displays the user attributes in colon-separated records that are easier to parse than the output of the **-c** flag: `#name:attribute1:attribute2: ...` `User1:value1:value2: ...` `User2:value1:value2: ...` The output is preceded by a comment line that has details about the attribute represented in each colon-separated field. If you also specify the **-a** flag, the order of the attributes matches the order specified in the **-a** flag. If you do not have a value for a given attribute, the field is still displayed, but is empty. If a value contains a `:` symbol, then in the output the `:` symbol is prefixed with `#!` symbols.The last field in each entry ends with a newline character rather than a colon.
-f	Displays the output in stanzas, with each stanza identified by a user name. Each *Attribute*=*Value* pair is listed on a separate line: `user:` `attribute1=value` `attribute2=value` `attribute3=value`
-R *load_module*	Specifies the loadable I&A module that is used to display the user account attributes. If the *domainlessgroups* attribute is set in the /etc/secvars.cfg file and the **-R** *LDAP* command is used, the attribute list is obtained from the LOCAL module. This condition applies if the user exists on the LOCAL module, and does not exist on the LDAP module. This condition also applies to the **-R** *files* command.

Exit Status

This command returns the following exit values:

Item	Description
0	The command runs successfully and all requested changes are made.
>0	An error occurred. The printed error message lists further details to the type of failure.

Security

Access Control: This command must be a general user program with execute (x) access for all users. Since the attributes are read with the access rights of the user who starts the command, some users might not be able to access all the information. This command must have the *trusted computing base* attribute.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

To get the full functionality of the command, besides the **accessauths**, the role must also have the **aix.security.user.audit** authorization.

On a Trusted AIX system, only users with authorization aix.mls.clear.read can list clearance attributes of other users. See Trusted AIX in the *Security* for more information.

Files Accessed:

Mode	File
r	**/etc/passwd**

Mode	File
r	/etc/security/user
r	/etc/security/user.roles
r	/etc/security/limits
r	/etc/security/environ
r	/etc/group
r	/etc/security/audit/config
r	/etc/security/enc/LabelEncodings

Examples

1. To display the user `id` and group-related information about the `smith` account in stanza form, enter the following command:

   ```
   lsuser -f -a id pgrp groups admgroups smith
   ```

   Information similar to the following is displayed:

   ```
   smith:
      ID=2457
      pgrp=system
      groups=system,finance,staff,accounting
      admgroups=finance,accounting
   ```

2. To display the user `id`, `groups`, and home directory of `smith` in colon format, enter the following command:

   ```
   lsuser -c -a id home groups smith
   ```

   Information similar to the following is displayed:

   ```
   # name: ID:home:groups
   smith:  2457:/home/smith:system,finance,staff,accounting
   ```

3. To display all the attributes of user `smith` in the default format, enter the following command:

   ```
   lsuser smith
   ```

   All the attribute information is displayed, with each attribute separated by a blank space.

4. To display all the attributes of all the users, enter the following command:

   ```
   lsuser ALL
   ```

   All the attribute information is displayed, with each attribute separated by a blank space.

Files

Item	Description
/usr/sbin/lsuser	Contains the **lsuser** command.
/etc/passwd	Contains basic user information.
/etc/security/limits	Defines resource quotas and limits for each user.
/etc/security/user	Contains the extended attributes of users.
/etc/security/user.roles	Contains the administrative role attributes of users.
/etc/security/environ	Contains the environment attributes of users.

Item	Description
/etc/group	Contains basic group attributes.
/etc/security/audit/config	Contains the audit configuration files.
/etc/security/enc/LabelEncodings	Contains label definitions for the Trusted AIX system.

lsusil Command

Purpose

Lists one or more user-specified installation location (USIL) instances.

Syntax

lsusil [**-R** *RelocatePath* | **ALL**]

Description

The **lsusil** command lists one or more USIL instances.

Flags

Item	Description
-R *RelocatePath*	The path to an existing USIL location

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/usr/sbin/lsusil	Contains the **lsusil** command.

lsvfs Command

Purpose

Lists entries in the **/etc/vfs** file.

Syntax

lsvfs { **-a** | *VfsName* }

Description

The **lsvfs** command lists entries in the **/etc/vfs** file. You can display information about a specific Virtual File System (VFS) type or all known VFS types.

Flag

Item	Description
-a	Lists all stanzas in the **/etc/vfs** file, including the default stanza.

Parameter

Item	Description
VfsName	Specifies the name of a virtual file system.

Examples

1. To list the vfs entry named `newvfs`, enter:

```
lsvfs newvfs
```

2. To list all vfs types, enter:

```
lsvfs -a
```

Files

Item	Description
/etc/vfs	Contains descriptions of virtual file system types.

lsvg Command

Purpose

Displays information about volume groups.

Syntax

lsvg [**-L**] [**-o**] | [**-n** *descriptorphysicalvolume*] | [**-i**] [**-l** | **-M** | **-p**] [**-m**] [**-P**] *volumegroup* ...

Description

The **lsvg** command displays information about volume groups. If you use the *volumegroup* parameter, only the information for that volume group is displayed. If you do not use the *volumegroup* parameter, a list of the names of all defined volume groups is displayed.

When information from the Device Configuration database is unavailable, some of the fields will contain a question mark (?) in place of the missing data. The **lsvg** command attempts to obtain as much information as possible from the description area when the command is given a logical volume identifier.

> **Note:** To determine a volume group's major number, use the **ls -al /dev/***VGName* command. This command lists the special device file that represents the volume group. The volume group major number is the same as the major device number of the special device file. For example, for a volume group named `ha1vg`, enter the following command:
>
> ```
> ls -al /dev/ha1vg
> ```
>
> This command returns the following:
>
> ```
> crw-rw---- 1 root system 52, 0 Aug 27 19:57 /dev/ha1vg
> ```

In this example, the volume group major number is 52.

You can use the System Management Interface Tool (SMIT) **smit lsvg** fast path to run this command.

Flags

Item	Description
-i	Reads volume group names from standard input.
-l	Lists the following information for each logical volume within the group specified by the *volumegroup* parameter: **LV** A logical volume within the volume group. **Type** Logical volume type. **LPs** Number of logical partitions in the logical volume. **PPs** Number of physical partitions used by the logical volume. **PVs** Number of physical volumes used by the logical volume. **Logical volume state** State of the logical volume. `Opened/stale` indicates the logical volume is open but contains partitions that are not current. `Opened/syncd` indicates the logical volume is open and synchronized. `Closed` indicates the logical volume has not been opened. **Mount point** File system mount point for the logical volume, if applicable.
-L	Specifies no waiting to obtain a lock on the Volume group. **Note** : If the volume group is being changed, using the **-L** flag gives unreliable data.
-m	Lists the mirror pool that each logical volume copy in the volume group belongs to.

Item	Description
-M	Lists the following fields for each logical volume on the physical volume: `PVname:PPnum [LVname: LPnum [:Copynum] [PPstate]]` **PVname** Name of the physical volume as specified by the system. **PPnum** Physical partition number. Physical partition numbers can range from 1 to 1016. **LVname** Name of the logical volume to which the physical partitions are allocated. Logical volume names must be system-wide unique names, and can range from 1 to 64 characters. **LPnum** Logical partition number. Logical partition numbers can range from 1 to 64,000. **Copynum** Mirror number. **PPstate** Only the physical partitions on the physical volume that are not current are shown as stale.
-n *descriptorphysicalvolume*	Accesses information from the descriptor area specified by the *descriptorphysicalvolume* variable. The information may not be current, since the information accessed with the -n flag has not been validated for the logical volumes. If you do not use the -n flag, the descriptor area from the physical volume that holds the most validated information is accessed, and therefore the information displayed is current. The volume group need not be active when you use this flag.
-o	Lists only the active volume groups (those that are varied on). An active volume group is one that is available for use.
-p	Lists the following information for each physical volume within the group specified by the *volumegroup* parameter: **Physical volume** A physical volume within the group. **PVstate** State of the physical volume. **Total PPs** Total number of physical partitions on the physical volume. **Free PPs** Number of free physical partitions on the physical volume. **Distribution** The number of physical partitions allocated within each section of the physical volume: outer edge, outer middle, center, inner middle, and inner edge of the physical volume.
-P	Lists the mirror pool that each physical volume in the volume group belongs to.

Information displayed if you do not specify any flags:

Item	Description
VOLUME GROUP	Name of the volume group. Volume group names must be unique systemwide and can range from 1 to 15 characters.
VOLUME GROUP STATE	State of the volume group. If the volume group is activated with the **varyonvg** command, the state is either active/complete (indicating all physical volumes are active) or active/partial (indicating some physical volumes are not active).
PERMISSION	Access permission: read-only or read-write.
MAX LVs	Maximum number of logical volumes allowed in the volume group.
LVs	Number of logical volumes currently in the volume group.
OPEN LVs	Number of logical volumes within the volume group that are currently open.
TOTAL PVs	Total number of physical volumes within the volume group.
ACTIVE PVs	Number of physical volumes that are currently active.
VG IDENTIFIER	The volume group identifier.
PP size	Size of each physical partition.
TOTAL PPs	Total number of physical partitions within the volume group.
FREE PPs	Number of physical partitions not allocated.
ALLOC PPs	Number of physical partitions currently allocated to logical volumes.
QUORUM	Number of physical volumes needed for a majority.
VGDS	Number of volume group descriptor areas within the volume group.
AUTO-ON	Automatic activation at IPL (yes or no).
CONCURRENT	States whether the volume group is Concurrent Capable or Non-Concurrent Capable.
AUTO-CONCURRENT	States whether you should autovary the Concurrent Capable volume group in concurrent or non-concurrent mode. For volume groups that are Non-Concurrent Capable, this value defaults to Disabled.
VG MODE	The vary on mode of the volume group: Concurrent or Non-Concurrent.
NODE ID	Node id of this node if volume group is varied on in concurrent node.
ACTIVE NODES	Node ids of other concurrent nodes that have this volume group varied on.
MAX PPs Per PV	Maximum number of physical partitions per physical volume allowed for this volume group.
MAX PVs	Maximum number of physical volumes allowed in this volume group. This information is displayed only for 32 and 128 PV volume groups.
LTG size	Logical track group size of the volume group. The maximum amount of data that can be transferred in one I/O request to the disks of the volume group. The LTG size will be displayed in kilobytes unless the LTG size is greater than 1 MB, in which case megabytes will be used. If the volume group was created on AIX 5.3, then it is capable of dynamically determining the LTG size based-on the disk topology and it is listed as Dynamic. If that capability is disabled by the user with the **varyonvg -M** option, then it will be listed as Static. If the capability does not exist because the volume group was created prior to AIX 5.3, then the VG will not be listed as Static or Dynamic.

Item	Description
BB POLICY	Bad block relocation policy of the volume group.
SNAPSHOT VG	Snapshot volume group name if the snapshot volume group is active, else snapshot volume group identifier.
PRIMARY VG	Original volume group name of a snapshot volume group if the original volume group is active, else original volume group identifier.
PV RESTRICTION	Displays the existing PV type restriction on the physical volumes comprising the volume group. A value of *none* indicates no PV restriction exists for the volume group. A value of *SSD* indicates the volume group has a PV restriction requiring all PVs to be *SSD* type PVs. No other values are supported.
INFINITE RETRY	Lists the infinite retry option of the volume group.
CRITICAL VG	Lists whether the `Critical VG` option is turned on or off for the volume group.
FS SYNC OPTION	Lists whether the logical volume manager resynchronizes the blocks allocated only by the Enhanced Journaled File System (JFS2), if JFS2 is mounted.
CRITICAL PVs	Lists Critical PVs option of the volume group. This information is available in IBM AIX 7.2 with Technology Level 1, or later.
ENCRYPTION	Lists the data encryption option of the volume group. This information is available in AIX 7 with 7200-05, or later.

Examples

1. To display the names of all active volume groups, enter the following command:

   ```
   lsvg -o
   ```
2. To display the names of all volume groups within the system, enter the following command:

   ```
   lsvg
   ```
3. To display information about volume group `vg02`, enter the following command:

   ```
   lsvg vg02
   ```

   The characteristics and status of both the logical and physical partitions of volume group `vg02` are displayed.
4. To display the names, characteristics, and status of all the logical volumes in volume group `vg02`, enter the following command:

   ```
   lsvg -l vg02
   ```

Files

Item	Description
/usr/sbin	Contains the directory where the **lsvg** command resides.

lsvgfs Command

Purpose

Displays a list of file systems belonging to a volume group.

Syntax

lsvgfs *volumegroup*

Description

The **lsvgfs** command displays a list of file systems that belong to the specified volume group.

Parameters

Item	Description
volumegroup	Specifies a volume group.

Examples

1. To display a list of file systems in the volume group vg02, enter the following command:

```
lsvgfs vg02
```

Exit Status

The **lsvgfs** command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Files

Item	Description
/usr/sbin	Contains the directory where the **lsvgfs** command resides.

lsvirprt Command

Purpose

Displays the attribute values of a virtual printer.

Syntax

lsvirprt [**-q** *QueueName* **-d** *DeviceName* { [**-f** *Format*] [**-n**] [**-a** *AttributeName* | **-s** *SectionName*] ... | **-i** | **-D** }]

Description

The **lsvirprt** command displays the attribute values for the virtual printer assigned to the *PrintQueueName* and *QueueDeviceName* variables.

The **lsvirprt** command becomes interactive if no flags are specified with the command. A list of print queue names is displayed, and a prompt appears requesting that the desired print queue name be selected. After a valid print queue name is selected, a prompt appears requesting that attribute names be entered. If an attribute name of * (asterisk) is entered, a list of all attributes is displayed.

> **Note:** Attribute names for default values of the **qprt** command line flags can be specified by entering the flag letters. For example, to view the default value for the **-w** flag (page width), enter the **w** attribute name. All other attribute names must be 2 characters long.

You can use the System Management Interface Tool (SMIT) **smit lsvirprt** fast path to run this command.

Flags

Item	Description
-a *AttributeName*	Specifies the name of an attribute for which information is to be displayed. The flag cannot be used with the **-s** flag. The **-a** flag can be specified many times to list multiple attributes. The *AttributeName* value can be a single-character name (for example, **j**), a simple two-character name (for example, **ci**), or a regular expression that specifies multiple attributes (for example, **^i.***.)
-d *QueueDeviceName*	Specifies the name of the queue device to which the virtual printer is assigned. This flag is optional, but can be specified only if the **-q** flag is specified.
-D	Displays data streams supported by a given queue and queue device name variable values. The **-D** flag displays the default data stream first and all other supported data streams in alphabetical order.
-f *Format*	Specifies the display format for attribute information. Attribute information includes the attribute value, limits field, and attribute description. The *Format* value is specified in **printf** format. The **-f** *Format* option also supports the following predefined set of position arguments: **Note:** [*.*] is not a required element for the following format values. **%1$[*.*]s** Message catalog name **%2$[*.*]d** Message number **%3$[*.*]s** Attribute name **%4$[*.*]s** Limits field **%5$[*.*]s** Attribute value **%6$[*.*]s** Attribute description **%7$c** Second character of attribute name.
-i	Sets the command to interactive mode. The **-q** and **-d** flags must be specified with the **-i** flag. If values have been assigned to the *QueueName* and *DeviceName* variables, the command does not prompt for the queue and device names and accepts attribute names interactively.
-n	Displays only the specified attributes that have nonnull values.
-s *SectionName*	Specifies a section name in the virtual printer attribute database of the specified queue and queue device. The *SectionName* values begin with two underscores and contain three characters that identify a section. For example, the name of a section that contains all the flag attributes is **__FLG**. The **-s** flag can not be used with the **-a** flag. This option can be repeated to list multiple attributes. The *SectionName* variable value can be a regular expression.

Item	Description
-q *PrintQueueName*	Specifies the name of the print queue to which the virtual printer is assigned. This flag is optional, but can be specified only if the **-d** flag is specified.

Examples

1. To show the attribute values for the `w` (default page width) and `si` (user to receive the "Intervention Required" messages) attributes for the virtual printer assigned to the `mypro` queue device on the `proq` print queue, enter:

   ```
   lsvirprt -dmypro -qproq -a w -a si
   ```

   The output from this command is:

   ```
   Name            Description                     Value
   _w              COLUMNS per page                136
   si              USERS to get intervention messages
   ```

2. To show the same attributes in Example 1, but the be prompted for the flag values, enter:

   ```
   lsvirpt
   ```

   The output from this command is:

   ```
   1       e4039c          @piobe            ibm4039 (PCL Emulation)
   2       e4039s          @piobe            ibm4039 (PostScript)
   3       fjzhp4s         jzfile            hplj-4 (PostScript)
   4       hpc14           hp@pc15           hplj-4 (PCL)
   ...
   ```

3. To list attributes in a section for header and trailer pipelines for the `que` queue and the `dev` device, enter:

   ```
   lsvirpt -qque -ddev -s__HTP
   ```

   The output from this command is:

   ```
   Name    Description                     Value
   sh      Pipeline for Header Page        %Ide/pioburst %F[H]
                                           %Idb/H.ascii |
                                           %Ide/pioformat
                                           -@%Idd/%Imm
                                           -!%Idf/piof5202 -L! -J!
                                           %IsH
   st      Pipeline for Trailer Page       %Ide/pioburst %F[H]
                                           %Idb/T.ascii |
                                           %Ide/pioformat
                                           -@%Idd/%Imm
                                           -!%Idf/piof5202-L!
                                           -t%o%G_1%r%{14}%-%d
                                           %IsT
   ```

4. To list all the data streams supported for the `que` queue and the `dev` device, enter:

   ```
   lsvirpt -qque -ddev -D
   ```

   The output from this command is:

   ```
   a ASCII
   p pass-through
   s PostScript
   ```

5. To list names and descriptions of all attributes in a printer attribute database for the `que` queue and the `dev` device in a specific format, enter:

```
lsvirpt -qque -ddev -a'.*' -f' %3$5.5s: %6$s\\n'
```

The output from this command is:

```
__FLG:  Values That May Be Overridden With Flags
_A:     stderr returned?
_E:     Double spacing flag
_F:     (not used) Font file name
_H:     Name to Replace Host Name of Burst Page
...
```

6. To list all the sections in a printer attribute data base for the que queue and the dev device in a specific format, enter:

```
lsvirpt -qque -ddev -a'__.*' -f'%3$s: %6$s\\n'
```

The output from this command is:

```
__FLG:  Values That May Be Overridden With Flags On the Command
        Line
__SYS:  Other Values Of Interest To the Streams Administrator
__IDS:  Pipelines For Input Data Streams (2 char,1st="i",2nd=data
        stream name)
__PFL:  Flags Prohibited For Input Data Streams (2 char,1st="I",
        2nd=data stream name)
__FIL:  Command Strings For Filter Flags (2 char, 1st="f",
        2nd=flag)
__DIR:  Directories
...
```

Files

Item	Description
/etc/qconfig	Contains the configuration file.
/usr/sbin/lsvirprt	Contains the **lsvirprt** command.
/var/spool/lpd/pio/@local/custom/*	Contains virtual printer attribute files.
/var/spool/lpd/pio/@local/ddi/*	Contains the digested virtual printer attribute files.

lsvmode Command

Purpose

Display the current video mode of the X server.

Note: This command is usable only while the X server is running.

Syntax

lsvmode

Description

The **lsvmode** command displays the current output device and viewport size used by the X server.

Security

Access Control: Any User

Auditing Events: None

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

To display the current video mode of the X server.

```
lsvmode
```

Something similar to the following displays:

```
Current video mode information
Logical screen size   [1024x768]
Viewport size        [ 640x480]
Vertical sync. (Hz)       [60]
Active output device  [LCD][CRT]
```

Files

Item	Description
/usr/bin/X11/lsvmode	Contains the **lsvnode** command.

lsvpd Command

Purpose

Lists the vital product data (VPD) associated with the field replaceable units (FRUs) configured on a system.

Syntax

`lsvpd` [`-m`] [`-s` *serial_number*] [`-t` *type_model*] [`-v`]

Description

The `lsvpd` command collects vital product data (VPD) for field replaceable units (FRUs). It reads the appropriate device configuration object classes in the Object Data Manager (ODM) and gathers VPD and general system information. The `lsvpd` command can extract additional VPD by reading data structures that are specific to the platform on which it is running. Data is provided in a format that aids service personnel in monitoring device quality and performance.

Note: Output from the `lsvpd` command is informational only and subject to change as hardware definitions change. Portable applications should not parse this data.

Flags

Item	Description
`-m`	Distinguishes between a FRU with global VPD and a FRU with partition private VPD. FRUs with global VPD begin with a line in the format of `*FC ********`. FRUs with partition private VPD begin with a line in the format of `*FC  ========`. If this flag is not specified, the output begins with a line in the format of `*FC  ????????`. For LPARs, this option distinguishes between FRUs associated with the overall system and FRUs assigned to a specific partition.
`-s` *serial_number*	Specifies the serial number for the system. The optional *serial_number* parameter is obsolete and should not be used. If the serial number is entered, that value will be used in the output of the command. In some cases, `lsvpd` is unable to automatically determine the serial number. In these cases, the user must supply the value in order for it to be displayed in the command output.
`-t` *type_model*	Specifies the type model for the system. The optional *type_model* parameter is obsolete and should not be used. If the type model is entered, that value will be used in the output of the command. In some cases, `lsvpd` is unable to automatically determine the type model. In these cases, the user must supply the value in order for it to be displayed in the command output.
`-v`	Produces verbose output for debugging purposes only.

Exit Status

Item	Description
`0`	The command completed successfully.
`1`	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. Output for the `lsvpd` command is similar to the following.

   Note: Portable applications should not parse this data.

```
*VC 5.0
*TM IBM,7029-6E3
*SE IBM,0110B721E
*PI 000B721E
*OS AIX 5.3.0.0
```

```
*FC ????????
*DS Platform Firmware
*YL U0.1-P1-X1/Y1
*RM 3F041029
*VK RS6K
*FC ????????
*DS System Firmware
*YL U0.1-P1-X1/Y2
*RM RG041029_d79e00_regatta
*VK RS6K
*FC ????????
*DS System VPD
*YL U0.1
*SE 10B721E
*TM 7029-6E3
*MN IBM980
*VK RS6K
*PA Y
*BR I0
*FC ????????
*DS PS CEC OP PANEL
*YL U0.1-L1
*SN YL1124350190
*EC H64013
*CC 28D3
*FN  97P3352
*DC BD 200210290851
*VK RS6K
*FC ????????
*DS 2 WAY BACKPLANE
*YL U0.1-P1
*SN YL1123354433
*PN 80P3099
*CC 26F5
*CE 1
*FN  80P3099
*VK RS6K
*FC ????????
*DS CSP
*YL U0.1-P1-X1
*SN YL1024360048
*PN 80P5573
*CC 28D0
*CE 1
*FN  80P5573
*RM 3F041029
*VK RS6K
*FC ????????
*DS IBM 1.8V VRM
*YL U0.1-P1-V1
*FN  24P6892
*VK RS6K
*FC ????????
*DS IBM 2.5V VRM
*YL U0.1-P1-V2
*FN  53P5623
*VK RS6K
*FC ????????
*DS IBM 1.2V VRM
*YL U0.1-P1-V3
*FN  53P5621
*VK RS6K
*FC ????????
*DS A IBM AC PS
*YL U0.1-V2
*SN YL1023C90045
*EC H85582
*CC 51B5
*FN  97P5101
*VK RS6K
*FC ????????
*DS IBM Air Mover
*YL U0.1-F1
*FN  53P4612
*VK RS6K
*FC ????????
*DS IBM Air Mover
*YL U0.1-F2
*FN  53P4612
*VK RS6K
*FC ????????
*DS IBM Air Mover
```

```
*YL U0.1-F3
*FN  53P4612
*VK RS6K
*FC ????????
*DS VSBPD4E1  U4SCSI
*YL U0.1-P2
*SN YL11243550F4
*PN 80P4611
*EC H85823
*CC 28D2
*FN  80P4610
*FS
*VK RS6K
*FC ????????
*DS MEDIA BACKPLANE
*YL U0.1-P4
*SN YL1124341459
*PN 80P3510
*EC H85610
*CC 28D1
*FN  80P3516
*VK RS6K
*FC ????????
*DS PCI-X Dual Channel Ultra320 SCSI Adapter
*AX sisscsia1
*PL 1Z-08
*CD 10140266
*PN 97P6513
*FN 97P6513
*SN YL11A5013461
*MN 001A
*EC 1
*RM 05080064
*Z0 5702
*YL U0.1-P1-I1
*FC ????????
*DS IDE DVD-ROM Drive
*AX cd0
*PL 1G-19-00
*MF IBM
*TM DROM00205
*RL NR38
*Z0 058002028F000010
*YL U0.1-P1-X1/Q6-A0
*FC ????????
*DS 16 Bit LVD SCSI Disk Drive
*AX hdisk0
*PL 1S-08-00-5,0
*MF IBM
*TM ST336607LC
*FN 00P3068
*RL 4335304A
*SN 000D7D3B
*EC H12094
*PN 00P2676
*Z0 000003129F00013E
*Z1 0812C512
*Z2 0002
*Z3 04341
*Z4 0001
*Z5 22
*Z6 H12094
*YL U0.1-P1/Z1-A5
*FC ????????
*DS 16 Bit LVD SCSI Disk Drive
*AX hdisk1
*PL 1S-08-00-8,0
*MF IBM
*TM ST336607LC
*FN 00P3068
*RL 4335304A
*SN 000D7996
*EC H12094
*PN 00P2676
*Z0 000003129F00013E
*Z1 0812C512
*Z2 0002
*Z3 04340
*Z4 0001
*Z5 22
*Z6 H12094
*YL U0.1-P1/Z1-A8
```

```
*FC ????????
*DS Diskette Drive
*AX fd0
*PL 01-D1-00-00
*YL U0.1-P1-X1-D1
*FC ????????
*DS Asynchronous Terminal
*AX tty0
*PL 01-S1-00-00
*YL U0.1-P1-X1/S1-L0
*FC ????????
*DS SCSI Enclosure Services Device
*AX ses0
*PL 1S-08-00-15,0
*MF IBM
*TM VSBPD4E1  U4SCSI
*RL 4610
*SN 243550F4
*Z0 0D0002022F004000
*FN 80P4610
*FL DB1
*FS
*YL U0.1-P1/Z1-Af
*FC ????????
*DS IBM MS 512 MB
*YL U0.1-P1-M5
*SN YL10243591YT
*PN 00P5767
*CC 30D2
*FN  00P5767
*SZ 512
*VK RS6K
*FC ????????
*DS IBM MS 512 MB
*YL U0.1-P1-M7
*SN YL10243591YP
*PN 00P5767
*CC 30D2
*FN  00P5767
*SZ 512
*VK RS6K
*FC ????????
*DS IBM MS 512 MB
*YL U0.1-P1-M4
*SN YL1024359208
*PN 00P5767
*CC 30D2
*FN  00P5767
*SZ 512
*VK RS6K
*FC ????????
*DS IBM MS 512 MB
*YL U0.1-P1-M2
*SN YL1024359204
*PN 00P5767
*CC 30D2
*FN  00P5767
*SZ 512
*VK RS6K
```

Location

/usr/sbin/lsvpd

lsvsd Command

Purpose

Displays configured virtual shared disks and their characteristics.

Syntax

lsvsd [**-l** | **-s**[*vsd_name*...]] | [**-i**]

Description

The **lsvsd** command displays information about virtual shared disks currently configured on the node on which the command is run. If a list of virtual shared disks follows the flags, information about those virtual shared disks is displayed. **lsvsd** with no arguments or flags lists the names of all the virtual shared disks currently configured on the node.

The **lsvsd** command displays information about both the configuration and the usage of a virtual shared disk.

You can use the System Management Interface Tool (SMIT) to run the **lsvsd** command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Show All Managed Virtual Shared Disk Characteristics** option.

Flags

-l
Lists the name of the virtual shared disk, the minor number, the state, the current server node number, and, at the server only, the major and minor number of the logical volume. (This flag is lowercase **l**, as in **l**ist.)

The state field can have one of the following values:

STP Stopped
SUS Suspended
ACT Active

An asterisk (*) in front of any of these values indicates that the virtual shared disk has been fenced from this node.

This flag is not compatible with the **-s** flag.

The *server_list* of the virtual shared disk is listed.

-s
Lists usage statistics about the virtual shared disks. It lists the number of local logical read and write operations, the number of remote logical read and write operations, the number of client logical read and write operations, the number of physical reads and writes, and the number of 512-byte blocks read and written. The number of blocks read and written is cumulative, so issue **ctlvsd -V** to reset this count before measuring it.

The local logical operations are requests which were made by a process executing at the local node, whereas the remote logical operations were made by a process executing on a remote node. *Client operations* are those local logical requests that cannot be satisfied locally, and have to be sent to a remote node. *Physical* operations are those server operations which must be passed to the underlying disk device.

This flag is not compatible with the **-l** flag.

-i
Lists the “node to IP address” map that is currently used by the virtual shared disk driver.

Parameters

vsd_name
Specifies a virtual shared disk. This parameter is valid only with the **-l** and **-s** flags.

Security

You must be in the AIX **bin** group to run this command.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to *RSCT Administration Guide* .

Examples

1. To list all virtual shared disks in the system, enter:

   ```
   lsvsd
   ```

   The system displays a message similar to the following:

   ```
   vsd00

   vsd01

   .
   .
   .
   ```

2. To list virtual shared disks and their characteristics, enter:

   ```
   lsvsd -l
   ```

   The system displays a message similar to the following:

   ```
   minor  state  server  lv_major  lv_minor  vsd_name  size (MB)
   83     STP      -1        0         0      vsdn08v3    20
   84     STP      -1        0         0      vsdn08v4    16
   ```

3. To list statistics about virtual shared disks and precede the column output with a header, enter:

   ```
   lsvsd -s
   ```

   The system displays a message similar to the following:

   ```
   lc-rd lc-wt rm-rd rm-wt c-rd c-wt  p-rd  p-wt   br   bw  vsd_name
      84    84  2858   169    0    0   348   253  164  184  vsd.vsd1
       0     0     0     0    0    0     0     0    0    0  vsd.rl01
       0     0     0     0    0    0     0     0    0    0  vsd.rl02
   ```

The following table spells out the names of the headers used in the displays for the `-l` and `-s` options:

Header
Meaning

minor
Virtual shared disk minor number

state
State of this virtual shared disk:*active, stopped, suspended*

server
Primary node for this virtual shared disk

lv major
Logical volume major number

lv minor
Logical volume minor number

vsd_name
Name of this virtual shared disk

lc-rd
Local logical reads

lc-wt
Local logical writes

rm-rd
Remote logical reads

rm-wt
Remote logical writes

c-rd
Client logical reads

c-wt
Client logical writes

p-rd
Physical reads

p-wt
Physical writes

br
Blocks read

bw
Blocks written

Location

/opt/rsct/vsd/bin/lsvsd

lswlmconf Command

Purpose

Lists Workload Manager (WLM) configurations.

Syntax

lswlmconf [**-r** | **-s** | **-c** | **-d** *Config*] [**-l**] [**-t** *TimeSpec*]

Description

The **lswlmconf** command lists by default all WLM configurations and, using its flags, it is able to do the following:

- Tell which is the current configuration or set name.
- Give the list of all existing regular WLM configurations.
- Give the list of all existing WLM configuration sets.
- Tell which configuration of a set is (or would be) applicable currently (or at some time of the week).
- Tell the type of a configuration.

Flags

Item	Description
-c	Restricts the displayed configurations to the current configuration or set.
-d *Config*	Restricts the displayed configurations to the *Config* configuration or set.
-l	Modifies the way dates are displayed for configuration sets (ineffective for regular configurations). Sets are displayed with their currently applicable regular configuration, in the form *confset/config*.

Item	Description
-r	Restricts the displayed configurations to regular configurations only.
-s	Restricts the displayed configurations to configuration sets only.
-t *TimeSpec*	Uses *TimeSpec* instead of the current time to display applicable regular configuration of sets. *TimeSpec* consists in the day of the week (0 for Sunday to 6 for Saturday) and the time of the day in 24 hours format separated with a comma, in a form similar to time ranges as described in the **confsetcntrl** command. For example, to know which configuration would apply on Mondays at noon, use `-t 1,12:00`. **Note:** The **-t** flag is only effective with **-l** flag.

Examples

The following examples demonstrate how to display, change, and use WLM configurations using the **lswlmconf** command, the **confsetcntrl** command, the **wlmcheck** command, and the **wlmcntrl** command.

1. To find the WLM configurations, type:

   ```
   lswlmconf
   ```

   The output to this command might look similar to the following:

   ```
   standard
   template
   fvtrules
   fvtlimits
   fvtregul
   fvtdfct
   fvtsynt
   fvtthreads
   ```

2. To show the current WLM configuration, type:

   ```
   lswlmconf -c
   ```

   The output might look similar to the following:

   ```
   fvtlimits
   ```

3. To show configuration sets, use the **lswlmconf** with the **-s** flag as follows:

   ```
   lswlmconf -s
   ```

   Since this example configuration contains no configuration sets, this command produces a message indicating that no matching configuration was found.

4. In order to create a configuration set using `standard` as the default configuration, type:

   ```
   confsetcntrl -C confset1 standard
   ```

5. Now, use the **lswlmconf** command to show the new configuration set, as follows:

   ```
   lswlmconf -s
   ```

   The command now produces the following output:

   ```
   confset1
   ```

6. In order to use the `fvtlimits` configuration for `confset1` on week days (Monday through Friday) by specifying a time range, type:

   ```
   confsetcntrl -d confset1 -a fvtlimits 1-5
   ```

7. You might want this configuration only in the morning. You cannot change a time range. Instead you must remove the time range and then create a new time range.

   First, remove the old time range, as follows (confsetcntrl accepts day names, as reported by **locale day** or **locale abday** commands):

   ```
   confsetcntrl -d confset1 -r fvtlimits monday-friday
   ```

   Then create the new time range, as follows:

   ```
   confsetcntrl -d confset1 -a fvtlimits 1-5,8:00-12:00
   ```

8. In order to add another time range for using the `fvtregul` configuration on Sundays, type:

   ```
   confsetcntrl -d confset1 -a fvtregul 0
   ```

9. In order to display configuration set `confset1`, type:

   ```
   confsetcntrl -d confset1
   ```

   In this example, this command produces the following output:

   ```
   fvtlimits:
           time = "1-5,8:00-12:00"

   fvtregul:
           time = "0"

   standard:
           time = "-"
   ```

10. In order to create a configuration set called `confset2` using `template` as the default configuration, type:

    ```
    confsetcntrl -C confset2 template
    ```

    In order change `confset2` so it will use the configuration `fvtsynt` every nigh, type:

    ```
    confsetcntrl -d confset2 -a fvtsynt 18:00-10:00
    ```

11. In order to display the list of regular configurations, type:

    ```
    lswlmconf -r
    ```

    In this example, this produces the following output, (which demonstrates that in this example the list of regular configurations has not changed):

    ```
    standard
    template
    fvtrules
    fvtlimits
    fvtregul
    fvtdfct
    fvtsynt
    fvtthreads
    ```

    However, as expected, the list of configurations sets in this example has changed, as shown by the following command:

    ```
    lswlmconf -s
    ```

    This command produces the following output in this example:

    ```
    confset1
    confset2
    ```

12. In order to show which configuration would be currently active when that the **date** command reports the current time as `Tue Jul 16 18:55:10 EET 2002` with configuration set `confset2`, type:

```
lswlmconf -d confset2 -l
```

In this example, this command produces the following output:

```
confset2/fvtsynt
```

You can also show which configurations would be active at another time. To show which configurations would be active on Sunday at 9:00am, type:

```
lswlmconf -l -t 0,9:00
```

This command produces the following output in this example:

```
standard
template
fvtrules
fvtlimits
fvtregul
fvtdfct
fvtsynt
fvtthreads
confset1/fvtregul
confset2/fvtsynt
```

In order to display this information only for configuration sets, type:

```
lswlmconf -s -l -t 0,9:00
```

This produces the following output in this example:

```
confset1/fvtregul
confset2/fvtsynt
```

13. In order to remove configuration set `confset2`, type:

```
confsetcntrl -D confset2
```

lswlmconf -s now produces the following output in this example:

```
confset1
```

14. In order to check configuration set `confset1`, use the **wlmcheck** command as follows:

```
wlmcheck -d confset1
```

In this example, this produces the following output:

```
WLM is not running.
Checking classes and rules for 'confset1' configuration...
fvtlimits/System
fvtlimits/Default
fvtlimits/Shared
fvtlimits/login
fvtregul/System
fvtregul/Default
fvtregul/Shared
standard/System
standard/Default
standard/Shared
```

15. In order to start using configuration set `confset1` used in this example, type:

```
wlmcntrl -a -d confset1
```

The command **lswlmconf -c** now produces the following output:

```
confset1
```

The command **lswlmconf -cl**, which shows the active regular configuration, now produces the following output:

```
confset1/standard
```

Files

The configurations or sets files are subdirectories of **/etc/wlm**.

lswpar Command

Purpose

Lists characteristics of workload partitions.

Syntax

Tabular Formats:

lswpar [**-b** | **-Br** | **-Bf** | **-D** | **-I** | **-M** | **-N**] [**-X**] [**-a** *fieldname* [,...]] [**-q**] [**-s** *state*] [**-t** *type*] [*wparname* ...]

Paragraph Formats:

lswpar {**-G** | **-L** | **-R** | **-S** | **-T**} [**-s** *state*] [**-t** *type*] [*wparname* ...]

Delimited Formats:

lswpar {**-c** | **-d** *delim*} [**-a** *fieldname* [,...] | **-G** | {**-b** | **-Br** | **-Bf** | **-D** | **-X** **-I** | **-M** | **-N**} [**-a** *fieldname* [,...]] | **-R** | **-S** | **-T**] [**-q**] [**-s** *state*] [**-t** *type*] [*wparname* ...]

Description

The **lswpar** command prints information about 1 or more specified workload partition (or all workload partitions if none are specified) to standard output.

You can filter all listings according to the following workload partition states by using the **-s** flag:

Item	Description
Defined	The workload partition has been defined by the **mkwpar** command and is ready for use, but is not active. Start workload partitions in this state with the **startwpar** command.
Loaded	The workload partition has been configured in the kernel, but processes have not yet been started. **Note:** This state is visible only to programmatic consumers that use the **lswpar** command to start a workload partition.
Active	The workload partition is running normally.
Frozen	A checkpoint operation is initiated, and the processes of the workload partition are quiesced, awaiting the storing phase. **Note:** The **Frozen** state is only visible when you use the **lswpar** command to checkpoint a workload partition. The checkpoint or restart function requires additional software package other than base WPAR.

Item	Description
Paused	A checkpoint or restart operation has been performed, and the processes of the workload partition are ready to be resumed or killed. The checkpoint or restart functionality requires additional software.
Maintenance	A workload partition can be put into maintenance mode with the **startwpar** command. During maintenance mode, the workload partition has been configured in the kernel and the file systems have been mounted, but processes do not start.
Moving	An asynchronous checkpoint-restart operation has been performed. Although the workload partition is **Active** on the arrival server, the workload partition appears in the **Moving** state on the departure server until all resources have been successfully transferred. The checkpoint or restart functionality requires additional software.
Transitional	An administrative operation is in progress. The workload partition is being created, started, stopped, configured, and so on.
Broken	An administrative operation failed, leaving this workload partition in an unusable state.
Error	An error occurred because of invalid elements such as workload partition name and flags.

You can filter all listings according to the following workload partition types by using the **-t** flag:

Item	Description
Application	This type is an application workload partition, running a single process (or a group of processes that are invoked by that means) without isolated system services. The process or group of processes inherits its operating environment (file systems, security, devices. and so on) from the environment where the application workload partition was created.
System	This type is a system workload partition, emulating an independent, fully functional instance of the operating system.

If additional checkpoint or restart software is installed, you can also specify the following type:

Item	Description
Checkpointable	This workload partition is enabled for checkpoint or restart functions. **Tip:** This type is not a mutually exclusive workload partition type. **Checkpointable** workload partitions are still either **System** or **Application** workload partitions.

If additional versioned workload partition software is installed, you can also specify the following type:

Item	Description
Versioned	This workload partition is running in operating system compatibility mode. **Tip:** This type is not a mutually exclusive workload partition type. **Versioned** workload partitions are still **System** workload partitions.

Versioned

Tabular Formats

If no options are used, the output is tabular as shown in the following example:

```
Name         State    Type    Hostname    Directory   RootVG WPAR
--------------------------------------------------------------------
wpar name    state    type  hostname    root directory   yes/no
...            ...    ...   ...           ...             ...
```

In tabular formats, there might be multiple records per WPAR. The **-D**, **-I**, **-M**. and **-N** flags display in a tabular format, but can be combined with the **-c** and **-d** flags to generate a delimited format. You can use the **-a** flag to customize which fields are displayed in tabular formats. You can use the **-q** flag to suppress the table headers.

Tip: Do not rely on the exact format and content of tabular output for automated purposes. Delimited formats are provided for output that can be parsed.

The width of each field in a tabular format is expanded according to the longest value in that column. Therefore, the output might wrap on narrow screens, depending on the fields requested.

Paragraph Formats

In paragraph formats, each field has one value for one WPAR. You can use the **-G**, **-R**, **-S**, and **-T** flags to display paragraph-style subsets of workload partition configurations. The **-L** flag displays a long listing, which is a combination of the data that is presented by the **-D**, **-G**, **-I**, **-M, -N**, **-R**, **-S**, and **-T** flags. Otherwise, formats cannot be combined.

Delimited Formats

Delimited formatsare used to produce machine-readable formats. You can select any delimiting characters. You can generate delimited formats by using the **-c** or **-d** flag. You can use the **-a** flag to customize which fields are displayed. You can use the **-q** flag to suppress the header line. The paragraph format flags (**-G**, **-R**, **-S**, and **-T**) and tabular format flags (**-D**, **-I**, **-M**, and **-N**) can be used individually to limit the display to the corresponding predefined set of fields.

Flags

Item	Description
-a *fieldname*	Limits tabular or delimited displays to the specified 1 or more fields. Multiple field names must be separated by commas with no spaces. This flag is mutually exclusive with the **-G**, **-R**, **-S**, **-L**, or **-T** flag. By default, the display consists of 1 WPAR per line. You can specify any of the following fields: **General** • **Name** (the WPAR name) • **Cid** (the ID of aWPAR • **Key** (the key of a WPAR • **Rootvgwpar** RootVG WPAR (A yes/no value is displayed to identify whether the WPAR is a RootVG WPAR • **Uuid** (the UUID of a WPAR • **Vipwpar** VIP WPAR (A yes/no value is displayed to identify whether the WPAR is a VIP WPAR. This field only applicable for an Application WPAR • **State** • **Type** (system or application) • **Hostname** • **Routing** • **Directory** • **Privateusr** The sample output is displayed as shown here: 0> lswpar -a name,privateusr test Name Private /usr? --- test no • **Script** (user-supplied start or stop script) • **Auto** – If the value for this field is **yes**, the process is automatically started on global system restart. – If the value for this field is **no**, the process is not automatically started on global system restart. • **Application** (tracked process for application WPARs) • **Checkpointable** • **Owner** • **OStype** (a non-zero value indicates a versioned WPAR, a value of 0 or null indicates a native WPAR) **Resource Controls** • **Active** – If the value for this field is **yes**, resource controls are active – If the value for this field is **no**, resource controls are inactive • **Rset** • **Shares_CPU** • **CPU** • **Shares_memory** • **Memory** • **ProcVirtMem** • **TotalProcesses** • **TotalThreads** • **totalPTYs** • **totalLargePages** • **totalVirtmem** • **pct_msgIDs** • **pct_semIDs** • **pct_shmIDs** • **pct_pinMem**

Item	Description
	(*Fields that you can specify with the* **-a** *flag, are as follows*) **Devices** • **Name** (the name of the WPAR) • **Devname** (the name of the device) • **Devtype** (pseudo, disk, clone) • **Rootvg** The display consists of 1 device per line. The following displays sample output: 0> lswpar -Da name,devname,rootvg test Name Device Name RootVG ------------------------------ test hdisk1 yes 0> lswpar test Name State Type Hostname Directory RootVG WPAR -- test D S test /wpars/test yes **Kernel Extensions** • **Name** (the name of the WPAR) • **Kext** (the full path to the kernel extension) • **Local** • **Major** • **kextstatus** (allocated or exported) • **checksum** (checksum of the kernel extension) • **mtime** (modification time of the kernel extension) The display consists of 1 kernel extension per line. **WPAR-Specific Routes** A workload partition might have more than 1 route. So if you use the **-I** flag, you can specify the **-a** flag with the following fields: • **name** (the WPAR name) • **rtdest** • **rtgateway** • **rtinterface** • **rttype** • **rtfamily** The display consists of 1 route per line. **Networks** A WPAR might have more than 1 network. So when you use the **-N** flag, you can specify the **-a** flag with the following fields: • **Name** (the WPAR name) • **Interface** • **Address** • **Netmask** • **Broadcast** The display consists of 1 network per line. **Mounts** A workload partition might have more than 1 mount. So when you use the **-M** flag, you can specify the **-a** flag with the following fields: • **Name** (the WPAR name) • **Mountpoint** (the mount point name) • **Device** (the object mounted) • **Vfs** (the virtual-file-system type) • **Nodename** (node name, if the mount is remote) • **Options** (any mount options) The display consists of 1 mount per line. **Security** • **Privs** (the list of privileges) **Operation** • **Opname** (the name of the administration operation that is being performed) • **Oppid** (the process ID of the operation) • **Opstart** (the start time of the operation)

Item	Description
	Bootlist When you use the **-b** flag, you can specify the **-a** flag with the following fields: • **name** (the name of the WPAR) • **bootlist** (an ordered list of bootsets in comma-separated format) The display consists of 1 bootlist per line. **Bootset** When you use the **-Br** flag, you can specify the **-a** flag with the following fields: • **name** (the name of the WPAR) • **devname** (the name of the device) • **vdevname** (the name of the virtual device) • **rootvg** • **bootset** (the bootset devices of the WPAR) The display consists of 1 bootset per line. When you use the **-Bf** flag, you can specify the **-a** flag with the following fields: • **name** (the name of the WPAR) • **mountpoint** (the name of the mountpoint) • **device** (the object mounted) • **vfs** (the type of the virtual file system) • **options** (any mount options) • **bootset** (the bootset file systems of the WPAR) The display consists of 1 bootset per line.
-b	Shows the bootlist of the workload partition. If the **-c** or **-d** flag is not specified, the output for each WPAR has the following tabular format: `Name - Bootlist`
-Br	Produces detailed bootset information for each requested RootVG WPAR. If the **-c** or **-d** flag is not specified, the output for each WPAR has the following tabular format: `Name - Device Name - Type - Virtual Device - RootVG - Bootset`
-Bf	Produces detailed bootset information for each requested non-RootVG WPAR. If the **-c** or **-d** flag is not specified, the output for each WPAR has the following tabular format: `Name - Mount Point - Device - Vfs - Options - Bootset`
-c	Produces colon-separated output suitable for machine parsing. It is mutually exclusive with the **-L** flag. The default output format (when the **-D**, **-G**, **-I**, **-M**, **-N**, **-R**, **-S**, and **-T** flags are not used) is as follows: `name:state:type:hostname:directory` The **state** field is one or more of the following valid states: **D** Defined **L** Loaded **A** Active **F** Frozen **P** Paused **N** Maintenance **M** Moving **T** Transitional **B** Broken **E** Error The **type** field is one or more of the following valid types: **A** Application workload partition **S** System workload partition **L** Versioned system workload partition
-d *delim*	Produces delimiter-separated output suitable for machine parsing. It is mutually exclusive with the **-L** flag. The output format when the **-d** flag is specified is the same as with when the **-c** flag is specified, but with *delim* as the delimiter output between fields.

Item	Description
-D	Produces detailed device information for each requested WPAR. It is mutually exclusive with the **-G**, **-I**, **-L**, **-M**, **-N**, **-R**, **-S**, or **-T** flag. If the **-c** or **-d** flag is not specified, each WPAR output has the following tabular format: `==========` *`Name - Device Name - Type - Virtual Device - RootVG - Status`*
-G	Produces detailed general setting information for each requested WPAR. It is mutually exclusive with the **-I**, **-L**, **-M**, **-D**, **-N**, **-R**, or **-T** flag. If you do not specify the **-c** or **-d** flag, each workload partition output has the following paragraph format: `==========` *`Name - State`* `==========` `Type:` *`{S|A}`* `Hostname:` *`Hostname`*`WPAR` `-Specific Routing:` *`{yes|no}`* `Directory:` *`Directory`* `Start/Stop Script:` *`/path/to/userScript`* `Auto Start:` *`{yes|no}`* `Private /usr:` *`{yes|no}`* `Checkpointable:` *`{yes|no}`* `Application:` *`/path/to/trackedProcess`* `Owner:` `Architecture: WPAR compatibility architecture` `OStype:` *`<i>Integer value representing operating system type<i>`* `Cross-WPAR IPC: {yes|no}` `UUID:` *`String value representing universally unique ID`* With the **-c** or **-d** flag, the output is as follows: `name:state:type:rootvgwpar:hostname:routing:directory:owner:script:` `auto:privateusr:checkpointable:application:ostype`
-I	Produces detailed information about user-specified network routes. The **-I** flag is mutually exclusive with the **-D**, **-G**, **-L**, **-M**, **-N**, **-R**, **-S**, or **-T** flag. The **-I** flag displays only the routing table entries that are explicitly specified with the **-I** flag of the **mkwpar**, **wparexec**, or **chwpar** command. To see the full routing table for a workload partition, use the **netstat** command with the **-r** and **-@** flags. If you do not specify the **-c** or **-d** flag, tabular output is as produced as shown in the following example: `Name  Type  Destination  Gateway  Interface` `----------` *`name  net|host  destination  gateway  if`* *`...  ...  ...  ...  ...`* With the **-c** or **-d** flag, delimited output is produced as shown in the following example: `name:rttype:rtdest:rtgateway:rtinterface:rtfamily` You can use the **-I** flag with the **-a** flag to limit the output to any combination of the following fields: • **name** (the workload partition name) • **rtdest** • **rtgateway** • **rtinterface** • **rttype** • **rtfamily**

Item	Description
-L	Specifies long format. Produces detailed paragraph-formatted information for each requested workload partition. It is mutually exclusive with the **-c**, **-d**, **-D**, **-G**, **-I**, **-M**, **-N**, **-q**, **-R**, **-S**, or **-T** flag. If you want to parse data, do not use the **-L** output. Use the delimiter-separated forms (the **-c** or **-d** flag) for generating output that can be parsed. Each workload partition has formatted output similar to the following example: `================` `Name - State` `================` `GENERAL` `Type:  {S|A}` `Hostname:  HostnameWPAR` `-Specific Routing:  {yes|no}` `Directory:  Directory` `Start/Stop Script:  /path/to/userScript` `Auto Start:  {yes|no}` `Private /usr:  {yes|no}` `Checkpointable:  {yes|no}` `Application:  /path/to/trackedProcess` `Owner:` `OStype:  <i>Integer value representing operating system type<i>` `Cross-WPAR IPC:  {yes|no}` `Architecture:WPAR compatibility architecture` `UUID:  String value representing universally unique ID` `NETWORK` `Interface  Address  Mask/Prefix  Broadcast` `---` `if  A.B.C.D A.B.C.D  A.B.C.D` `...  ...  ...  ...` `USER-SPECIFIED ROUTES` `Type  Destination  Gateway  Interface` `---` `net|host  destination  gateway  if` `...  ...  ...  ...` `FILESYSTEMS` `MountPoint  Device  Vfs  Nodename  Options` `---` `mountpoint  device  vfs  node  options` `...  ...  ...  ...  ...` (*A long-format example by the* **-L** *flag, is as follows*) `RESOURCE CONTROLS` `Active:  {yes|no}` `RSet:  rset` `CPU Shares:  n` `CPU Limits:  m%-S%,H%` `Memory Shares:  n` `Memory Limits:  m%-S%,H%` `Per Process Virtual Memory Limit:  nMB` `Total Processes:  n` `Total Threads:  n` `Total PTYs:  n` `Total Large Pages:  n` `Max Message queue IDs:  n%` `Max Semaphore IDs:  n%` `Max Shared memory IDs:  n%` `Max Pinned memory:  n%` `OPERATION` `Operation:  %c` `Process ID: %p` `Start time: %t` `SECURITY SETTINGS` `Privileges: privilege list` `…` `DEVICE EXPORTS` `Name  Type` `Virtual Device  RootVG  Status` `device name type virtual device name yes/no device status` `…  …`

Item	Description
-M	Produces detailed mount information for each requested workload partition. The file systems that are mounted from outside the workload partition are listed and the file systems that are defined within the workload partition are not included. The **-M** flag is mutually exclusive with the **-G**, **-I**, **-L**, **-N**, **-R**, or **-T** flag. If you do not specify the **-c** or **-d** flag, tabular output is produced as shown in the following example: `Name MountPoint Device Vfs Nodename Options` `---` `name mountpoint device vfs node options` `... ... ... ... ... ...` With the **-c** or **-d** flag, delimited output is produced as shown in the following example: `name:mountpoint:device:vfs:nodename:options` It can be used with the **-a** flag to limit the output to any combination of the following fields: • **Name** (the workload partition name) • **Mountpoint** (the mount point name) • **Device** (the object mounted) • **Vfs** (the virtual-file-system type) • **Nodename** (node name, if the mount is remote) • **Options** (any mount options)
-N	Produces detailed network information for each requested workload partition. It is mutually exclusive with the **-G**, **-I**, **-L**, **-M**, **-R**, **-D**, **-S**, or **-T** flag. If you do not specify the **-c** or **-d** flag, tabular output is produced as shown in the following example: `Name Interface Address(6) Mask/Prefix Broadcast` `---` *`name if A.B.C.D A.B.C.D A.B.C.D`* `...` *`name if S:T:U:V:W:X:Y:Z R`* `...` With the **-c** or **-d** flag, delimited output is produced as shown in the following example: `name:interface:address:mask_prefix:broadcast` You can specify the **-N** flag with the **-a** flag to limit the output to any combination of the following fields: • **Name** (the WPAR name) • **Interface** • **Address** (the IPv4 or IPv6 address) • **Mask_Prefix** (the IPv4 **netmask** field or the IPv6 **prefixlen** field) • **Broadcast** If a WPAR contains 1 or more name-mapped interfaces, the **lswpar** command shows only the information that is specified in the configuration file when the WPAR is in the **Defined** state. When the WPAR is in the **Active** state, the actual runtime network attributes are displayed. **Note:** When a delimited output is expected to contain IPv6 addresses, use the **-d** flag to specify an alternative delimiter because IPv6 addresses contain colons.
-q	Suppresses table headers (quiet). It is valid only for tabular and delimited output formats.
-R	Produces detailed resource control information for each requested WPAR. It is mutually exclusive with the **-G**, **-I**, **-L**, **-M**, **-N**, **-D**, **-S**, or **-T** flag. If you do not specify the **-c** or **-d** flag, each workload partition output has the following paragraph format: `================` *`Name - State`* `================` `Active: {yes|no}` `RSet: rset` `CPU Shares: n` `CPU Limits: m%-S%,H%` `Memory Shares: n` `Memory Limits: m%-S%,H%` `Per-Process Virtual Memory Limit: n MB` `Total Processes: n` `Total Threads: n` `Total PTYs: n` `Total Large Pages: n` `Max Message queue IDs: n%` `Max Semaphore IDs: n%` `Max Shared memory IDs: n%` `Max Pinned memory: n%` With the **-c** or **-d** flag, delimited output is as shown in the following example: `name:state:active:rset:shares_CPU:CPU:shares_memory:memory:` `procVirtMem:totalProcesses:totalThreads:totalPTYs:` `totalLargePages:pct_msgIDs:pct_semIDs:pct_shmIDs:pct_pinMem`
-s {[D] [L] [A] [F] [P] [N] [M] [T] [B]}	Filters the output based on workload partition states. You can use more than 1 state code. See the **-c** flag for a description of the state codes.
-S	Produces detailed security privilege information for each requested WPAR. It is mutually exclusive with the **-D**, **-G**, **-I**, **-L**, **-M**, **-N**, **-R**, or **-T** flag. If you do not specify the **-c** or **-d** flag, each workload partition output has the following paragraph format: `================================` *`Name - State`* `================================` `Privileges:` *`comma-separated list of privileges assigned to the workload partition`*
-t {[A][S][C][L]}	Filters the output based on workload partition types. You can use more than 1 type code. See the **-c** flag for a description of the type codes.

Item	Description
-T	Produces detailed locking information for each requested workload partition. This flag is mutually exclusive with the **-D**, **-G**, **-I**, **-L**, **-M**, **-N**, **-R**, **-S**, or **-s** flag. It is mutually exclusive with the **-q** flag unless the **-c** flag is also specified. If you do not specify the **-c** flag, each workload partition output has the following format: `================` *Name - State* `================` `Operation:  %c` `Process ID: %p` `Start time: %t` With the **-c** or **-d** flag, the output is as shown in the following example: `name:state:opname:oppid:opstart`
-X	Produces detailed kernel extension information for each requested workload partition in turn. It is mutually exclusive with the **-D**, **-G**, **-I**,**-L**, **-M**, **-N**, **-R**, **-S**, or **-T** flag. If the **-c** or **-d** flag is not specified, each workload partition output has the following tabular format: `Name      Extension Name     Local  Major  Status  checksum` `----------------------------------------------------------------` `name   /path/to/extension    local  major  status  checksum` `...`
wparname	Specifies 1 or more workload partitions. It must be last on the command line. It can contain shell-style wildcards to match multiple workload partition names. (In this case, use appropriate shell quotation marks to preclude shell expansion before the **lswpar** command receives the metacharacters.)

Security

Access Control: Only the root user can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To view tabular information about all workload partitions, enter the following command:

```
# lswpar
Name        State  Type   Hostname            Directory     RootVG WPAR
------------------------------------------------------------------------
bar         A      S      bar.austin.ibm.com  /wpars/bar      yes
foo         D      S      foo.austin.ibm.com  /wpars/foo      no
trigger     A      A      trigger             /
```

2. To view limited tabular information about application workload partitions only, enter the following command:

```
# lswpar -t A -a name,application,script
Name     Application                Script
-------------------------------------------------------------
trigger  /usr/sbin/apachectl start  /home/joe/trigger.script
```

3. To view colon-separated general information with no headers for all active and defined workload partitions, enter the following command:

```
# lswpar -G -c -q -s AD
bar:A:S:bar.austin.ibm.com:/wpars/bar:/home/bar/wpar.scr:no:no:yes::no
foo:D:S:foo.austin.ibm.com:/wpars/foo::no:no:no::no
trigger:A:A:trigger:/:/home/joe/trigger.script
:no:no:yes:/usr/sbin/apachectl start:no
```

4. To view extended information about the workload partition named `trigger`, enter the following command:

```
 # lswpar -L trigger
================
trigger - Active
================
GENERAL
Type:                   A
Hostname:               triggerWPAR
-Specific Routing:  yes
Directory:              /
Start/Stop Script:      /home/joe/trigger.script
Auto Start:             no
Private /usr:           no
```

```
Checkpointable:         yes
Application:            /usr/sbin/apachectl start

NETWORK
Interface    Address          Mask/Prefix      Broadcast
------------------------------------------------------------------
en0          1.2.3.4          255.255.255.0    1.2.3.255
en1          5.6.7.8          255.255.255.0    5.6.7.255

USER-SPECIFIED ROUTES
Type        Destination   Gateway     Interface
------------------------------------------------------------
net         9.1.2.24      1.2.3.1     en0
host        192.168.1.2   1.2.3.1     en1

FILESYSTEMS
MountPoint              Device          Vfs    Nodename  Options
------------------------------------------------------------------
/share                  /nfs2/share     nfs    nfsserver rw

RESOURCE CONTROLS
Active:                              yes
RSet:                                isp1
CPU Shares:                          2
CPU Limits:                          5%-10%,50%
Memory Shares:                       3
Memory Limits:                       10%-20%,30%
Per-Process Virtual Memory Limit:    1024MB
Total Processes:                     64
Total Threads:                       1024
Total PTYs:                          8
Total Large Pages:                   16
Max Message queue IDs:               20%
Max Semaphore IDs:                   30%
Max Shared memory IDs:               50%
Max Pinned memory:                   20%
OPERATION:
Operation:                           restart
Process ID:                          905266
Start time:                          11:19


Privileges:                          PV_AU_,PV_AU_ADD,PV_AU_ADMIN,PV_AU_PROC,
                                     PV_AU_READ,PV_AU_WRITE,PV_AZ_ADMIN,
                                     PV_AZ_CHECK,PV_AZ_READ,PV_AZ_ROOT,PV_DAC_,
                                     PV_DAC_GID,PV_DAC_O,PV_DAC_R,PV_DAC_RID,
                                     PV_DAC_UID,PV_DAC_W,PV_DAC_X,PV_DEV_CONFIG,
                                     PV_DEV_QUERY,PV_FS_CHOWN,PV_FS_CHROOT

DEVICE EXPORTS
Name                                 Type         Virtual Device    RootVG      Status
-------------------------------------------------------------------------------------
hdisk4                               disk                           yes         ALLOCATED
/dev/null                            pseudo                                     ALLOCATED
/dev/tty                             pseudo                                     ALLOCATED
/dev/random                          pseudo                                     ALLOCATED
/dev/urandom                         pseudo                                     ALLOCATED
/dev/console                         pseudo                                     ALLOCATED
/dev/zero                            pseudo                                     ALLOCATED
/dev/clone                           pseudo                                     ALLOCATED
/dev/sad                             clone                                      ALLOCATED
```

5. To view machine-readable network information that is separated by pipes for workload partitions called `roy`, enter the following command:

```
# lswpar -d'|' -N roy
#name|interface|address|mask_prefix|broadcast
roy|en0|192.168.1.50|255.255.255.128|192.168.1.127
roy|en1|2001:DB8::|32|
```

6. To view machine-readable, resource-control information for all workload partitions, enter the following command:

```
# lswpar -cR
#name:state:active:rset:shares_CPU:CPU:shares_memory:memory:procVirtMem:
totalProcesses:totalThreads:totalPTYs:
totalLargePages:pct_msgIDs:pct_semIDs:pct_shmIDs:pct_pinMem
dale:A:no:::::::
roy:A:yes:rogers:3::2::32:128
trigger:A:yes:isp1:2:5%-10%,50%:3:10%-20%,30%:1024MB:64:1024:8:
16:20%:30%:50%:20%
```

7. To view operation information about the workload partition named `foo`, enter the following command:

```
# lswpar –T foo
================================================================
foo - Transitional
================================================================
Operation: restart
Process ID: 905266
Start time: 11:19
```

8. To view information about devices that are exported and allocated in the workload partitions named `roy`, enter the following command:

9.
```
# lswpar -D roy
Name  Device Name      Type    Virtual Device  RootVG    Status
-----------------------------------------------------------------
roy   /dev/null       pseudo                             EXPORTED
...
roy   fcs0            adapter                          EXPORTED
roy   hdisk2          disk      hdisk0          yes
```

10. To view information about bootset of a RootVG workload partition, enter the following command:

```
lswpar -Br <WPAR name>
```

11. To view information about bootlist of a workload partition, enter the following command:

```
lswpar -b <WPAR name>
```

luit Command

Purpose

Supports locale and ISO 2022 for Unicode terminals.

Syntax

luit [options] [--] [program [args]]

Description

The **luit** command is a filter that runs between an arbitrary application and a UTF-8 terminal emulator. The **luit** command converts application output from the locale's encoding into UTF-8, and converts terminal input from UTF-8 into the locale's encoding.

Note: Multilingual applications must be set to generate only the UTF-8 code. You must not use the command to use different output other than UTF-8.

The **luit** command is invoked transparently by the terminal emulator. For information on running the **luit** command from the command line, see Examples.

Options

Item	Description
-h	Displays the help summary.
-list	Lists the supported **charsets** and **encodings**.
-v	Verbose.
-c	Converts the standard input to standard output.
-x	Exits as soon as the child function dies. This might cause the **luit** command to loose data at the end of the output of the child function.
-argv0 *name*	Sets the name of the child that is passed along with the **argv[0]** command.
-encoding *encoding*	Specifies that the **luit** command use encoding rather than the current locale encoding.
+oss	Disables interpretation of single shifts in the application output.
+ols	Disables interpretation of locking shifts in the application output.

Item	Description
+osl	Disables interpretation of character set selection sequences in the application output.
+ot	Disables interpretation of all sequences and passes all sequences in the application output to the terminal unchanged.
-k7	Generates 7-bit characters for keyboard input.
+kss	Disables generation of single-shifts for keyboard input.
+kssgr	Uses GL codes after a single shift for keyboard input. By default, GR codes are generated after a single shift when generating 8-bit keyboard input.
-kls	Generates locking shifts (SO/SI) for keyboard input.
-gl *gn*	Sets the initial assignment of GL. The argument must be one of **g0**, **g1**, **g2**, or **g3**. The default value depends on the locale, and is usually **g0**.
-gr *gk*	Sets the initial assignment of GR. The default value depends on the locale, and is usually **g2** except for EUC locales, where it is **g1**.
-g0 *charset*	Sets the value of **charset** that is initially selected in **G0**. The default value depends on the locale, but is usually ASCII.
-g1 *charset*	Sets the value of **charset** that is initially selected in **G1**. The default value depends on the locale.
-g2 *charset*	Sets the value of **charset** that is initially selected in **G2**. The default value depends on the locale.
-g3 *charset*	Sets the value of **charset** that is initially selected in **G3**. The default value depends on the locale.
-ilog *filename*	Logs all the bytes received from the child into *filename*.
-olog *filename*	Logs all the bytes sent to the terminal emulator into *filename*.

Examples

1. To adapt an instance of XTerm to the locale's encoding, current versions of XTerm invoke the **luit** command automatically when it is needed. If you are using an older release of XTerm, or a different terminal emulator, you can invoke the **luit** command manually:

```
$ xterm -u8 -e luit
```

2. If you are running in a UTF-8 locale but need to access a remote machine that does not support UTF-8, the **luit** command can adapt the remote output to your terminal:

```
$ LC_ALL=fr_FR luit ssh legacy-machine
```

Files

Item	Description
`/usr/lib/X11/fonts/encodings/encodings.dir`	Contains the system-wide encoding directory.
`/usr/lib/X11/locale/locale.alias`	Contains the file mapping locales to locale encoding.

lvmo Command

Purpose

Manages lvm pbuf tunable parameters.

Syntax

`lvmo -v` *Name* `-o` *Tunable* [*=NewValue*]

`lvmo -a` [`-v` *vgname*]

`lvmo -L` [*Tunable*]

Description

The `lvmo` command sets or displays pbuf tuning parameters. The equal sign can be used to set a particular tunable to a given value. Otherwise, if no equal sign is used, the value of the tunable will be displayed.

Attention: Misuse of the **lvmo** command can cause performance degradation or operating-system failure.

The **lvmo -a** command generates pbuf and blocked I/O statistics. The pbuf and blocked I/O report has the following label:

Label	Description
`vgname`	Volume group name specified with the `-v` option.
`pv_pbuf_count`	The number of pbufs that are added when a physical volume is added to the volume group.
`total_vg_pbufs`	Current total number of pbufs available for the volume group.
`max_vg_pbuf_count`	The maximum number of pbufs that can be allocated for the volume group.
`pervg_blocked_io_count`	Number of I/O's that were blocked due to lack of free pbufs for the volume group.
`pv_min_pbuf`	The minimum number of pbufs that are added when a physical volume is added to any volume group.
`global_blocked_io_count`	Number of I/O's that were blocked due to lack of free pbufs for all volume groups.
`aio_cache_pbuf_count`	Current total number of pbufs available for aio_cache logical volume in the volume group.

Flags

Item	Description
`-a`	Displays value for all tunable parameters, one per line in pairs *tunable* = *value*.
`-o` *Tunable* [*=NewValue*]	Displays the value or sets *Tunable* to *NewValue*.

Item	Description
-L [*Tunable*]	Lists the characteristics of one or all of the tunables, one per line, using the following format:

```
NAME                      CUR    DEF    BOOT   MIN    MAX    UNIT   TYPE
     DEPENDENCIES
------------------------------------------------------------------------------
global_blocked_io_count    0      0     n/a     0      0             S
     pervg_blocked_io_count
------------------------------------------------------------------------------
max_vg_pbufs              16384  n/a    n/a   16384    none          S
     max_vg_pbuf_count
     pv_min_pbuf
     pv_pbuf_count
     total_vg_pbufs
------------------------------------------------------------------------------
max_vg_pbuf_count          0      0      n/a    0      none          M
     max_vg_pbufs
     pv_min_pbuf
     pv_pbuf_count
     total_vg_pbufs
------------------------------------------------------------------------------
pervg_blocked_io_count     0      0      n/a    0      0             S
     global_blocked_io_count
------------------------------------------------------------------------------
pv_min_pbuf               512    512     n/a    0      none          D
     max_vg_pbufs
     max_vg_pbuf_count
     pv_pbuf_count
     total_vg_pbufs
------------------------------------------------------------------------------
pv_pbuf_count             512    512    n/a     1      16384         D
     max_vg_pbufs
     max_vg_pbuf_count
     pv_min_pbuf
     total_vg_pbufs
------------------------------------------------------------------------------
total_vg_pbufs            512    n/a    n/a     0      0             S
     max_vg_pbufs
     max_vg_pbuf_count
     pv_min_pbuf
     pv_pbuf_count
------------------------------------------------------------------------------
...
where:
    n/a means parameter not supported by the current platform or kernel
Parameter types:
    S = Static: cannot be changed
    D = Dynamic: can be freely changed
    B = Bosboot: can only be changed using bosboot and reboot
    R = Reboot: can only be changed during reboot
    C = Connect: changes are only effective for future socket connections
    M = Mount: changes are only effective for future mountings
    I = Incremental: can only be incremented
    d = deprecated: deprecated and cannot be changed
Value conventions:
    K = Kilo: 2^10       G = Giga: 2^30       P = Peta: 2^50
    M = Mega: 2^20       T = Tera: 2^40       E = Exa: 2^60
```

Tunable Parameters

Item	Description
pv_pbuf_count	The number of pbufs that is added when a physical volume is added to the volume group.
max_vg_pbuf_count	The maximum number of pbufs that can be allocated for the volume group. **Note**: The volume group must be varied off and varied on again for this value to take effect. This value does not affect rootvg.
pv_min_pbuf	The minimum number of pbufs that is added when a physical volume is added to any volume group. **Note:** Use the **ioo** command to change this value.
aio_cache_pbuf_count	The total number of pbufs that is allocated for aio_cache logical volume in the volume group.
workQ_size	The size of the hash table that is used to track I/O requests for logical volumes in the volume group.

Exit Status

This command returns zero for successful completion; otherwise it returns nonzero.

Security

You must have root authority to run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the value for the pv_pbuf_count, type the following:

```
lvmo -v rootvg -o pv_pbuf_count
```

2. To set the pv_pbuf_count value to 2048, type the following:

```
lvmo -v rootvg -o pv_pbuf_count=2048
```

3. To generate pbuf and blocked I/O statistics, type the following:

```
lvmo -a
```

Location

/usr/sbin/lvmo

lvmstat Command

Purpose

Reports input/output statistics for logical partitions, logical volumes and volume groups. Also reports pbuf and blocked I/O statistics and allows pbuf allocation changes to volume groups.

Syntax

lvmstat { **-l** | **-v** } *Name* [**-e** | **-d**] [**-F**] [**-C**] [**-c** *Count*] [**-s**] [*Interval* [*Iterations*]]

lvmstat -v *Name* **-r** [**-L** | **-C**]

Description

The **lvmstat** command generates reports that can be used to change logical volume configuration to better balance the input/output load between physical disks.

By default, the statistics collection is not enabled in the system. You must use the **-e** flag to enable this feature for the logical volume or volume group in question. Enabling the statistics collection for a volume group enables for all the logical volume in that volume group.

Note: The **-e** flag and the **-d** flag are not applicable for the space reclamation statistics specified by the **-r** flag.

The first report generated by **lvmstat** provides statistics concerning the time since the system was booted. Each subsequent report covers the time since the previous report. All statistics are reported each

time **lvmstat** runs. The report consists of a header row followed by a line of statistics for each logical partition or logical volume depending on the flags specified.

If the **-l** flag is specified, *Name* is the logical volume name, and the statistics are for the physical partitions of this logical volume. The mirror copies of the logical partitions are considered individually for the statistics reporting. They are listed in descending order of number of i/os (iocnt) to the partition.

The *Interval* parameter specifies the amount of time, in seconds, between each report. The first report contains statistics for the time since the volume group startup, **varyonvg**. Each subsequent report contains statistics collected during the interval since the previous report. If the *Count* parameter is specified, only the top *Count* lines of the report are generated. For a logical volume if *Count* is 10, only the 10 busiest partitions are identified. If the *Iterations* parameter is specified in conjunction with the *Interval* parameter, then only that many iterations are run. If no *Iterations* parameter is specified, **lvmstat** generates reports continuously. If *Interval* is used to run lvmstat more than once, no reports are printed if the statistics did not change since the last run. A single period **.** (period) is printed instead.

The **lvmstat** command is useful in determining whether a physical volume is becoming a hindrance to performance by identifying the busiest physical partitions for a logical volume.

Note: The **lvmstat** commands reports I/O statistics of the local node only.

Input/Output Reports

The **lvmstat** command generates two types of reports, per partition statistics in a logical volume and per logical volume statistics in a volume group. The reports have the following format:

Column	Description
Log_part	Logical partition number
mirror#Log_part	Mirror copy number of the logical partition
iocntLog_part	Number of read and write requests
Kb_readLog_part	The total number of kilobytes read
Kb_wrtnLog_part	The total number of kilobytes written
KbpsLog_part	The amount of data transferred in kilobytes per second

Space reclaim statistics reports

The **lvmstat -r** command generates report for space reclaim statistics for physical volumes in the **volume** group. The reports have the following format:

Volume Group	Description
PV_name	Physical volume name
Reclaim	Space reclamation state. The possible state values are: **on** Space reclaim is supported for the physical volume. **off** Space reclaim is not supported for the physical volume. **suspend** Space reclaim is suspended by LVM configuration commands.
Mb_freeed	Amount of physical partition space is freed from logical volume by commands like **rmlv**, **rmlvcopy**, and **chfs** in megabytes
Mb_pending	Space reclamation pending for the physical volume space in megabytes.
Mb_success	Space reclamation requests succeeded at disk driver in megabytes.
Mb_failed	Space reclamation requests failed by the disk driver in megabytes.

Volume Group	Description
Mb_reused	Free physical partition space reused for the logical volume without requesting the space reclamation in megabytes.
Mb_inprog	Amount of space reclaim request outstanding at the disk driver in megabytes.
io_count	Number of space reclaim I/O requests submitted to disk driver.
io_failed	Number of space reclaim I/O requests failed by disk driver.
io_misalign	Number of space reclaim requests reported as misaligned by the disk driver.
Mb_misalign	Amount of space reclaim failed by the disk driver due to misalignment in megabytes.
Mb_resubmit	Amount of space reclaim resubmitted due to reclaim block not being aligned on physical partition blocks.
num_pp_free	Number of physical partitions are freed by LVM commands like **rmlv**, **rmlvcopy**, **chfs**, and so on.
Kb_blksize	Space reclaim block size reported by the disk driver for alignment purpose.

Flags

Item	Description
-c *Count*	Prints only the specified number of lines of statistics.
-C	Causes the counters that keep track of the iocnt, Kb_read and Kb_wrtn be cleared for the specified logical volume/volume group. This flag can also be used to resets the space reclaim statistics.
-d	Specifies that statistics collection should be disabled for the logical volume/volume group in question.
-e	Specifies that statistics collection should be enabled for the logical volume/volume group in question.
-F	Causes the statistics to be printed colon-separated.
-l	Specifies the name of the stanza to list.
-L	Displays the space reclaim statistics in long listing mode.
-r	Prints the space reclaim statistics for all physical volumes in the volume group.
-s	Suppresses the header from the subsequent reports when *Interval* is used.
-v	Specifies that the *Name* specified is the name of the volume group.

Security

To use **lvmstat**, you must have root user authority.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To enable the statistics collection for volume group datavg (all the LVs in datavg are enabled), enter the following command:

```
lvmstat -v datavg -e
```

2. To display the history of all the partitions of logical volume hd2, enter the following command:

```
lvmstat -l hd2
```

3. To display the history of top five logical volumes of volume group uservg, enter the following command:

```
lvmstat -v uservg -c 5
```

4. To display a continuous report at two second intervals for the logical volume ramlv, enter the following command:

```
lvmstat -l ramlv 2
```

5. To display six reports at two second intervals for the volume group rootvg, enter the following command:

```
lvmstat -v rootvg 2 6
```

6. To reset the counters for statistics for all the logical volumes in the volume group uservg, enter the following command:

```
lvmstat -v uservg -C
```

7. To disable statistics collection for datalv, enter the following command:

```
lvmstat -l datalv -d
```

8. To display statistics for space reclamation, enter the following command:

```
lvmstat -v uservg -r
```

9. To display statistics for space reclamation in long listing mode, enter the following command:

```
lvmstat -v uservg -r -L
```

10. To clear the statistics for space reclamation, enter the following command:

```
lvmstat -v uservg -r -C
```

Files

Item	Description
/usr/sbin/lvmstat	Contains the **lvmstat** command.

lvupdateInit Command

Purpose

Manages the list of entries to be added to the `/etc/inittab` file that is used to start the surrogate partition during the AIX Live Update operation.

Syntax

To add an entry to the `/etc/inittab` file in the surrogate partition, use the following syntax:

```
lvupdateInit -a [-i Identifier] { [ Identifier ] : [ RunLevel ] : [ Action ] : [ Command ] }
```

To list the entries to be added to the /etc/inittab file in the surrogate partition, use the following syntax:

```
lvupdateInit -l
```

To remove an entry from the list to be added to the /etc/inittab file in the surrogate partition, use the following syntax:

```
lvupdateInit -r -i Identifier
```

Description

The Live Update operation creates a customized root volume group (rootvg) to start the surrogate partition and prepare it for the Live Update operation. When the surrogate partition is started, it runs commands in a customized /etc/inittab file and other scripts like /etc/rc.tcpip. The /etc/inittab file is customized to exclude the services that are moved from the original partition during the mobility phase.

Some services that are running on the original partition might choose not to be check-pointed and keep running on the original partition. In such cases, this command can be used to add that service to the /etc/inittab file for the surrogate partition so that it is restarted during the Live Update operation. The /etc/liveupdate/lvup_inittab file contains a list of command entries to be added to the customized /etc/inittab file.

Parameters

Item	Description
Identifier	A 14-character parameter that uniquely identifies an object. The *Identifier* must be unique. If the *Identifier* is not unique, the command is unsuccessful.
RunLevel	A 20-character parameter that defines the run levels in which the *Identifier* parameter can be processed. Each process started by the **init** command can be assigned with one or more run levels in which it can be started.
Action	A 20-character parameter that informs the **init** command how to process the *Command* parameter that you specify. Refer to the **mkitab** command for a list of valid actions that are recognized by the **init** command.
Command	A 1024-character field that specifies the shell command to be run.  **Attention:** To avoid possible corruption of system files, the stdin, stdout, and stderr files must be specified in the *Command* parameter with redirection, or they must be explicitly opened by the program being run by the command line.

Flags

Item	Description
-a	Adds an entry to the /etc/liveupdate/lvup_inittab file.
-i *Identifier*	Specifies an identifier for the new entry.
-r *Identifier*	Removes an entry from the /etc/liveupdate/lvup_inittab file.
-l	List the entries in the /etc/liveupdate/lvup_inittab file.

Examples

1. To add an entry to the /etc/inittab file that is used to start the Live Update operation in the surrogate partition, enter the following command:

```
# lvupdateInit -a -i myserver  myserver:2:once:/opt/myapp/start_my_server
```

2. To remove an entry from the /etc/inittab file of the surrogate partition, enter the following command:

```
# lvupdateInit -r -i myserver
```

3. To list the entries to be incorporated into the /etc/inittab file of the surrogate partition, enter the following command:

```
# lvupdateInit -l
```

lvupdateRegKE Command

Purpose

The **lvupdateRegKE** command is used to register a command to be used to load a kernel extension on the surrogate logical partition (LPAR) during the AIX Live Update operation.

Syntax

```
lvupdateRegKE  [ -a kext_path -c command | -r kext_path | -l ]
```

Description

The Live Update operation includes an opportunity to load specific kernel extensions before the workload resumes execution on the updated surrogate LPAR. This command manages the list of kernel extensions to be loaded on the surrogate partition when it is started. This command can be used to add a kernel extension to the list along with loading and configuring the kernel extension. It also provides options to remove a kernel extension from the list or to display the list of kernel extensions. The list is placed in the /etc/liveupdate/lvup_preload_KE file. To be loaded at the start of the surrogate LPAR, a kernel extension must be included in that file, and must be loaded on the original LPAR when the Live Update operation starts.

Note: The command that is to be registered with the **lvupdateRegKE** command must be present in one of the following file systems: /, /var, /usr, /opt, /tmp. Also, the kernel extension to be loaded by this command must be present in one of these file systems.

Parameters

Item	Description
kext_path	A string of up to 1024 characters that specifies the full path of a kernel extension.
command	A string of up to 1024 characters that specifies a command to be used to load a kernel extension on the surrogate LPAR before the applications are resumed.

Flags

Item	Description
-a *kext_path*	Adds a kernel extension to the list of kernel extensions to be loaded during the Live Update operation. **Note:** If the **-a** flag is specified, the **-c** flag is required.
-c *command*	Loads the kernel extension. Command arguments can be included by double-quoting the string.
-l	Lists the set of kernel extensions that are loaded during a Live Update operation and the commands that are specified to load them.
-r *kext_path*	Removes a kernel extension from the list.

Examples

1. To list all the commands that are registered to load kernel extensions during the Live Update operation, enter the following command:

```
# lvupdateRegKE -l
```

2. To register a command to load a kernel extension on the surrogate LPAR during the Live Update operation, enter the following command:

```
# lvupdateRegKE -a /usr/lib/drivers/mykext -c "/opt/myapp/bin/load_kext -x -y"
```

3. To remove a command that was previously registered to load a kernel extension during the Live Update operation, enter the following command:

```
# lvupdateRegKE -r /usr/lib/drivers/mykext
```

lvupdateRegScript Command

Purpose

The **lvupdateRegScript** command is used to register a script to be executed at notification points during an AIX Live Update operation.

Syntax

To register a script, use the following syntax:

```
lvupdateRegScript  -a -n label -s script -d { orig | surr } -P phase -p priority
```

To unregister a script, use the following syntax:

```
lvupdateRegScript  -r -n label -d { orig | surr } -P phase [ -p priority ]
```

To list all registered scripts, use the following syntax:

```
lvupdateRegScript  -l
```

Description

The valid phases that a script can be registered for are as follows:

LVUP_CHECK
Executed at the beginning of a Live Update operation. This phase ensures the executed scripts that any associated services are ready for the Live Update operation.

LVUP_PRE
Executed before the applications are frozen on the original logical partition (LPAR). This phase allows the executed scripts to save any data as required before the applications are frozen.

LVUP_PRE_KERNEL
Executed after the applications are frozen on the original LPAR and before the applications are restarted on the surrogate LPAR. This phase is executed on the original LPAR before it is executed on the surrogate LPAR.

LVUP_POST
Executed on the surrogate LPAR after the applications are restarted. These scripts are executed in the `chroot` environment.

LVUP_ERROR
Executed in case of an error during the LVUP_CHECK phase or any later phase of the Live Update operation.

LVUP_COMPLETE
Executed on the surrogate LPAR after the Live Update operation is complete. This phase includes deletion of the original LPAR and resetting of the surrogate LPAR's UUID. The scripts that are registered for this phase are executed in the `chroot` environment.

During the Live Update operation, before the DR_CHECK scripts are invoked, the scripts that are registered with the LVUP_CHECK phase are executed. For the phases LVUP_CHECK, LVUP_PRE, LVUP_PRE_KERNEL, LVUP_POST, and LVUP_COMPLETE, the scripts are executed in a specific priority order that varies between the original and the surrogate logical partitions. On the original LPAR, the order is from priority 1 to priority 10, and the order is reversed on the surrogate LPAR, executing priority 10 first and on down to priority 1. In case of an LVUP_ERROR event, the scripts are executed in the opposite priority order.

The same methodology is applied to rest of the phases.

The script owner must specify if the script must be run on the original or the surrogate LPAR. If the script is to be run on both the original and the surrogate LPARs, it must be registered twice, once for the original LPAR and once for the surrogate LPAR.

When a registered script is executed, it must return 0 to indicate success, or a non-zero value if it failed. The Live Update operation fails if a script fails during the LVUP_CHECK or LVUP_PRE event.

Note: The script that is to be registered with the **`lvupdateRegScript`** command must be present in one of the following file systems: `/`, `/var`, `/usr`, `/opt`, `/tmp`.

Parameters

Item	Description
label	A string of up to 80 characters that specifies a label, which identifies a particular script.
script	A string of up to 1024 characters that specifies a script to be executed. The string must contain the full path to the script as well as any desired arguments.
phase	A string that specifies one of a set of phases: LVUP_CHECK, LVUP_PRE, LVUP_PRE_KERNEL, LVUP_POST, LVUP_ERROR, and LVUP_COMPLETE.
priority	An integer from 1 to 10 that identifies a priority for executing the script.

Flags

Item	Description
-a	Registers a script to be executed during the Live Update operation.
-d `orig` | `surr`	Specifies the LPAR on which the script is to be executed: original LPAR (`orig`) or surrogate LPAR (`surr`).
-l	Lists the scripts and associated labels that are registered.
-n *label*	Specifies a label to associate with a registered script.
-P *phase*	Selects the phase when the script is to be invoked.
-p *priority*	Specifies a priority from 1 to 10. The scripts that are registered for a particular phase are executed in order from highest (1) to lowest (10) priority.
-r	Unregisters a script.
-s *script*	Specifies the script to be executed.

Examples

1. To list all the notification scripts that are registered to be invoked during the Live Update operation, enter the following command:

```
# lvupdateRegScript -l
```

2. To register a script to execute on the original LPAR during an LVUP_PRE event of the Live Update operation, enter the following command:
3. To register a script to execute on the surrogate LPAR during the LVUP_POST event of the Live Update operation, enter the following command:

```
# lvupdateRegScript -a -n putFiles -s "/opt/myapp/bin/rest_files /var/myapp/data"
                              -P LVUP_POST -p 10 -d surr
```

4. To remove a script that was previously registered to be invoked during the Live Update operation, enter the following command:

```
# lvupdateRegScript -r -n getFiles -d orig -P LVUP_PRE
```

lvupdateSafeKE Command

Purpose

The **lvupdateSafeKE** command is a utility that manipulates the list of safe kernel extensions for the AIX Live Update operation.

Syntax

```
lvupdateSafeKE  [ -a kext_path [-p] | -r kext_path | -l ]
```

Description

During a Live Update operation, a new logical partition (LPAR) is dynamically created and booted with the updated AIX kernel. When the new LPAR is booted, the device drivers are reloaded while the devices are configured. The **lvupdateRegKE** command can be used to specify commands to be executed during a Live Update operation to load other kernel extensions. After the new LPAR is prepared, the active workload from the original LPAR is checkpointed and restarted from the same point on the new LPAR.

By default, any data from the kernel extensions is not checkpointed. If there is no state data in a kernel extension, reloading the kernel extension on the new LPAR does not cause any problems while the applications are checkpointed and restarted. This condition makes a kernel extension Live Update *safe*. A kernel extension can be made safe by registering scripts to be executed during the Live Update operation that either capture any necessary state information and restore it on the new LPAR, or quiesce any subsystems that are necessary to ensure that there is no state data in the kernel extension.

A kernel extension that is in the safe kernel list ensures that if it is loaded, it does not prevent a Live Update operation. If a Live Update operation fails because of a loaded kernel extension that is not in the safe list, an error with the kernel extension name is logged under the `/var/adm/ras/liveupdate/logs` directory. Kernel extensions can also be marked safe by specifying the **SYS_LUSAFE** flag when you load the extension by using the `sysconfig()` system call. The **lvupdateSafeKE** command does not list the extensions that were marked safe by using this method.

To bypass the check for safe kernel extensions, the Live Update operation must be started with an `lvupdate.data` file that has the entry, `kext_check = no`.

Note: Any kernel extension that is to be loaded during the Live Update operation must be in one of the following five file systems: `/`, `/var`, `/usr`, `/opt`, `/tmp`.

Parameters

Item	Description
kext_path	A string of up to 1024 characters that specifies the full path of a kernel extension.

Flags

Item	Description
-a *kext_path*	Adds a kernel extension to the list that indicates that it is safe for the Live Update operations.
-l	Lists the set of kernel extensions that are specified as safe for the Live Update operation.
-p	Specify the *-p* flag with the *-a* flag to indicate that the kernel extension can be ignored while checking preview. However, do not use this flag during the Live Update operation.
-r *kext_path*	Removes a kernel extension from the list.

Examples

1. To add a kernel extension to the safe list for the Live Update operation, enter the following command:

```
# lvupdateSafeKE -a /usr/lib/drivers/mydev_driver
```

2. To list the kernel extensions that are registered as safe, enter the following command:

```
# lvupdateSafeKE -l
```

3. To remove a kernel extension from the safe list, enter the following command:

```
# lvupdateSafeKE -r /usr/lib/drivers/mydev_driver
```

lvupdateSetProcs Command

Purpose

To add, remove, or list entries in the base process list that is used for the AIX Live Update operations.

Syntax

```
lvupdateSetProcs -b [ -n label -a command | -n label -r | -l ]
```

Description

Base processes are not checkpointed during a Live Update operation. These processes are left unchanged on the original logical partition (LPAR), rather than being migrated to the surrogate LPAR. This command provides a mechanism to manage the list of base processes. The list of base processes is placed in the `/etc/liveupdate/lvup_BaseProcs` file.

Parameters

Item	Description
command	A string of up to 1024 characters that specifies an executable file, which includes a full path.
label	A string of up to 80 characters that specifies a label, which is associated with a particular command that executes as a base process.

Flags

Item	Description
-a *command*	Adds the command to the specified process list. The *command* specified with the **-a** flag must execute as a direct child of the `init` process.
-b	Designates the **lvupdateSetProcs** command to work on the base process list.
-l	Lists the commands from the specified process list.
-n *label*	Specifies a label to associate with a command that is being added or removed.
-r *command*	Removes the command from the specified process list.

Examples

1. To add an entry to the base process list that is used by the Live Update operation, enter the following command:

   ```
   # lvupdateSetProcs -b -n myserv -a /usr/sbin/myservice
   ```

2. To list the commands that are registered as base processes, enter the following command:

   ```
   # lvupdateSetProcs -bl
   ```

m

The following AIX commands begin with the letter *m*.

m4 Command

Purpose

Preprocesses files, expanding macro definitions.

Syntax

m4 [**-e**] [**-l**] [**-s**] [**-B** *Number*] [**-D** *Name* [**=***Value*]] ... [**-H** *Number*] [**-I** *Directory*] [**-S** *Number*] [**-T** *Number*] [**-U** *Name*] ... [*File* ...]

Description

The **m4** command is a macro processor used as a preprocessor for C and other languages. You can use it to process built-in macros or user-defined macros.

Each *File* parameter is processed in order. If you do not specify a *File* parameter or if you specify the **-** (dash) as a file name, the **m4** command reads standard input. It writes the processed macros to standard output. Macro calls follow the form:

```
macroname(argument . . . )
```

The left parenthesis must immediately follow *macroname*. If the left parenthesis does not follow the name of a defined macro, the **m4** command reads it as a macro call with no arguments. Macro names consist of ASCII alphabetic letters, digits, and the _ (underscore) character. Extended characters are not allowed in macro names. The first character cannot be a digit.

While collecting arguments, the **m4** command ignores unquoted leading blanks, tabs, and new-line characters. Use single quotation marks to quote strings. The value of a quoted string is the string with the quotation marks stripped off.

When the **m4** command recognizes a macro, it collects arguments by searching for a matching right parenthesis. If you supply fewer arguments than those that appear in the macro definition, the **m4** command considers the trailing arguments in the definition to be null. Macro evaluation proceeds normally during the collection of arguments. All commas or right parentheses within the value of a nested macro call are translated literally; they do not need an escape character or quotation marks. After collecting arguments, the **m4** command pushes the value of the macro back onto the input stream and scans again.

Built-in Macros

The **m4** command makes available the following built-in macros. You may redefine them, but you will lose the original meaning. The values of these macros are null unless otherwise stated:

Item	Description
define(*Name*,*NewName*)	Replaces the macro *Name* with the value of *NewName*. The *NewName* string can take the form $*n* . . . (where *n* is a digit). In this case, each occurrence of *n* in the replacement text is replaced by the *n*th argument of *Name*. $0 is the name of the macro. The null string replaces missing arguments. The number of arguments replaces **$#**. A comma-separated list of all arguments replaces $*. $@ acts like $*, but each argument is quoted with the current quotation character (see **changequote**).
undefine(*Name*)	Removes the definition of *Name*.
defn(*Name* . . .)	Returns the quoted definition of *Name*.
pushdef(*Name***,** *NewName***)**	Redefines *Name* with *NewName* as in **define**, but saves any previous definition.
popdef(*Name* . . .)	Removes the current definition of *Name* and returns to the previous definition, if one existed.
ifdef(*Name*,*True*,[*False*])	Returns the value of *True* only if *Name* is defined, otherwise returns *False*. If you do not supply *False*, its value is null. **Note:** The behavior of **ifdef** has changed to comply with the Single UNIX Specification, Version 10. The previous behavior of **ifdef** returns the value of *True* only if *Name* is defined and is not defined as 0. By default, **ifdef** works like it did before UNIX10. The UNIX 10 behavior can be obtained by setting the environment variables *XPG_SUS_ENV* to *ON* and *XPG_UNIX98* to *OFF*.
shift(*Argument* . . .)	Returns all but the first argument. The other arguments are quoted and pushed back with commas in between. The quoting nullifies the effect of the extra scan that is subsequently performed.
changequote(*L*,*R*)	Changes quote symbols to *L* and *R*. The symbols can be up to 5 bytes long. **changequote** without arguments restores the original values (` ').
changecom(*L*,*R*)	Changes left and right comment markers from the default # and new-line character to *L* and *R*. With no arguments, the comment mechanism is disabled. With one argument, the left marker becomes the parameter and the right marker becomes a new-line character. With two arguments, both markers are affected. Comment markers can be up to 5 bytes long.
divert(*Number*)	Changes the current output stream to stream *Number*. There are 10 output streams, numbered 0-9. The final output is the concatenation of the streams in numerical order. Initially, stream 0 is the current stream. The **m4** command discards output diverted to a stream other than 0-9.
undivert(*Number* . . .)	Causes immediate output of text from the specified diversions (or all diversions if there is no argument). Text may be undiverted into another diversion. Undiverting discards the diverted text.
divnum	Returns the value of the current output stream.
dnl	Reads and discards characters up to and including the next new-line character.

Item	Description
ifelse([*String1*,*String2*,*True*, [*False*]] . . .)	If *String1* and *String2* are the same then the value is *True*. If they are not and if there are more than four arguments, the **m4** command repeats the process with the additional arguments (4, 5, 6, and 7). Otherwise, the value is either *False* or null if you provide no value for *False*.
incr(*Number*)	Returns the value of its argument incremented by 1.
decr(*Number*)	Returns the value of its argument decreased by 1.
eval(*Expression*[,*Number1*[,*Number2*]])	Evaluates its first argument as an arithmetic expression, using 32-bit signed arithmetic. The operators you can use are +, -, *, /,%, ^ (exponentiation), bitwise &, | , ~, and ^ relationals, and parentheses. Octal and hex numbers can be specified as in C. *Number1* specifies the radix for the result of the expression. The default radix is 10. The optional *Number2* specifies the minimum number of digits in the result. **Note:** The behavior of **eval** has changed to comply with the Single UNIX Specification, Version 10. The previous behavior of **eval** evaluates its first argument as an arithmetic expression, using 32-bit unsigned arithmetic. By default, **eval** works like it did before UNIX 10. The UNIX 10 behavior can be obtained by setting the environment variables *XPG_SUS_ENV* to *ON* and *XPG_UNIX98* to *OFF*.
len(*String*)	Returns the number of bytes in *String*.
dlen(*String*)	Returns the number of displayable characters in *String*; that is, two-byte extended characters are counted as one displayable character.
index(*String1*,*String2*)	Returns the position in the *String1* string where the *String2* string begins (zero origin), or -1 if the second parameter does not occur.
substr(*String*,*Position*, [*Number*])	Returns a substring of *String*. The beginning of the substring is selected with *Position*, and *Number* indicates the length of the substring. Without *Number*, the substring includes everything to the end of the first string.
translit(*String*,*From*,*To*)	Transliterates the characters in *String* from the set given by *From* to the set given by *To*. No abbreviations are permitted. Two-byte extended characters are correctly mapped into the corresponding replacement characters.
include(*File*)	Returns the contents of *File* or displays an error message if it cannot access the file.
sinclude(*File*)	Returns the contents of *File*, but it gives no error message if *File* is inaccessible.
syscmd(*Command*)	Runs the *Command*. No value is returned.
sysval	Returns the return code from the last call to **syscmd**.
maketemp(. . . *nnnn* . . .)	Replaces *nnnn* in its argument with the current process ID number.

Item	Description
mkstemp(*template*)	The string in the *template* argument is a file name with at least six trailing X characters. The **mkstemp** macro replaces each X character in the file name with a character from the character set of the portable file name. A file is created with the newly created file name and is closed. If the **mkstemp** macro contains an empty string as the *template* argument, the file is not created by using the newly created file name and the **m4** command writes a diagnostic message to the **standard** output. Also, if the **mkstemp** macro contains an empty string as the template argument, the **m4** command continues to process the input even if the file is not created and returns a non-zero value as the exit status.
m4exit(*Value*)	Exits from **m4** immediately, returning the specified exit *Value* (the default is 0).
m4wrap(*LastMacro*)	Runs *LastMacro* after reading the end-of-file character. For example, `m4wrap (`cleanup ()')` runs the cleanup macro at the end of **m4**.
errprint(*Message*)	Includes *Message* on the diagnostic output file.
dumpdef([*Name* . . .])	Writes to standard output the current names and definitions for the named items or for all if no arguments are provided.
traceon(*Macro*)	Turns on tracing for *Macro*. If none is named, tracing is turned on for all macros.
traceoff(*Macro* . . .)	Turns off trace globally and for any *Macro* specified. Macros specifically traced by **traceon** can be untraced only by specific calls to **traceoff**.

Flags

Item	Description
-B *Number*	Makes the *Number* variable the size of the push-back and parameter collection buffers (the default is 4096).
-e	Operates interactively. Interrupts are ignored and the output is not buffered.
-H *Number*	Makes the *Number* variable the size of the symbol table hash array (the default is 199). The size must be a prime number.
-I *Directory*	(Uppercase i) Searches the *Directory* variable first, then searches the directories on the standard list for include (built-in macro) files with names that do not begin with a **/** (slash).
-l	(Lowercase L) Enables line-numbering output for the assembler (.xline . . .).
-s	Enables the line-sync output for the C preprocessor (#line . . .).
-S *Number*	Makes the *Number* variable the size of the call stack (the default is 100 slots). Macros take three slots, and non-macro arguments take one.
-T *Number*	Makes the *Number* variable the size of the token buffer (the default is 512 bytes).

Note: All flag options can be interspersed with operands.

The preceding flags must appear before any file names and before any **-D** or **-U** flags.

Item	Description
-D *Name*[**=***Value*]	Defines the *Name* variable as the *Value* variable. If the *Value* variable is not specified, the *Name* variable becomes null.
-U *Name*	Undefines a the *Name* variable previously defined with the **-D** flag.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

If the **m4exit** macro is used, the exit value can be specified by the input file.

Examples

To preprocess a C language program with the **m4** command and compile it, enter:

```
m4 prog.m4 > prog.c
cc prog.c
```

Files

Item	Description
/usr/ccs/bin/m4	Contains the **m4** command.

mach Command

Purpose

Displays the processor type of the current host .

Syntax

mach

Description

The **mach** command displays the architecture of the system processor.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Examples

1. To display the processor type of the current host use the **mach** command in the following way:

```
mach
```

Files

Item	Description
/usr/bin/mach	Contains the System V **mach** command.

machstat Command

Purpose

Reports the value of the first 4 bits of the power status register.

Syntax

machstat { **-p** | -f }

Description

The **machstat** command returns the value of a status register. There is no standard output or error except when using the -f flag on CHRP hardware.

Flags

Item	Description
-f	On non-CHRP machines, returns Power Status Register bits 10–13. On CHRP machines, displays the EPOW status, EPOW modifier, and, if present, EPOW version.
-p	Returns the first 4 bits of the power status register.

Exit Status

The **machstat** command returns a value of 255 if an error occurs. Otherwise it returns the value of the register.

Security

Access Control: root only

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To see the current value of the power status register, enter:

```
machstat -p
echo $?
```

Files

Item	Description
/etc/rc.powerfail	Shuts down a system when a power failure is detected

macref Command

Purpose

Produces a cross-reference listing of macro files.

Syntax

macref [**-n**] [**-s**] [**-t**] [**—**] [*File* ...]

Description

The **macref** command reads the named English-language files (which are assumed to be **nroff** or **troff** command input) and produces a cross-referenced listing of the symbols in the input.

The default output is a list of the symbols found in the input, each accompanied by a list of all references to that symbol. The **macref** command lists the symbols alphabetically in the left column, with references following to the right. Each reference is given in the following form:

```
[ [(  NMName ) ]
 MName- ]
 Type  LNumber
[  # ]
```

Generated names are listed under the artificial symbol name ~sym.

Input Parameters

Item	Description
File	Specifies the **nroff** or **troff** file from which the **macref** command produces output containing a list cross-referencing macros.

Output Parameters

Item	Description
NMName	The name of the macro within which MName is defined.
MName	The name of the macro within which the reference occurs. This field is not present if the reference occurs outside a macro.

Item	Description
Type	The type associated, by context, with this occurrence of the symbol. The types can be the following: **r** Request **m** Macro **d** Diversion **s** String **n** Number register **p** Parameter. For instance, \$x is a parameter reference to x. **Note:** Parameters are never modified, and the only valid parameter symbol names are 1, 2, . . . 9.
LNumber	The line number on which the reference occurred.
#	This reference modifies the value of the symbol.

Flags

Item	Description
-n	Causes one line to be printed for each reference to a symbol.
-s	Causes symbol-use statistics to be printed.
-t	Causes a macro table of contents to be printed.

The flags can be grouped behind one - (minus sign). Use a — (dash) to delimit the end of flags.

Note: The **macref** command does not accept - as standard input.

Files

Item	Description
/tmp/macref.t*XXXXXX*	Contains a temporary file.
/tmp/macref.s*XXXXXX*	Contains a temporary file.
/tmp/macref.c*XXXXXX*	Contains a temporary file.

mail, Mail, or mailx Command

Purpose

Sends and receives mail.

Syntax

To Read Incoming Mail

mail -e

mail -f [**-dlHNn**] [**-F**] [*FileName*]

mail [**-dlHNn**] [**-F**] [**-u** *UserID*]

To Send Mail

mail [**-s** *Subject*] [**-c** *Address(es)*] [**-dinNv**] *Address*

Description

The **mail** command invokes the mail utility, enabling you to:

- Read incoming mail.
- Send mail.

In addition, you can use the available options and subcommands to customize the way you send and receive mail.

The **mail** command operates on two types of mailboxes, the system mailbox and the personal mailbox.

Incoming mail is stored in the system mailbox. By default, a user's system mailbox is a file located in the /**var**/**spool**/**mail** directory. The mailbox file is named after the userID. For example, if your user ID is *jeanne*, then your system mailbox is /**var**/**spool**/**mail**/**jeanne**.

By default, when a user has read, deleted, or saved all the mail in their system mailbox, the mailbox is deleted. To prevent the mailbox from being deleted, use the **set** subcommand to set the **keep** option.

In addition to the system mailbox, there is the user's personal mailbox. As messages are read, if they are not deleted or saved to a file, they will be marked to be moved to the personal mailbox. The personal mailbox, by default, is **$HOME/mbox**. For example, if your home directory is **/home/lance**, then **/home/lance/mbox** is your personal mailbox. The messages remain in your personal mailbox until you move them to a folder or delete them.

Folders provide a way to save messages in an organized fashion. You can create as many folders as you need. Name each folder with a name that pertains to the subject matter of the messages it contains.

Notes:

- Results can be unpredictable when you run multiple instances of the **mail** command on one mailbox.
- Although the command names are different, the **mail**, **Mail**, or **mailx** command provides identical functionality.

Examining the Contents of Your Mailbox

To process your mail, type `mail` at the system prompt. The Mail program displays a one-line entry for each piece of mail in your system mailbox:

```
Mail [5.2 UCB] [AIX 7.1] Type ? for help.
"/var/spool/mail/lance": 2 messages 2 new
>N 1 karen         Thu Sep 17 14:36 13/359 "Dept Meeting"
 N 2 lance@zeus    Thu Sep 17 15:06 10/350 "Delay"
 N 3 karen         Thu Sep 17 14:36 13/359 "Meeting Cancel"
```

The current message is marked by a > at the beginning of a line in the header summary.

Each one-line entry displays the following fields:

Item description of mailbox

Item	Description
status	Indicates the current class of a piece of mail. The status can be any of the following: **N** A new message **P** A message to be preserved in system mailbox. **U** An unread message. An unread message is a message that was listed in the mailbox last time you invoked the Mail program, but whose contents you did not examine. ***** A message that was saved or written to a file or folder. A message without a status indicates that the message has been read but has not been deleted or saved.
number	Identifies the numerical order of the message.
sender	Identifies the address of the person who sent the mail.
date	Specifies the date the message was received.
size	Defines the number of lines and characters contained in the letter (this includes the header).
subject	Identifies the subject of the message.

Finally, following the list of mail, the Mail program displays the mailbox prompt, which by default is **?**, to indicate that it is waiting for input.

Flags

Flags Description

Item	Description
-c *Address(es)*	Specifies the list of users to which a copy of the message is sent. You can specify one or more addresses. When specifying more than one address, the list of addresses must be in (" ") quotes.
-d	Specifies the debug information associated with the users mailbox • **uid** • **user name** • **mail file folder** (the system mailbox) • **dead letter** (the system saves incomplete messages in the **dead.letter** file in the **$HOME** • **mbox** (the personal mailbox) **Note:** The message is not sent when the program is in the debug mode.
-e	Tests for the presence of mail in the system mailbox. The **mail** utility will write nothing and exit with a successful return code if there is mail to read.
-f *FileName*	Reads messages from the named file. If a file operand is not specified, then reads messages from **mbox**. When you quit from reading the messages, undeleted messages are written back to this file.

Flags Description *(continued)*

Item	Description
-F	Records the message in a file named after the recipient. The name is the portion of the address found first on the **To:** line in the mail header. Overrides the **record** variable if set.
-H	Writes a header summary only.
-i	Causes tty interrupt signals to be ignored.
-n	Inhibits reading the **/usr/share/lib/Mail.rc** file.
`-l`	Expands the `From User` field to 256 characters to handle the long user names.
-N	Suppresses the initial printing of headers.
-s *Subject*	Specifies a subject for a message to be created.
-u *UserID*	Specifies an abbreviated equivalent of doing **mail -f /var/spool/mail/***UserID*. Starts the Mail program for a specified user's mailbox. You must have access permission to the specified mailbox.
-v	Puts the Mail program into verbose mode. Displays the details of delivery on the user's terminal.

Environmental Variables

The following environment variables affect the execution of mail:

Description of environment variables

Item	Description
DEAD	Pathname of the file in which to save partial messages in case of interrupts or delivery errors.
EDITOR	Pathname of the editor to use when the **edit** or **~e** command is used.
HOME	Pathname of the user's home directory.
LISTER	String representing the command for writing the contents of the folder directory to standard output when the **folders** command is given. Any string acceptable as a command_string operand to the **sh -c** command is valid. If this variable is null or not set, the output command will be *ls*. The default value is unset.
MAILBOX	Specifies the location of the system mailbox for the **mail** command. The **MAILBOX** value is where the **mail** command searches for mail messages. The system default value if the **MAILBOX** environment variable is not specified is the **/var/spool/mail** directory.
MAILRC	Pathname of your personal startup file. The default is **$HOME/.mailrc**.
MBOX	Pathname of your personal mailbox where messages are saved from the system mailbox that have been read. The **exit** command overrides this function, as will saving the message explicitly in another file. The default is **$HOME/mbox**.
PAGER	String representing an output filtering or pagination command for writing the output to the terminal. Any string acceptable as a command_string operand to the **sh-c** command is valid. When standard output is a terminal device, the message output will be piped through the command if the mail internal variable **crt** is set to a value less the number of lines in the message. If the **PAGER** variable is null or not set, the paginator is the **pg** shell command.
SHELL	Pathname of a preferred command interpreter.
VISUAL	Pathname of a utility to invoke when the **visual** command or **~v** command-escape is used. If this variable is not set, the full screen editor will be *vi*.

Internal Variables in Mail

Internal variables in a mail

Item	Description
allnet	Treats all network names, whose login name components match, identically. Causes the **msglist** message specifications to behave similarly. The default is **noallnet**.
append	Adds the message saved in your mailbox to the end, rather than the beginning, of the **$HOME/mbox** file. The default is **noappend**.
ask, asksub	Prompts for the subject of each message if not specified on the command line with the **-s** option. If you do not wish to create a subject field, press Enter at the prompt. It is not possible to set both **ask** and **noasksub**, or **noask** and **asksub**. The default is **asksub**.
askbcc	Prompts for the addresses of people to add to the blind copy list. If you do not wish to send blind copies, press Enter at the prompt.
askcc	Prompts for the addresses of people who should receive copies of the message. If you do not wish to send copies, press Enter at the prompt.
autoprint	Sets the **delete** subcommand to delete the current message and display the next message.
crt	Specifies the minimum number of lines that a message must contain before any output filtering or pagination is used when the message is displayed.
debug	Displays debugging information. Messages are not sent while in debug mode. This is the same as specifying the **-d** flag on the command line.
dot	Interprets a period entered on a line by itself as the end of a message you are sending.
escape=*c*	Sets the command escape character to be the character *c*. By default the command escape character is ~ (tilde).
Replyall, flipr	Reverses the meanings of the **Respond** and **respond** or **Reply** and **reply** commands. The default is **noflipr**.
folder=*directory*	The directory name in which to store mail folders. After the directory is defined, you can use the + (plus sign) notation to refer to it when using the *FileName* parameter with **mailbox** subcommands.
header	Enables writing of the header summary when entering mail in receive mode. The default is **header**.
hold	Holds messages that you have read but have not deleted or saved in the system mailbox instead of in your personal mailbox. The default is **nohold**.
ignore	Ignores interrupts while entering messages. Echoes interrupts as @ (at) characters.
ignoreeof	Sets the **mail** command to refuse the Ctrl+D key sequence as the end of a message. Input can be terminated only by entering a period . (period) on a line by itself or by the **~.** command escape. The default is **noignoreeof**.

Internal variables in a mail *(continued)*

Item	Description
indentprefix=*string*	A string that will be prefixed to each line that is inserted into the message by the **~m** command escape. This variable defaults to one tab character.
keep	When a system mailbox, secondary mailbox, or mbox is empty, truncate it to zero length instead of removing it. The default is **nokeep**.
keepsave	Keep messages that have been saved with the **(s)ave** or **(w)rite** subcommands in the system mailbox instead of deleting them. The default is **nokeepsave**.
metoo	Includes the sender in the alias expansion if sender's name is part of the alias. By default, expanding the alias removes the sender.
onehop	When responding to a message that was originally sent to several recipients, the other recipient addresses are usually forced to be relative to the originating author's machine for the response. This flag disables alteration of the recipient's addresses, improving efficiency in a network where all machines can send directly to all other machines (that is, one hop away). The default is **noonehop**.
outfolder	Causes the files used to record outgoing messages to be located in the directory specified by the **folder** variable unless the pathname is absolute. The default is **nooutfolder**. See the **record** and **folder** variables.
page	Insert a form-feed after each message sent through the pipe created by the **pipe** command. The default is **nopage**.
prompt=*string*	Set the command-mode prompt to *string*. If *string* is null or if **noprompt** is set, no prompting will occur. The default is to prompt with the "?" string.
quiet	Refrain from writing the opening message and version when entering mail. The default is **noquiet**.
record=*file*	Defines a file in which to record all outgoing mail. The default is **norecord**.
save	Enables saving of messages in the **dead.letter** file on interrupt or delivery error. The default is **save**.
screen=*number*	Sets the number of lines in a screenful of headers for the **headers** and **z** commands.
sendmail=*shell_command*	Alternative command for delivering messages.
sendwait	Wait for the background mailer to finish before returning. The default is **nosendwait**.
showto	When the sender of the message was the user who is invoking mail, write the information from the **To:** line instead of the **From:** line in the header summary. The default is **noshowto**.
sign=*string*	Inserts *string* into the text of a message when the **~a** command escape is given. The default is **nosign**. The character sequences **/t** and **/n** are recognized in the string as tab and newline characters, respectively.

Internal variables in a mail *(continued)*

Item	Description
Sign=string	Inserts *string* into the text of a message when the **~A** command escape is given. The default is **noSign**.
toplines=number	Number of lines displayed by the **top** subcommand.
verbose	Displays the actual delivery of messages on the terminal. This is the same as specifying the **-v** flag on the command line.

Setting Environment Variables

The Bourne shell (**bsh** command) uses and checks the following variables. These variables can be set in **$HOME/.profile**.

Item descriptions of Bourne shell

Item	Description
MAIL	Specifies the location and name of the user's system mailbox that is checked by the Bourne shell to determine whether or not you have mail. If the system mailbox is not empty, the Bourne shell sends a message that you have new mail. The Bourne shell checks the system mailbox periodically based on the value of the **MAILCHECK** environment variable.
MAILCHECK	Specifies the interval at which the Bourne shell checks the system mailbox for mail.
MAILMSG	Specifies the message sent to your console shell by the system when you have mail. The default message is similar to the following: `YOU HAVE NEW MAIL`

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To start the Mail program and list the messages in your mailbox, type the following at the command line prompt:

   ```
   mail
   ```

   The **mail** command lists every messages in your system mailbox. The mail system then displays the mailbox prompt (?) to indicate *waiting for input*. When you see this prompt, enter any mailbox subcommand.

   To see a list of subcommands, type:

   ```
   ?
   ```

   This entry lists the Mail subcommands.

2. To send the message `letter` to the recipient `user1@host1` and copies to `user2@host2` and `user3@host3`, type:

   ```
   mail -c "user2@host2 user3@host3" user1@host1<letter
   ```

3. To look at the contents of your personal mailbox, type:

```
mail -f
```

This command displays a list of the messages in your personal mailbox, **$HOME/mbox**.

4. To look at the contents of a specific mail folder, type:

```
mail -f +dept
```

This command displays a listing of the messages in the `dept` folder.

5. To send a message to a user on your local system, type:

```
mail ron
```

When you finish typing the message to user `ron`, press Enter and press either . (period) or Ctrl+D to exit the editor and send the message. To determine if a user is on your local system, check for the user's name in your **/etc/passwd** file.

If your message is delivered successfully, you receive no notification. If your message could not be delivered, an error message is sent to you.

6. To mail a file to another user on your local system, type:

```
mail karen < letter1
```

This command sends the contents of the file `letter1` to user `karen` on your local system. After the command sends the file, the Mail program displays the command line prompt.

7. To send a message to a user on a remote system, type:

```
mail dale@zeus
```

You now can create a message to `dale`. In this example, you are sending a message to user `dale` on remote system `zeus`. To send a message to a user on another system connected to your system through a network, you must know that person's login ID and the name of the other system.

Mailbox Subcommands for the mail, Mail, and mailx Command

From the mail prompt, ? (question mark), you can enter subcommands to manipulate mail in your mailbox. Subcommands that work on more than one message at a time use the *MessageList* parameter. Subcommands that work with files or folders use the *FileName* parameter. These parameters are discussed in Mail command and subcommands in *Networks and communication management*.

The following list describes the Mailbox subcommands and their functions:

Mailbox subcommands

Item	Description
=	Echoes the number of the current message.
#	Comment character for writing comments in mail script files.
-	Displays the previous message.
?	Displays a brief summary of mailbox subcommands. Identical to the **help** subcommand.
!*Command*	Executes the workstation shell command specified by *Command*.
alias	(**a**) With no arguments, displays all currently defined aliases and their corresponding addresses. With one argument, displays one alias. With more than one argument, creates a new alias or changes an old alias. Identical to the **group** subcommand.

Mailbox subcommands *(continued)*

Item	Description
alternates *AlternatesList*	(**alt**) The **alternates** subcommand is useful if you have accounts on several machines. Use the subcommand to inform the Mail program that the addresses listed in *AlternatesList* all refer to you. When you use the **reply** subcommand to reply to messages, the Mail program does not send a copy of the message to any of the addresses given in *AlternatesList*. If you enter the **alternates** subcommand with no argument, the Mail program displays the current set of alternate names.
chdir *Directory*	(**cd**) Changes your working directory to the indicated *Directory*. If no directory is given, it changes to your login directory.
copy [*MessageList*] *File*	(**c**, **co**) Appends each message in *MessageList* to the end of *File*. Displays the file name in quotes, followed by the line count and character count, on the user's terminal. Does not delete any messages when you quit.
Copy [*MessageList*]	(**C**) Saves the specified message in a file whose name is derived from the author of the message to be saved, without marking the messages as saved. Otherwise, it is equivalent to the **Save** subcommand.
delete [*MessageList*]	(**d**) Marks the messages in *MessageList* to be deleted when you quit the Mail program. Entering the **d** subcommand without a message list deletes the current message. Deleted messages are not saved in your **$HOME/mbox** file nor are they available for most other commands. However, you can use the **undelete** subcommand to restore messages you have deleted while in the same mailbox session. If you delete a message and either change to another mailbox or quit the mailbox with the **quit** subcommand, the deleted message cannot be recalled.
discard [*FieldList*]	(**di**) Identical to the **ignore** subcommand. **Note:** The **retain** subcommand overrides the **discard** subcommand.
dp	Deletes the current message and displays the next message. If there is no next message, the Mail program displays E0F. Identical to the **dt** subcommand.
dt	Deletes the current message and displays the next message. If there is no next message, the Mail program displays E0F. Identical to the **dp** subcommand.
echo *String*	Displays the character string *String* on the command line.
edit [*MessageList*]	(**e**) Starts the alternate editor using the *MessageList* as input files. To define an alternate editor, use the **set EDITOR=** statement or place an entry in your **$HOME/.mailrc** file. Any message specified by the *MessageList* parameter retains the changes made during the editor session.
exit	(**ex** or **x**) Leaves the mailbox and returns to the operating system without changing the original contents of the mailbox. The mailbox returns to the condition that it was when the Mail program was started. Messages marked to be deleted are not deleted. Identical to the **xit** subcommand.
file [*Name*]	(**fi**) Identical to the **folder** subcommand.

Mailbox subcommands *(continued)*

Item	Description
folder [*Name*]	(**fo**) Switches to a new mail file or folder. With no arguments, the subcommand displays the name of the current mailbox. If an argument is included, it stores the current mailbox with changes (such as messages deleted) and reads in the new mailbox specified by the *Name* parameter. Identical to the **file** subcommand. Some special conventions are recognized for the *Name*: **#** Refers to the previous file. **%** Refers to the system mailbox (**/var/spool/mail/***UserID*). **&** Refers to your personal mailbox (**$HOME/mbox**). ***+Name*** Refers to a file in your folder directory.
folders	Lists the names of the folders in your folder directory.
followup [*message*]	(**fo**) Responds to a message, recording the response in a file whose name is derived from the author of the message. Overrides the **record** variable, if set.
Followup [*MessageList*]	(**F**) Responds to the first message in the *msglist*, sending the message to the author of each message in the *msglist*. The subject line is taken from the first message and the response is recorded in a file whose name is derived from the author of the first message.
from [*MessageList*]	(**f**) Displays the headings of messages in *MessageList*.
group	(**g**) Identical to the **alias** subcommand.
headers [*Message*]	(**h**) Lists the headings in the current group of messages (each group of messages contains 20 messages by default; change this with the **set screen=** statement). If the mailbox contains more messages than can be displayed on the screen at one time, information about only the first group of messages will be displayed. To see information about the rest of the messages, use the **h** subcommand with a message number in the next range of messages, or use the **z** subcommand to change the current message group.
help	Displays a brief summary of mailbox subcommands. Identical to the **?** subcommand.
hold [*MessageList*]	(**ho**) Marks each message in *MessageList* to be saved in your system mailbox (**/var/spool/mail/***UserID*) instead of in your **$HOME/mbox** file. Does not override the **delete** subcommand. Identical to the **preserve** subcommand.

Conditional execution of mail subcommands

Item	Description
if *Condition* **elseend if**	Construction for conditional execution of the **mail** subcommands. Subcommands following **if** are executed if *Condition* is true. Subcommands following **else** are executed if *Condition* is not true. The **else** is not required. The **endif** ends the construction and is required. The *Condition* can be receive (receiving mail) or send (sending mail).

Conditional execution of mail subcommands *(continued)*

Item	Description
	ignore [*FieldList*] Adds the header fields in *FieldList* to the list of fields to be ignored. Ignored fields are not displayed when you look at a message with either the **type** or **print** subcommand. Use this subcommand to suppress machine-generated header fields. Use either the **Type** or **Print** subcommand to print a message in its entirety, including ignored fields. The **ignore** subcommand with no arguments lists all header fields that are not included when you use a **type** or **print** subcommand to display a message. Identical to the **discard** subcommand. **list** (**l**) Displays a list of all mailbox subcommands with no explanation of what they do. **mail *AddressList*** (**m**) Starts the mail editor. Enables you to create and send a message to people specified in *AddressList*. The newly created message is independent from any receive messages. **mbox [*MessageList*]** Indicates that the messages in *MessageList* are to be sent to your personal mailbox (**$HOME/mbox**) when you quit the Mail program. This operation is the default action for messages that you have read if you are looking at your system mailbox (**/var/spool/mail/***UserID*) and the **hold** option is not set. **more [*MessageList*]** (**mo**) Displays the messages in *MessageList* using the defined pager program to control display to the screen. Identical to the **page** subcommand. **More [*MessageList*]** (**Mo**) Similar to the **more** subcommand, but also displays ignored header fields. **new [*MessageList*]** Marks each message in *MessageList* as *not* having been read. Identical to the **New**, **unread**, and **Unread** subcommands. **New [*MessageList*]** Marks each message in *MessageList* as *not* having been read. Identical to the **new**, **unread**, and **Unread** subcommands. **next [*Message*]** (**n**) Makes the next message in the mailbox the current message and displays that message. With an argument list, it displays the next matching message. **page [*MessageList*]** (**pa**) Displays the messages in *MessageList* using the defined pager program to control display to the screen. Identical to the **more** subcommand. **Page [*MessageList*]** (**Pa**) Similar to the **page** subcommand but also displays ignored header fields.

pi command

Item	Description
pipe [[msglist command]] | [[msglist] command]]	(**pi**) Pipe the messages through the given command by invoking the command interpreter specified by **SHELL** with two arguments: **-c** and *command*. The command must be given as a single argument. This can be accomplished by quoting. If no arguments are given, the current message will be piped through the command specified by the value of the **cmd** variable. If the **page** variable is set, a form-feed character will be inserted after each message.
preserve	(**pre**) Identical to the **hold** subcommand.
print [*MessageList*]	(**p**) Displays the text of a specific message. Identical to the **type** subcommand.
Print [*MessageList*]	(**P**) Displays the text of a specific message along with the ignored header fields. Identical to the **Type** subcommand.

Item	Description
quit	(**q**) Leaves the mailbox and returns to the operating system. All messages read, but not deleted or saved are stored in your personal mailbox (**$HOME/mbox**). All messages you have marked to be deleted are removed from the mailbox and cannot be recovered. All messages marked with the **hold** or **preserve** option and messages you have not viewed are saved in the system mailbox (**/var/spool/mail/***UserID*). If the **quit** subcommand is given while editing a mailbox file with the **-f** flag, the edit file is saved with changes. If the edit file cannot be saved, the Mail program does not exit. Use the **exit** subcommand to exit without saving the changes.
reply [*Message*]	(**r**) Allows you to reply to the sender of a message and to all others who receive copies of the message. Identical to the **respond** subcommand.
Reply [*Message*]	(**R**) Allows you to reply to only the sender of a message. Identical to the **Respond** subcommand.
respond [*Message*]	Allows you to reply to the sender of a message and to all others who receive copies of a message. Identical to the **reply** subcommand.
Respond [*Message*]	Allows you to reply to only the sender of a message. Identical to the **Reply** subcommand.
retain [*FieldList*]	Adds the header fields in *FieldList* to the list of fields to be retained. Retained fields are displayed when you look at a message with the **type** subcommand or **print** subcommand. Use this subcommand to define which header fields you want displayed. Use the **Type** or **Print** subcommand to print a message in its entirety, including fields that are not retained. If the **retain** subcommand is executed with no arguments, it lists the current set of retained fields. **Note:** The **retain** subcommand overrides the **discard** subcommand.
save [*File*]	(**s**) Saves the current message including header information to a file or folder. If the file already exists, the message is appended to the file. If *File* is omitted, the message will be saved to the user's **mbox**.
save [*MessageList*] *File*	(**s**) Saves a *MessageList* including heading information to a file or folder. If the file already exists, the *MessageList* is appended to the file. Displays the file name and the size of the file when the operation is complete. If you save a message to a file, that message is not returned to the system mailbox (**/var/spool/mail/***UserID*) nor saved in your personal mailbox (**$HOME/mbox**) when you quit the Mail program.
Save [*MessageList*]	(**S**) Saves the specified messages in a file whose name is derived from the author of the first message. The name of the file is taken to be the author's name with all network addressing stripped off.

Item	Description
set [*OptionList* | *Option*=*Value*...]	(**se**) With no arguments, displays the options that are currently enabled. Otherwise, sets an option as specified. The argument following the **set** command can be either: • An *OptionList* giving the name of a binary option (an option that is either set or unset) • An *Option*=*Value* entry used to assign a value to an option. The options are listed in the **.mailrc** file format. **Note:** The form **unset name** is equivalent to **noname**.
shell	(**sh**) Starts an interactive version of the shell.
size [*MessageList*]	Displays the sizes in lines/characters of the messages in *MessageList*.
source *File*	(**so**) Reads and executes the **mail** subcommands from *File*.
top [*MessageList*]	Displays the top few lines of the messages specified by *MessageList*. The number of lines displayed is determined by the valued option **toplines** and defaults to five.
touch [*MessageList*]	Within your system mailbox (**/var/spool/mail/***UserID*), this subcommand marks the messages in *MessageList* to be moved to your personal mailbox (**$HOME/mbox**) when you quit the Mail program. The messages are moved even though you have not read them. The messages are displayed in your personal mailbox as unread messages. The last message in *MessageList* becomes the current message.
type [*MessageList*]	(**t**) Displays the text of a specific message. Identical to the **print** subcommand.
Type [*MessageList*]	(**T**) Displays the text of a specific message along with the ignored header fields. Identical to the **Print** subcommand.
unalias	Deletes the specified alias names.
undelete [*MessageList*]	(**u**) Removes the messages in *MessageList* from the list of messages to be deleted when you quit the Mail program. Entering the **u** subcommand without a message list recalls the last deleted message.
unread [*MessageList*]	(**U**) Marks each message in *MessageList* as *not* having been read. Identical to the **new**, **New**, and **Unread** subcommands.
Unread [*MessageList*]	Marks each message in *MessageList* as *not* having been read. Identical to the **new**, **New**, and **unread** subcommands.
unset *OptionList*	Disables the values of the options specified in *OptionList*. This action is the inverse of the **set** subcommand. **Note:** The form **unset name** is equivalent to **noname**.
version	(**ve**) Displays the version banner for the Mail program.
visual [*MessageList*]	(**v**) Starts the visual editor using the *MessageList* as the input field. (This editor can be defined with the **set VISUAL=** statement.) Any changes made during the editor session are saved back to the messages in the *MessageList*.

pi command *(continued)*

Item	Description
write [*MessageList*] *File*	(**w**) Saves a message without heading information to a file or folder. Displays the file name and the size of the file when the operation is complete. Does not include message headers in the file.
xit	(**x**) Identical to the **exit** subcommand.
z [**+** | **-**]	Changes the current message group (group of 20 messages) and displays the headings of the messages in that group. If a + or no argument is given, then headings in the next group are shown. If a **-** argument is given, the headings in the previous group are shown.

Mail Editor Subcommands for the mail, Mail Command

By default, the Mail program treats lines beginning with the ~ (tilde) character as subcommands. The following list describes the subcommands used while in the mail editor. The editor recognizes subcommands only if you enter them at the beginning of a new line.

Mail editor subcommands

Item	Description
~?	Displays a summary of the **mail** subcommands.
~!*Command*	The command interpreter specified by **SHELL** will be invoked with two arguments: **-c** and *command*. The standard output of command will be inserted into the message.
~a	Inserts the value of the **sign** variable into the text of the message, followed by a newline character. Identical to **~i sign**.
~A	Inserts the value of the **Sign** variable into the text of the message, followed by a newline character. Identical to **~i Sign**.
~b *AddressList*	Adds names in *AddressList* to the list of addresses to receive blind copies of the message. The **~b** subcommand can only be used to add to, not change or delete, the contents of the Bcc: *List*.
~c *AddressList*	Adds names in *AddressList* to the list of people to receive copies of the message. The **~c** subcommand can only be used to add to, not change or delete, the contents of the Cc: *List*.
~d	Appends the contents of the **dead.letter** file to the end of the message.
~e	Starts the alternate editor using the message text as the input file. (This editor can be defined with the **set EDITOR =** statement in the Bourne shell.) When you exit that editor, you return to the mail editor, where you may add text, or send the message by exiting the Mail program.
~f [*MessageList*]	Includes a *MessageList* in the current message to forward the message to another user. This subcommand reads each message in the *MessageList* and appends it to the current end of the message, but does not indent the appended message. This subcommand is also used to append messages for reference whose margins are too wide to embed with the **~m** subcommand. This subcommand works only if you entered the mail editor from the mailbox prompt using either the **mail**, **reply**, or **Reply** subcommand.
~F [*MessageList*]	Equivalent of the **~f**, except that all headers will be included in the message, regardless of previous **discard**, **ignore**, and **retain** commands.

Mail editor subcommands *(continued)*

Item	Description
~h	Enables you to add or change information in all of the heading fields. The system displays each of the four heading fields, one at a time. You can view the contents of each field and delete or add information to that field. Press the Enter key to save any changes to the field and to display the next field and its contents.
~i *string*	Inserts the value of the named variable, followed by a newline character, into the text of the message. If the string is unset or null, the message will not be changed.
~m [*MessageList*]	Includes a *MessageList* in the current message for reference purposes. This subcommand reads each message in the *MessageList* and appends it to the current end of the message. The included message is indented one tab character from the usual left margin of the message. This subcommand works only if you entered the mail editor from the mailbox prompt using either the **mail**, **reply**, or **Reply** subcommand.
~M [*MessageList*]	Equivalent of the **~m**, except that all headers will be included in the message, regardless of previous **discard**, **ignore**, and **retain** commands.
~p	Displays the entire message, including header information.
~q	Quits the editor without sending the message. Saves the message in the **dead.letter** file in your home directory, unless the **nosave** option is set. The previous contents of the **dead.letter** file are replaced with the partially completed message. **Note:** You can also quit the editor by using the Interrupt (Ctrl+C) key sequence twice.
~r *File*	Reads the contents of a file into the current message.
~s *String*	Changes the subject field to the phrase specified in *String*. You cannot append to the subject field with this subcommand.
~t *AddressList*	Adds the addresses in *AddressList* to the To: field of the message. The **~t** subcommand can only be used to add to, not change or delete, the contents of the To: *List*.
~v	Starts the visual editor using the message text as the input file. This editor can be defined with the **set VISUAL=** statement in the Bourne shell.) When you exit that editor, you return to the mail editor where you may add text to the message, or send the message by exiting the Mail program.
~w *File*	Writes the message to the named file.
~x	Exits as with **~q**, except the message will not be saved in the **dead.letter** file.
~: *Subcommand*	Executes the subcommand specified by *Subcommand* and returns to the mail editor.
~| *Command*	Pipes the message through the command *Command* as a filter. If *Command* gives no output or terminates unusually, it retains the original text of the message. Otherwise, the output of *Command* replaces the current message. The **fmt** command is often used as *Command* to format the message.
~< *file*	Reads the contents of a file into the current message.

Mail editor subcommands *(continued)*

Item	Description
~<!Command	Allows you to run a shell command. The shell runs with the **-c** flag and the **Command** specified. The standard output of **Command** is inserted into the message.
~~	Allows you to use the ~ (tilde) character in a message without it being interpreted as a command prefix. The ~~ key sequence results in only one ~ character being sent in the message.

Files

Files

Item	Description
$HOME/.mailrc	Contains the **mail** subcommands to customize the Mail program for a specific user.
$HOME/mbox	Contains your personal mailbox.
/usr/share/lib/Mail.rc	Contains the file with mail subcommands to change the Mail program for all users on the system.
/var/spool/mail/*	Contains system mailboxes for all users.
/usr/bin/mail	Contains the **mail** command.
/usr/bin/Mail	Contains the **Mail** command.
/usr/bin/mailx	Contains the **mailx** command.

mailq Command

Purpose

Prints the contents of the mail queue.

Syntax

/usr/sbin/mailq [**-v**]

Description

The **mailq** and **MAILQ** commands print a list of messages that are in the mail queue. The first line printed for each message shows:

- The internal identifier used on this host for the message with a possible status character
- The size of the message in bytes
- The date and time the message is accepted into the queue
- the envelope sender of the message

Note: Starting from AIX 7 with 7200-04, the non-root users must run the **mailq -Ac** command instead of the **mailq** command to view the messages in the mail queue.

The second line shows the error message that caused the message to be retained in the in the queue, it is not displayed if the message is being displayed for the first time. The status characters are either:

*****
Indicates the job is being processed

X
Indicates that the load is too high to process the job

-
Indicates that the job is too young to process

The following lines show message recipients, one per line.

The **mailq** command is the same as the **sendmail -bp** command.

Specify the **-v** flag to display message priority.

Flags

Item	Description
-v	Prints verbose information. This adds the priority of the message and a single character indicator (+ or blank) indicating wheather a warning message has been sent on the first line of the message. Additionally, extra lines may be intermixed with the recipents indicating the *controlling user* information; this shows who owns any program that are executed on behalf of this message and the name of the alias this command expanded from, if any.
-Ac	Prints the list of messages that are queued in the `/var/spool/clientmqueue` directory.

Exit Status

The command returns the following exit values:

Item	Description
0	Exits successfully.
>0	An error occurred.

Examples

The **mailq** command prints two types of lists:

- The **mailq** command lists the mail queue as shown in the following example:

```
     Mail Queue (1 request)
---QID---- --Size-- -----Q-Time----- ------Sender/Recipient-----
AA02508        3    Thu Dec 17 10:01            root
            (User unknown)
                                    bad_user
```

- The **mailq -v** command lists the mail queue as follows:

```
     Mail Queue (1 request)
---QID---- --Size-- -Priority- ---Q-Time--- --Sender/Recipient--
AA02508        3      1005     Dec 17 10:01        root
            (User unknown)
                                           bad_user
```

- The **mailq -Ac** command lists the `clientmqueue` mail queue as follows:

```
     /var/spool/clientmqueue (1 request)
-----Q-ID----- --Size-- -----Q-Time----- ------------Sender/Recipient-----------
00FGNTRH12845482  5       Wed Jan 15 10:23                root
            (Deferred: Connection refused by [127.0.0.1])
             unknown-user
Total requests: 1
```

The fields have the following meanings:

Item	Description
QID	Contains the message queue ID of the message.
Size	Contains the number of bytes in the body of the message (heading information not included).
Priority	Contains the priority of the message, based primarily on the size of the message.
Q-Time	Contains the time the message entered the queue.
Sender/Recipient	Contains the user ID of the sender and the recipient of the message. A message on the line between the sender and the recipient indicates the status of the message.

Files

Item	Description
/usr/sbin/mailq	Contains the **mailq** command.
/var/spool/mqueue directory	Contains the log file and temporary files associated with the messages in the mail queue.
/var/spool/clientmqueue	Contains the messages that are in mail queue for non-root users.

mailstats Command

Purpose

Displays statistics about mail traffic.

Syntax

mailstats [**-C** *cfFile*] [**-c**] [**-P**] [**-f** *StatFile*] [**-o**] [**-p**]

Description

The **mailstats** command displays the current mail statistics. The time at which the statistics started displays reads the information in the format specified by **ctime**. The statistics for each mailer are displayed on a single line, with the following fields:

Item	Description
`M`	Contains the mailer number.
`msgsfr`	Contains the number of messages received by the local machine from the indicated mailer.
`bytes_from`	Contains the number of Kbytes in the messages received by the local machine from the indicated mailer.
`msgsto`	Contains the number of messages sent from the local machine using the indicated mailer.
`bytes_to`	Contains the number of bytes in the messages sent from the local machine using the indicated mailer.
`msgsrej`	Contains the number of messages rejected.
`msgsdis`	Contains the number of messages discarded.
`Mailer`	Contains the name of mailer.

After the statistics are displayed, a line totaling the value of all of the mailers displays, preceeded with a **T**. This information is separated from the statistics by a line containing only **=** (equal characters). Another line preceeded with a **C** lists the number of connections.

Flags

Item	Description
-C *cfFile*	Specifies use of the *cfFile* instead of the default **sendmailcf** file.
-c	Specifies use of the **submit.cf** file instead of the **sendmail.cf** file.
-f *StatFile*	Specifies use of the *StatFile* instead of the statistics file specified in the **sendmail.cf** file.
-o	Specifies that the name of the mailer does not display in the output.
-p	Outputs information in program readable mode and clears the statistics.
-P	Outputs information in program readable mode without clearing the statistics.

Exit Status

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/etc/mail/statistics	Default **sendmail** statistics file.
/etc/mail/sendmail.cf	Default **sendmail** configuration file.

make Command

Purpose

Maintains, updates, and regenerates groups of programs.

Syntax

make [**-D***Variable*] [**-d** *Option*] [**-e**] [**-i**] [**-j**[*Jobs*]] [**-k**] [**-n**] [**-p**] [**-q**] [**-r**] [**-S**] [**-s**] [**-t**] [**-f** *MakeFile ...*] [*Target ...*]

Description

The **make** command assists you in maintaining a set of programs. Input to the **make** command is a list of file dependency specifications.

There are four types of lines in a makefile: file dependency specifications, shell commands, variable assignments, and comments. In general, lines can be continued from one line to the next by ending them with a \ (backslash). The trailing newline character and initial white space on the following line are compressed into a single space.

File Dependency Specifications

Dependency lines consist of one or more targets, an operator, and zero or more prerequisites (sources). This creates a relationship where the targets depend on the prerequisites and are usually created from them. The exact relationship between the target and the prerequisite is determined by the operator that separates them. The operators are as follows:

Item	Description
:	A target is considered out-of-date if its modification time is less than that of any of its prerequisites. Prerequisites for a target accumulate over dependency lines when this operator is used. The target is removed if the **make** command is interrupted, unless the target has the **.PRECIOUS** attribute.
::	If no prerequisites are specified, the target is always recreated. Otherwise, a target is considered out-of-date if any of its prerequisites were modified more recently than the target. Prerequisites for a target do not accumulate over dependency lines when this operator is used. The target is not removed if the **make** command is interrupted.

File dependency specifications have two types of rules, inference and target. Inference rules specify how a target is to be made up-to-date. These rules have one target with no / (slash) and a minimum of one . (period). Target rules specify how to build the target. These rules can have more than one target.

Makefile Execution

The **make** command executes the commands in the makefile line by line. As **make** executes each command, it writes the command to standard output (unless otherwise directed, for example, using the **-s** flag). A makefile must have a Tab in front of the commands on each line.

When a command is executed through the **make** command, it uses **make**'s execution environment. This includes any macros from the command line to the **make** command and any environment variables specified in the **MAKEFLAGS** variable. The **make** command's environment variables overwrite any variables of the same name in the existing environment.

Note: When the **make** command encounters a line beginning with the word **include** followed by another word that is the name of a makefile (for example, **include depend**), the **make** command attempts to open that file and process its contents as if the contents were displayed where the include line occurs. This behavior occurs only if the first noncomment line of the first makefile read by the **make** command is not the **.POSIX** target; otherwise, a syntax error occurs.

Comments: Comments begin with a # character, anywhere but in a shell command line, and continue to the end of the line.

Environment: The **make** command uses the **MAKEFLAGS** environment variable, if it exists.

Target Rules

Target rules have the following format:

```
target[target...] : [prerequisite...] [;command]
<Tab>command
```

Multiple targets and prerequisites are separated by spaces. Any text that follows the ; (semicolon) and all of the subsequent lines that begin with a Tab character are considered commands to be used to update the target. A new target entry is started when a new line does not begin with a Tab or # character.

Note: The list of prerequisites can be empty.

Special Targets

Special targets cannot be included with other targets; that is, they must be the only target specified. These targets control the operation of the **make** command. These targets are:

Item	Description
.DEFAULT	This is used as the rule for any target (that was used only as a prerequisite) that the **make** command cannot figure out any other way to create. Only the shell script is used. The **<** (left angle bracket) variable of a target that inherits **.DEFAULT**'s commands is set to the target's own name.
.IGNORE	Prerequisites of this target are targets themselves; this causes errors from commands associated with them to be ignored. If no prerequisites are specified, this is the equivalent of specifying the **-i** flag.
.**POSIX**	Causes the **make** command to use a different default rules file. The file, **/usr/ccs/lib/posix.mk**, provides the default rules as specified in the POSIX standard.
.PRECIOUS	Prerequisites of this target are targets themselves. **.PRECIOUS** prevents the target from being removed. If no prerequisites are specified, the **.PRECIOUS** attribute is applied to every target in the file. Usually, when **make** is interrupted (for example, with **SIGHUP**, **SIGTERM**, **SIGINT**, or **SIGQUIT**), it removes any partially made targets. If **make** was invoked with the **-n**, **-p**, or **-q** flags, the target is considered to have the **.PRECIOUS** attribute.
.SCCS_GET	This special target must be specified without prerequisites. If this special target is included in a makefile, the commands associated with this special target are used to get all SCCS files that are not found in the current directory. The default commands that are used to retrieve the source files from SCCS are replaced by the commands associated with this special target. When source files are named in a dependency list, make treats them just like any other target. When a target has no dependencies, but is present in the directory, make assumes that the file is up-to-date. If, however, a SCCS file named SCCS/s.source_file is found for a target source_file, make additionally checks to assure that the target is up-to-date. If the target is missing, or if the SCCS file is newer, make automatically issues the commands specified for the .SCCS_GET special target to retrieve the most recent version. However, if the target is writable by anyone, make does not retrieve a new version.
.SILENT	Prerequisites of the target are targets themselves. This causes commands associated with the target to not be written to standard output before they are executed. If no prerequisites are specified, the **.SILENT** attribute is applied to every command in the file.
.SUFFIXES	Use this name to add more suffixes to the list of file suffixes that **make** recognizes. Prerequisites of the target are appended to the list of known suffixes. If no suffixes are specified, any previously specified suffixes are deleted. These suffixes are used by the inference rules. To change the order of suffixes, you need to specify an empty **.SUFFIXES** entry and then a new list of **.SUFFIXES** entries. A makefile must not associate commands with **.SUFFIXES**.

Inference Rules

The **make** command has a default set of inference rules, which you can supplement or overwrite with additional inference rules definitions in the makefile. The default rules are stored in the external file, **/usr/ccs/lib/aix.mk**. You can substitute your own rules file by setting the **MAKERULES** variable to your own file name from the command line. The following line shows how to change the rules file from the command line:

```
make MAKERULES=/pathname/filename
```

Inference rules consist of target suffixes and commands. From the suffixes, the **make** command determines the prerequisites, and from both the suffixes and their prerequisites, the **make** command determines how to make a target up-to-date. Inference rules have the following format:

```
rule:
<Tab>command
...
```

where `rule` has one of the following forms:

Item	Description
.s1	A single-suffix inference rule that describes how to build a target that is appended with one of the single suffixes.
.s1.s2	A double-suffix inference rule that describes how to build a target that is appended with *.s2* with a prerequisite that is appended with *.s1*.

The *.s1* and *.s2* suffixes are defined as prerequisites of the special target, **.SUFFIXES**. The suffixes *.s1* and *.s2* must be known suffixes at the time the inference rule is displayed in the makefile. The inference rules use the suffixes in the order in which they are specified in **.SUFFIXES**. A new inference rule is started when a new line does not begin with a <Tab> or # character.

If `rule` is empty, for example:

```
rule: ;
```

execution has no effect, and the **make** command recognizes that the suffix exists, but takes no actions when targets are out-of-date.

A ~ (tilde) in the preceding rules refers to an SCCS file. Therefore, the rule, **.c~.o**, would transform an SCCS C language prerequisite file into an object file (**.o**). Because the **s.** of the SCCS file is a prefix, it is incompatible with the **make** command's suffix view. The ~ (tilde) is a way of changing any file reference into an SCCS file reference.

Libraries

A target or prerequisite can also be a member of an archive library and is treated as such if there are parentheses in the name. For example, *library(name)* indicates that *name* is a member of the archive library *library*. To update a member of a library from a particular file, you can use the format *.s1***.a**, where a file with the *.s1* suffix is used to update a member of the archive library. The **.a** refers to an archive library.

Using Macros

In makefiles, macro definitions are defined in the format:

```
variable=value
```

Macros can be displayed throughout the makefile, as follows:

- If a macro is displayed in a target line, it is evaluated when the target line is read.
- If a macro is displayed in a command line, it is evaluated when the command is executed.
- If a macro is displayed in a macro definition line, it is evaluated when the new macro is displayed in a rule or command.

If a macro has no definition, it defaults to **NULL**. A new macro definition overwrites an existing macro of the same name. Macros assignments can come from the following, in the listed order:

1. Default inference rules
2. Contents of the environment
3. Makefiles
4. Command lines.

   Note: The **-e** flag causes environment variables to override those defined in the makefile.

The **SHELL** macro is special. It is set by the **make** command to the path name of the **shell** command interpreter (**/usr/bin/sh**). However, if it is redefined in the makefile or on the command line, this default setting is overridden.

Note: The **SHELL** macro does not affect, and is not affected by, the **SHELL** environment variable.

Shell Commands

Each target can have associated with it a series of shell commands, usually used to create the target. Each of the commands in this script must be preceded by a Tab. While any target can be displayed on a dependency line, only one of these dependencies can be followed by a creation script, unless the **::** operator is used.

If the first, or first two characters, of the command line are one or all of @ (at sign), - (hyphen), and + (plus sign), the command is treated specially, as follows:

Item	Description
@	Causes the command not to be echoed before it is executed.
-	Causes any nonzero exit status of the command line to be ignored.
+	Causes a command line to be executed, even though the options **-n**, **-q**, or **-t** are specified.

A command that has no metacharacters is directly executed by the **make** command. For example, the **make** command consigns the first command in the following example to the shell because it contains the > (greater than sign) shell metacharacter. The second command in the following example does not contain any shell metacharacters, so the **make** command executes it directly:

```
target: dependency
        cat dependency > target
        chmod a+x target
```

Bypassing the shell saves time, but it can cause problems. For example, attempting to execute a C shell script from within a makefile by setting the **SHELL** macro to `/bin/csh` will not work unless the command line also contains at least one shell metacharacter.

```
SHELL=/bin/csh

target: dependency
        my_csh_script
```

This makefile fails because the **make** command attempts to run `my_csh_script` instead of consigning it to the C shell.

Variable Assignments

Variables in the **make** command are much like variables in the shell and consist of all uppercase letters. The **=** operator assigns values to variables. Any previous variable is then overridden. Any white space before the assigned value is removed.

Values can be appended to macro values as follows:

```
macro += word ...
macro += macro1
```

The += operator when used in place of = appends the new value with a single space is inserted between the previous contents of the variable and the appended value.

Variables are expended by surrounding the variable name with either { } (braces) or ( ) (parentheses) and preceding it with a $ (dollar sign). If the variable name contains only a single letter, the surrounding braces or parentheses are not required. This shorter form is not recommended.

Variable substitution occurs at two distinct times, depending on where the variable is being used. Variables in dependency lines are expanded as the line is read. Variables in shell commands are expanded when the **shell** command is executed.

The four classes of variables (in order of increasing precedence) are:

Item	Description
Environment	Variables defined as part of the **make** command's environment.
Global	Variables defined in the makefile or in included makefiles.
Command line	Variables defined as part of the command line.
Local	Variables defined specific to a certain target. The local variables are as follows: **$<** Represents either the full name of a prerequisite that made a target out-of-date (inference rule), or the full name of a target (**.DEFAULT** rule). **$*** Represents the file name section of a prerequisite that made a target out-of-date (in an inference rule) without a suffix. **$@** Represents the full target name of the current target or the archive file name part of the library archive target. **$%** Represents a library member in a target rule if the target is a member of the archive library. You can also use these local variables appended with **D** or **F**: **D** Indicates that the local variable applies to the directory part of the name. This is the path name prefix without a trailing / (slash). For current directories, **D** is a . (period). **F** Indicates that the local variable applies to the file name part of the name. In addition, the **make** command sets or knows about the following variables:
$	A single **$** (dollar sign); that is, **$$** expands to a single dollar sign.
LANG	Determines the locale to use for the locale categories when both **LC_ALL** and the corresponding environment variable (beginning with **LC_**) do not specify a locale.
LC_ALL	Determines the locale to be used to override any values for locale categories specified by the setting of **LANG** or any other **LC_** environment variable.
LC_CTYPE	Determines the locale for the interpretation of sequences of bytes of text data as characters; for example, single- versus multibyte characters in arguments.
LC_MESSAGES	Determines the language in which messages should be written.
MAKEFLAGS	The environment variable, **MAKEFLAGS**, can contain anything that can be specified on **make**'s command line. Anything specified on **make**'s command line is appended to the **MAKEFLAGS** variable, which is then entered into the environment for all programs that **make** executes. Note that the operation of the **-f** and **-p** flags in the **MAKEFLAGS** variable is undefined. Command line flags take precedence over the **-f** and **-p** flags in this variable.

Item	Description
VPATH	Allows you to specify a list of directories to search for prerequisites. The list of directories works like the **PATH** variable in the **SHELL**. The **VPATH** variable can specify multiple directories separated by colons. For example: `VPATH=src:/usr/local/src` This tells the **make** command to search for the following directories in the order given: • The current directory (this happens even without **VPATH**) • `src` (a subdirectory in the current directory) • `/usr/local/src`.

Flags

Item	Description
-D*Variable*	Sets the value of *Variable* to 1.
-d*Option*	Displays detailed information about the files and times that **make** examines (debug mode). The **-d** flag without any options or with the *A* option displays all the debug information available. Individually selectable debug options follow: ***A*** Displays all possible debug information. ***a*** Displays debug information about archive searching and caching. ***d*** Displays debug information about directory searching. ***g1*** Displays debug information about input graph before making anything. ***g2*** Displays debug information about input graph after making everything, or before exiting on an error. ***m*** Displays debug information about making targets, including modification dates. ***s*** Displays debug information about suffix searching. ***v*** Displays debug information about variable assignments.
-e	Specifies that environmental variables override macro assignments within makefiles.
-f *MakeFile*	Specifies a makefile to read instead of the default makefile. If *MakeFile* is - (hyphen), standard input is read. Multiple makefiles can be specified and are read in the order specified.
-i	Ignores nonzero exit of **shell** commands in the makefile. Equivalent to specifying - (hyphen) before each command line in the makefile.
-j[*Jobs*]	Specifies the number of parallel jobs that `make` should use to build the independent targets. The *Jobs* parameter can take any positive integral values. If *Jobs* is not specified, the `make` command does not limit the number of parallel jobs for building the main target.
-k	Continues processing after errors are encountered, but only on those targets that do not depend on the target whose creation caused the error.

Item	Description
-n	Displays commands, but does not run them. However, lines beginning with a + (plus sign) are executed.
-p	Displays the complete set of macro definitions and target descriptions before performing any commands.
-q	Returns a zero status code if the target file is up-to-date; returns a one status code if the target file is not up-to-date. Targets will not be updated when this option is specified. However, a command line with the + (plus sign) prefix will be executed.
-r	Does not use the default rules.
-S	Terminates the **make** command if an error occurs. This is the default and the opposite of **-k** flag.
-s	Does not display commands on the screen as they are performed.
-t	Creates a target or updates its modification time to make it seem up-to-date. Executes command lines beginning with a + (plus sign).
Target	Specifies a target name of the form *Target* or sets the value of variables.

Exit Status

When the **-q** flag is specified, this command returns the following exit values:

Item	Description
0	Successful completion.
1	The target was not up-to-date.
>1	An error occurred.

Otherwise, this command returns the following exit values:

Item	Description
0	Successful completion.
>1	An error occurred.

Examples

1. To make the first target found in the makefile, type:

   ```
   make
   ```

2. To display, but not run, the commands that the **make** command would use to make a file:

   ```
   make  -n search.o
   ```

   Doing this will verify that a new description file is correct before using it.

3. To create a makefile that says that **pgm** depends on two files, **a.o** and **b.o**, and that they, in turn, depend on their corresponding prerequisite files (**a.c** and **b.c**) and a common file, **incl.h**, type:

   ```
   pgm: a.o b.o
            c89 a.o b.o -o pgm
   a.o: incl.h a.c
            c89 -c a.c
   b.o: incl.h b.c
            c89 -c b.c
   ```

4. To make optimized **.o** files from **.c** files, type:

```
.c.o:
        c89 -c -o $*.c
or:
.c.o:
        c89 -c -o $<
```

5. To view the contents of the built-in rules, type:

```
make  -p  -f /dev/null 2>/dev/null
```

6. To use the make command in parallel mode with a maximum of 10 parallel jobs to be used for building the target specified in the makefile, type:

```
make -j10
```

Files

Item	Description
makefile	Contains a list of dependencies.
Makefile	Contains a list of dependencies.
s.makefile	Contains a list of dependencies. It is an SCCS file.
s.Makefile	Contains a list of dependencies. It is an SCCS file.
/usr/ccs/lib/posix.mk	Contains default POSIX rules for the **make** command.
/usr/ccs/lib/aix.mk	Contains default rules for the **make** command.

makedbm Command

Purpose

Makes a Network Information Services (NIS) database map.

Syntax

To Create an Map

/usr/sbin/makedbm [**-b**] [**-i** NIS*InputFile*] [**-o** NIS*OutputFile*] [**-d** NIS*DomainName*] [**-m** NIS*MasterName*] *InputFile OutputFile*

To Create a Non-dbm Formatted Map

/usr/sbin/makedbm [**-u** *dbmFileName*]

Description

The **makedbm** command makes an NIS map. It does this by converting the file named in the *InputFile* parameter into two output files: *OutputFile.pag* and *OutputFile.dir*. Each line in each input file is converted into a single Data Base Manager (DBM) record.

The **makedbm** command is most often invoked from the **/var/yp/Makefile** file to generate NIS maps. All characters leading up to the first space or tab in each line of the **/var/yp/Makefile** file form the key. The rest of the line contains value data. If a line ends with a \ (backslash), data for that record is continued on the next line. NIS clients must interpret the # (pound sign) symbol since the **makedbm** command does not treat it as a comment character. If the *InputFile* parameter is a - (minus sign), the **makedbm** command reads standard input instead.

This command generates a special entry in the output map by using the **YP_LAST_MODIFIED** key, which is the date that the file specified by the *InputFile* parameter was created (or the current time, if the *InputFile* parameter is a - (minus sign)).

Flags

Item	Description
-b	Propagates a map to all servers using the **named** name server.
-i	Creates a special entry with the **YP_INPUT_FILE** key.
-o	Creates a special entry with the **YP_OUTPUT_FILE** key.
-d	Creates a special entry with the **YP_DOMAIN_NAME** key.
-m	Creates a special entry with the **YP_MASTER_NAME** key.
-u	Undoes a DBM file. That is, prints out a DBM file one entry per line, with a single space separating keys from values.

Files

Item	Description
/var/yp/Makefile	Contains rules for making NIS maps.

makedepend Command

Purpose

Create dependencies in makefiles**.**

Syntax

makedepend [**-D***Name***=***Def*] [**-D***Name*] [**-I***IncludeDir*] [**-Y***IncludeDir*] [**-a**] [**-f***MakeFile*] [**-o***ObjSuffix*] [**-p***ObjPrefix*] [**-s***String*] [**-w***Width*] [**-v**] [**-m**] [**—***Options***—**] *SourceFile ...*

Description

The **makedepend** command reads each *SourceFile* in sequence and parses it like a C-preprocessor. It processes all #include, #define, #undef, #ifdef, #ifndef, #endif, #if, and #else directives to determine which #include directives need to be used in a compilation. Any #include directives can reference files having other #include directives, and parsing occurs in these files as well.

Every file that a *SourceFile* includes, directly or indirectly, is what **makedepend** calls a "dependency." These dependencies are then written to a makefile in such a way that the **make** command can determine which object files must be recompiled when a dependency has changed.

By default, **makedepend** places its output in the file named **makefile** if it exists, otherwise **Makefile**. An alternate makefile may be specified with the **-f** flag. **makedepend** first searches the available makefile for the line:

```
# DO NOT DELETE THIS LINE - make depend depends on it.
```

or one provided with the **-s** flag, as a delimiter for the dependency output. If it finds the line, it deletes everything following the line to the end of the makefile and puts the output after the line. If **makedepend** does not find the line, it appends the delimited string to the end of the makefile and places the output immediately after the string.

For each *SourceFile* appearing on the command line, **makedepend** puts lines in the makefile in the following form.

```
SourceFile.o: dfile ...
```

Where `SourceFile.o` is the name from the command line with its suffix replaced with `.o`, and `dfile` is a dependency discovered in an #include directive while parsing the *SourceFile* or one of the files it included.

The algorithm used in this command assumes that all files compiled by a single makefile will be compiled with roughly the same **-I** and **-D** flags, and that most files in a single directory will include largely the same files.

Given these assumptions, **makedepend** expects to be called once for each makefile, with all source files that are maintained by the make file appearing on the command line. It parses each source and include file only once, maintaining an internal symbol table for each. As a result, the first file on the command line takes an amount of time proportional to the amount of time that a normal C preprocessor takes. On subsequent files, if it encounters an include file that it has already parsed, it does not parse again.

For example, imagine you are compiling two files, **file1.c** and **file2.c**, each includes the header file **header.h**. The **header.h** file includes the files **def1.h** and **def2.h**. When you run the command:

```
makedepend file1.c file2.c
```

then **makedepend** will first parse **file1.c** and consequently, **header.h** and then **def1.h** and **def2.h**. It then decides that the dependencies for this first file are:

```
file1.o: header.h def1.h def2.h
```

But when the program parses the second file, **file2.c** and discovers that it, too, includes **header.h**, it does not parse the file, but simply adds **header.h**, **def1.h** and **def2.h** to the list of dependencies for **file2.o**.

> **Note:** If you do not have the source for cpp (the Berkeley C preprocessor), then **makedepend** will compile in such a way that all #if directives will evaluate to False, regardless of their actual value. This may cause the wrong #include directives to be evaluated. In these cases, it is recommended that you write a new parser for #if expressions. The need for a new parser should be clear from the following example:
>
> Imagine you are parsing two files **file1.c** and **file2.c**, each includes the file **def.h**. The list of files that **def.h** includes might be very different when **def.h** is included by **file1.c** than when it is included by **file2.c.** But once **makedepend** arrives at a list of dependencies for a file, it is cast in concrete.

Flags

> **Note:** The **makedepend** command ignores flags it does not understand. Flag usage is similar to that of the **cc** command.

Item	Description
-D*Name*=*Def* or **-D***Name*	Places a definition for the *Name* variable in the **makedepend** command's symbol table. Without the *=Def* specifier, the symbol is defined as 1.
-I*IncludeDir*	Prepends the *IncludeDir* variable to the list of directories searched by the **makedepend** command when it encounters an #include directive. By default, the **makedepend** command searches only the **/usr/include** directory**.**
-Y*IncludeDir*	Replaces all of the standard include directories with a single specified include directory, you can omit *IncludeDir* to prevent searching the standard include directories.

Item	Description
-a	Appends the dependencies to the end of the file instead of replacing them.
-f*MakeFile*	Enables you to specify an alternate makefile in which to place command output.
-o*ObjSuffix*	Specifies an object suffix. For example, some systems may have object files whose suffix is something other than **.o**. This flag allows you to specify another suffix, such as ".b" with **-o.b** or ":obj" with **-o.obj** and so forth.
-p*ObjPrefix*	Prepends the object file prefix to the name of the object file. This flag is used to designate a different directory for the object file. The default is the empty string.
-s*String*	Specifies the starting string delimiter. This flag permits you to specify a different string for **makedepend** to search for in the makefile.
-w*Width*	Changes the maximum line width of output lines. The default maximum is 78 characters.
-v	Causes **makedepend** to display a list of files included by each input file on standard input.
-m	Causes **makedepend** to display a warning if any input file includes another file more than once. In previous version of **makedepend** this was the default behavior. This flag is provided for backward compatibility and to aid in debugging problems related to multiple inclusion.
—*Options***—**	Ignores any unrecognized argument contained within a beginning and ending double hyphen. When **makedepend** encounters a double hyphen (—) in the argument list, any unrecognized argument following it is silently ignored; a second double hyphen terminates this treatment. The double hyphens enable **makedepend** to safely ignore esoteric compiler arguments that might normally be found in a CFLAGS **make** command macro (see the Examples section). All flags that **makedepend** recognizes and that appear between the pair of double hyphens are processed normally.

Examples

Normally, **makedepend** will be used in a makefile target so that typing `makedepend` updates the dependencies for the makefile.

```
SRCS=file1.c file2.c ...
CFLAGS=-O -DHACK -I../foobar -xyz
depend:
      makedepend -- $(CFLAGS) -- $(SRCS)
```

makedev Command

Purpose

Creates binary description files suitable for reading by the **troff** command and its postprocessors.

Syntax

makedev DESC | *FontFile ...*

Description

The **makedev** command creates binary files suitable for reading by the **troff** command and its postprocessors. When the **DESC** file is specified, the **makedev** command creates a **DESC.out** file and a set of font description files using the information contained in the **DESC** file. When a font file is specified, the **makedev** command creates the corresponding font description file.

Options

Item	Description
DESC	Causes a **DESC.out** file to be created.

Parameters

Item	Description
FontFile	Causes a *FontFile***.out** file to be created.

Examples

The following command:

```
makedev B
```

creates a **B.out** file, which contains the font tables for the Times-Bold fonts.

makekey Command

Purpose

Generates an encryption key.

Syntax

makekey

Description

The **makekey** command generates an encryption key for use with programs that perform encryption. Its input and output are usually pipes.

The **makekey** command reads 10 characters from standard input and writes 13 characters to standard output. The first 8 of the 10 input characters can be any sequence of ASCII characters, as specified by the *ASCIICharacters* parameter. The last two input characters, called the salt, are chosen from the sets 0 through 9, a through z, A through Z, . (period), and / (slash). The salt characters are repeated as the first two characters of the output. The remaining 11 output characters are chosen from the same set as the salt and constitute the output key that you use as the encryption key parameter of programs that perform encryption.

Examples

1. To generate an encryption key for input string 1234567890, type the following:

```
$ makekey
1234567890
```

Then press the Ctrl-D key. The following encryption key is generated, and the $ (shell prompt) follows immediately after the generated key on the same line:

```
90y744T/NXw1U$
```

2. To allow **makekey** to accept input strings through pipe, type the following command:

```
$ echo 1234567890 | makekey
```

3. To allow **makekey** to accept input strings from a file, type the following command:

```
$ cat infile
1234567890

$ makekey < infile
```

Alternatively, you can type the following command:

```
$ cat infile | makekey
```

makemap Command

Purpose

Creates database maps for **sendmail**.

Syntax

makemap [**-C** **-N** **-c** *CacheSize* **-d** **-D-e** **-f** **-l** **-o** **-r** **-s** **-t-u** **-v**] Maptype Mapname

Description

The **makemap** command creates the database maps used by the keyed map lookups in the **sendmail** command. It reads input from the standard input and outputs them to the indicated *Mapname*.

Parameters

Item	Description
Maptype	Depending upon how it is compiled, this command handles up to three different database formats: **dbm** DBM format maps. This requires the **ndbm** library. **btree** B-tree format maps. This requires the new Berkeley DB library. **hash** Hash format maps. This requires the new Berkeley DB library. **Note:** In all cases, this command reads lines from the standard input, consisting of two words separated by white space. The first is the database key, the second is the value. The value may contain ''%n'' strings to indicate parameter substitution. Literal percents should be doubled ("%%"). Blank lines and lines beginning with a "#" are ignored.
Mapname	Name of the map.

Note: Do not use **makemap** command to create the aliases data base, but **newaliases** which puts a special token into the data base that is required by **sendmail** command.

If the TrustedUser option is set in the **sendmail** configuration file and **makemap** is invoked as the root user, the generated files are owned by the specified trusted user.

Flags

Item	Description
-c *CacheSize*	Specifies to use the hash and B-tree cache size.
-C	Indicates to use the specified **sendmail** configuration file for looking up the **TrustedUser** option.
-d	Allows duplicate keys in the map. This is only allowed on B-tree format maps. If two identical keys are read, they are both inserted into the map.
-D	Specifies the character to use to indicate a comment (which is ignored) instead of the default of #.
-e	Allows empty values on the right side.
-f	Disables the function of folding all uppercase letters in the key to lowercase. This flag is intended to mesh with the **-f** flag in the **K** line in the **sendmail.cf** file. The value is never case folded.
-l	Lists supported map types.
-N	Includes the null byte that ends strings in the map. This flag must match the **-N** flag in the sendmail.cf **K** line.
-o	Specifies to append to an existing file. This flag allows you to augment an existing file.
-r	Allows replacement of existing keys. Normally, the **makemap** command complains if you repeat a key and does not perform the insert operation.
-s	Ignores safety checks on maps being created. This includes checking for hard or symbolic links in world writeable directories.
-t	Specifies the delimiter to use instead of white space. This flag is also used for dumping files.
-u	Dumps or unmaps the contents of the database to standard output.
-v	Specifies that the command verbosely print its status.

man Command

Purpose

Displays manual entries online.

Syntax

man [[[**-c**] [**-t**] [*section*]] | [**-k** | **-f**]] [**-F**] [**-m**] [**-M***path*] [**-r**] [-a] *title ...*

Description

The **man** command provides reference information on topics, such as commands, subroutines, and files. The **man** command provides one-line descriptions of commands specified by name. The **man** command also provides information on all commands whose descriptions contain a set of user-specified keywords.

The **man** command formats a specified set of manual pages. If you specify a section for the *section* parameter, the **man** command searches in that section of the manual pages for the title specified by the *title* parameter. The value of the *section* parameter can be either an Arabic number from 1 through 8 or a letter.

The section letters are:

Item	Description
C	Specifies commands (including system management commands).
F	Specifies file-type manual pages.
L	Specifies library functions.
n	Specifies new.
l	Specifies local.
o	Specifies old.
p	Specifies public.

Note: The **n**, **l**, **o**, and **p** section specifiers are not valid for reading the hypertext information bases, which contain the operating system documentation.

The section numbers are:

Item	Description
1	Indicates user commands and daemons.
2	Indicates system calls and kernel services.
3	Indicates subroutines.
4	Indicates special files, device drivers, and hardware.
5	Indicates configuration files.
6	Indicates games.
7	Indicates miscellaneous commands.
8	Indicates administrative commands and daemons.

Note: The operating system documentation in the hypertext information databases is grouped into three sections only: command manual pages (in section 1, equivalent to section C), subroutine manual pages (in section 3, equivalent to section L), and file manual pages (in section 4, equivalent to section F). When searching for hypertext information, specifying section 1, 6, 7, or 8 will default to the command manual pages, section 2 or 3 will default to the subroutine manual pages, and section 4 or 5 will default to the file manual pages.

If the *section* parameter is omitted, the **man** command searches all sections of the manual.

The search path the **man** command uses is a list of directories separated by a : (colon) in which manual subdirectories can be found.

The **man** command displays the manual pages as follows:

1. The **man** command searches the **nroff** directories (**man?**) under the **/usr/share/man** directory.
2. The **man** command searches the formatted version directories (**cat?**) under the **/usr/share/man** directory. If the formatted version is available, and if it has a more recent modify time than the **nroff** command source, the **man** command displays the formatted version. Otherwise, the manual page is formatted with the **nroff** command and displayed. If the user has permission, the formatted manual page is deposited in the proper place, so that later invocations of the **man** command do not format the page again.

   Note: There is no **nroff** source for the supplied manual pages. However, you can put **nroff** source for manual pages into the **man** directories and the **man** command can locate and process the **nroff** source.

3. If the **man** command does not find a manual page in the **/usr/share/man/man** or **/usr/share/man/cat** directory, the **man** command searches the paths specified through **-M** option or MANPATH environment variable for nroff directories (man?) and formatted version directories (cat?).
4. If the **man** command does not find a manual page in the **/usr/share/man/man** or **/usr/share/man/cat** or the user-specified **man/cat** directory, the **man** command reads from the hypertext information bases. The hypertext information bases reside in the **/usr/share/man/info** directory structure and contain the operating system documentation. When reading from the hypertext databases, the **man** command does not put any manual pages in the **/usr/share/man/cat** directory structure. The **man** command converts the HTML file into a formatted text file to fit on the display, and displays the manual page using the command described by the PAGER environment variable.
5. If the **man** command does not find a manual page in the hypertext information bases residing in the **/usr/share/man/info** directory structure, it looks for user-specified hypertext information base (through **-M** or MANPATH). The user-defined hypertext information base, should follow the following directory structure:

   ```
   BasePath[/%{ L | l }]/DocLibraryname/Section/command_or_routine_or_filename.htm
   ```

   Where:

   - %L represents the ISO language notation specified using the LC_MESSAGES, `%l` represents the first 2 characters of the ISO language notation specified using the LC_MESSAGES. For example, for LC_MESSAGES=en_US the documents can be placed in `Path/en_US` or `Path/en`.
   - `DocLibraryname` represents the name of the documentation library.
   - `Section` represents the section name, which must be one of the following:
     - `cmds` — Represents Commands Section
     - `libs` — Represents Library Section
     - `files` — Represents Files Section

   Note: If **-m** option is specified, then the search for manual pages will be done only in the order of paths specified through **-M** or the MANPATH environment variable.

When accessing the HTML databases, **man** looks for the operating system library before it proceeds to other LPP libraries. Within these libraries, it processes information in the following order:

Item	Description
cmds	Commands Reference
libs	Subroutines, System Calls
files	Files Reference

If the standard output is a tty, the **man** command pipes its output using the **more** command with the **-s** and **-v** flags. The **-s** flag eliminates multiple blank lines and stops after each page on the screen. The **-v** flag suppresses the display of nonprinting characters to the screen. To continue scrolling, press the space bar. To scroll an additional 11 lines when the output stops, press the Ctrl-D key sequence.

The **PAGER** environment variable can be set to whatever pager is desired. The default value is the **more** command. To change the default pager, enter:

```
PAGER=Somepager
export PAGER
```

For example, if there are customized manual pages which are formatted with reverse or fractional line feeds, the **PAGER** environment variable may be set to **/usr/bin/pg** so that the line feeds are not printed as control characters. This procedure is not necessary for the manual pages.

When the **man** command uses a hypertext database, it can retrieve several articles. For example, `man open` displays several articles. The use of **SIGINT** (Ctrl-C) exits the **man** command completely. On the other hand, `man open close` also displays several articles but the use of **SIGINT** (Ctrl-C) causes **man**

to display the **close** command information instead of exiting. Using **SIGINT** (Ctrl-C) again exits the **man** command completely.

When specifying one of the Network Computing System library routines that contains a **$** (dollar sign) in its name, enter a \ (backslash) preceding the **$**.

Flags

Item	Description
-a	Display all matching entries.
-c	Displays the manual information using the **cat** command.
-f	Displays entries in the keyword database related only to the command name given as the final parameter. You can enter more than one command name, each separated by a space. Use this flag to search for command articles only. To use the **-f** flag, a root user must have previously entered `catman -w` to create the **/usr/share/man/whatis** file.
-F	Display only the first matching entry.
-k	Displays each line in the keyword database that contains a string of characters matching the title given as the final parameter. You can enter more than one title, each separated by a space. To use the **-k** flag, a root user must have previously entered `catman -w` to create the **/usr/share/man/whatis** file.
-m	Only search in the paths specified in MANPATH or **-M**.
-M*path*	Changes the standard location where the **man** command searches for manual information. The path is a colon-separated list of paths, where the following special symbols can be used: • %D – • The default AIX paths for man pages. • %L – A locale-specific directory location corresponding to the LC_MESSAGES category of the current locale. • %l - A locale-specific directory location corresponding to the first 2 characters of the LC_MESSAGES category of the current.
-r	Searches remotely for the manual information. If for any reason the remote search fails, then **man** performs a local search for the requested man page. Any of the following conditions can cause the remote search to fail: • The remote machine is not reachable. • There is a problem reading the URL. • A Java™ applet is not installed or it is not found in the user's search path, specified in the PATH environment variable. **Note:** The **DOCUMENT_SERVER_MACHINE_NAME** environment variable should be set to the name of the documentation search server machine the user wants to use. If the AIX Base Documentation is not supported for the host's locale, the **man** command searches for the documentation for an alternate locale. If the search is successful, the documentation page is displayed after conversion to the local host's locale. If the alternate locale is not installed on the local host, the **man** command fails to display the documentation page.
-t	Formats the manual information using the **troff** command. This flag is ignored if the manual page is found in a hypertext information base.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To display information about the **grep** command, enter the following command:

   ```
   man grep
   ```

2. To display information about the **rpc_$register** library routine, enter the following command:

   ```
   man rpc_\$register
   ```

3. To display all entries in the **/usr/share/man/whatis** keyword database that contain the "mkdir" string , enter the following command:

   ```
   man -k mkdir
   ```

   The output is equivalent to the **apropos** command. You receive output from the **-k** flag only when the **/usr/share/man/whatis** keyword database already exists.

4. To display all entries from the keyword database related to the **nroff** and **troff** commands, enter the following command:

   ```
   man -f nroff troff
   ```

   The output is equivalent to the **whatis** command. You receive output from the **-f** flag only when the **/usr/share/man/whatis** keyword database already exists.

5. To display all **ftp** command related articles in the **/usr/share/man** or **/usr/share/man/local** path, enter the following command:

   ```
   man -M/usr/share/man:/usr/share/man/local ftp
   ```

6. To display all matching entries, enter the following command:

   ```
   man –a title
   ```

7. To display only the first matching entry, enter the following command:

   ```
   man –F title
   ```

8. To search only in the paths specified in MANPATH or **–M**, enter the following command:

   ```
   man -m –M PATH title
   ```

9. To search in the user-defined PATH, enter the following command:

   ```
   man –M PATH title
   ```

Files

Item	Description
/usr/bin/man	Contains the **man** command.

Item	Description
/usr/share/man	Standard manual directory structure.
/usr/share/man/cat?/*	Directory containing preformatted pages.
/usr/share/man/whatis	Contains the keyword database.
/usr/share/man/man?/*	Directory containing **nroff** format manual pages.

manage_disk_drivers Command

Purpose

Displays information about storage families and the driver that manages each family or changes the driver that manages a storage family.

Syntax

manage_disk_drivers [**-l**]

manage_disk_drivers [**-d** [*device*]] [**-o** [*driver_option*]

manage_disk_drivers [**-h**]

Description

The **manage_disk_drivers** command displays a list of storage families and the driver that manages or supports each family. A storage family is a storage model. All disks within the family must be managed by the same driver.

There are two types of drivers supported: multipath I/O (MPIO) and non-MPIO. Most users select the MPIO version of the driver (`AIX_AAPCM` or `AIX_APPCM`), which is the default behavior. However, there are instances where a third-party mulitpath driver is installed. In this case, you might want to disable the built-in AIX MPIO feature. You can perform this operation by selecting the `AIX_non_MPIO` option as the driver option.

Note: Not all the storage families support the `AIX_non-MPIO` driver.

If you specify the **-d** flag with the storage device name, the **manage_disk_drivers** command changes the driver selection to the alternate supported driver that is specified with the **-o** flag.

The **manage_disk_drivers** command saves the Object Data Manager (ODM) database that reflects the driver change, and displays a message to restart the system for the change to take effect.

Flags

Item	Description
-l	Lists all the storage devices and their supported drivers.
-d	Specifies the storage device family when you are selecting a driver. **Note:** The **-o** flag is required to use this flag.
-o	Specifies the driver to be used for the selected storage family. **Note:** The **-d** flag is required to use this flag.

NO_OVERRIDE driver option

If the present driver attribute is set to NO_OVERRIDE, the AIX operating system selects an alternate path control module (PCM), such as Subsystem Device Driver Path Control Module (SDDPCM), if it is installed.

Examples

1. To list all the storage families and their supported drivers, enter the following command:

```
manage_disk_drivers -l
```

   The system displays a message similar to the following example:

```
Device           Present Driver      Driver Options
2810XIV          AIX_AAPCM           AIX_AAPCM,AIX_non_MPIO
DS4100           AIX_APPCM           AIX_APPCM
DS4200           AIX_APPCM           AIX_APPCM
DS4300           AIX_APPCM           AIX_APPCM
DS4500           AIX_APPCM           AIX_APPCM
DS4700           AIX_APPCM           AIX_APPCM
DS4800           AIX_APPCM           AIX_APPCM
DS3950           AIX_APPCM           AIX_APPCM
DS5020           AIX_APPCM           AIX_APPCM
DCS3700          AIX_APPCM           AIX_APPCM
DS5100/DS5300    AIX_APPCM           AIX_APPCM
DS3500           AIX_APPCM           AIX_APPCM
XIVCTRL          MPIO_XIVCTRL        MPIO_XIVCTRL,nonMPIO_XIVCTRL
2107DS8K         NO_OVERRIDE         NO_OVERRIDE,AIX_AAPCM,AIX_non_MPIO
IBMFlash         NO_OVERRIDE         NO_OVERRIDE,AIX_AAPCM,AIX_non_MPIO
IBMSVC           NO_OVERRIDE         NO_OVERRIDE,AIX_AAPCM,AIX_non_MPIO
```

2. To modify the driver to use the AIX_non_MPIO option that manages 2810XIV device, enter the following command:

```
manage_disk_drivers -d 2810XIV -o AIX_non_MPIO
```

   The system displays a message similar to the following example:

```
# manage_disk_drivers -d 2810XIV -o AIX_non_MPIO
********************** ATTENTION *************************
For the change to take effect the system must be rebooted
```

managefonts Command

Purpose

Provides the user with a simple menu-based interface to update or change the set of installed font families on the system.

> **Note:** You must have root user authority to run the **managefonts** script. The **managefonts** script is contained in the **/usr/lib/ps/ditroff.fonts/managefonts** file.

Syntax

managefonts [*Option*]

Description

The **managefonts** command provides the user with a simple menu-based interface to update or change the set of installed font families on the system. If no command line arguments are provided, the menu-based interface is used. Command-line arguments can be used to provide the equivalent of the menu selections.

A set of font families is installed on the system at the time the TranScript Tools option of the Text Formatter Services Package is installed on the system. This default setup includes the standard 13 fonts comprising the Times, Courier, and Helvetica font families. You can use the program called up by the **managefonts** command to erase the current configuration and replace it with a new one. There are several predefined packages of font families that can be installed this way:

Item	Description
Times Family Only	This is the most minimal configuration that allows the TranScript Tools option to run.
Standard13 Package	This package builds the Times, Courier, and Helvetica font families. This was the package installed on your system with TranScript.
Standard35 Package	This font family package includes the Standard13 package font families in addition to the following: Avant Garde, Bookman, New Century Schoolbook, and Palatino font families.
All Font Families	This package installs all the font families available for installation.

You can also use the **managefonts** command to add new font families one at a time. A menu of available fonts is displayed and users can select which font family they want to be built. The program prevents building of font families that are already installed.

The **managefonts** command includes help screens to assist the user in installing font families.

Notes:

1. Font families cannot be deleted directly. To delete font families, it is first necessary to install a package containing the minimal subset of families desired. After the package is installed, it is possible to add font families, one at a time, from the Individual Fonts Menu. For instance, if your current configuration is Times, Courier, and Helvetica, and you want only Times and Courier, you can use the **managefonts** program to install the Times Only Package.
2. There is no command-line syntax equivalent to the menu items in the **managefonts** program.

The command line arguments are acted upon in the order they are given, reading left to right. The following are the valid values for the *option* parameter and their meanings:

Item	Description
init0	Initialize for the installation of a font package.
clean	Remove all temporary files and previously installed fonts.
cleanall	Remove all the temporary files, the previously installed fonts, and the TranScript troff font files installed.
default	Install the Standard 13 fonts.
standard13	Install the Standard 13 fonts.
standard35	Install the Standard 35 fonts.
all	Install all possible fonts.
CourierFamily	Install the Courier Family.
HelveticaFamily	Install the Helvetica Family.
HelvNarrowFamily	Install the Helvetica Narrow Family.
AvantGardeFamily	Install the Avant Garde Family.
BookmanFamily	Install the Bookman Family.
GaramondFamily	Install the Garamond Family.
LubalinFamily	Install the Lubalin Family.
NewCenturyFamily	Install the New Century Family.
OptimaFamily	Install the Optima Family.
PalatinoFamily	Install the Palatino Family.

Item	Description
SouvenirFamily	Install the Souvenir Family.
ZapfFamily	Install the Zapf Family.
BaseFamily	Install the Base Family, such as Times Roman.

Examples

1. To install the standard 13 fonts:

```
managefonts cleanall standard13
```

2. To install the standard 35 fonts:

```
managefonts cleanall standard35
```

3. To install all the fonts:

```
managefonts cleanall all
```

4. To install the Courier Family (the Times Roman or Base Family must have been previously installed):

```
managefonts init0 CourierFamily clean
```

mant Command

Purpose

Typesets manual pages.

Syntax

mant [**-M** *Media*] [**-a**] [**-c**] [**-e**] [**-t**] [**-z**] [**-T** *Name*] [*troffFlags*] [*File* ... | **-**]

Description

The **mant** command uses the manual page macros (**man** macro package) to typeset manual pages. The *File* parameter specifies the files to be processed by the **mant** command. Files must be displayed after all flags. If no file name is specified, the **mant** command prints a list of its flags. If a **-** (minus sign) is specified for the *File* parameter, standard input is read.

The **mant** command has flags to specify preprocessing by the **tbl** command, **cw** command, or **eqn** command. Flags from the **troff** command can be specified with the *troffFlag*s parameter.

If the input contains a **troff** command comment line consisting solely of the string `'\"  x` (single quotation mark, backslash, double quotation mark, *x*), where x is any combination of the three letters **c**, **e**, and **t**, and where there is exactly one character space between the double quotation mark and x, then the input is processed through the appropriate combination of the **cw** command, **eqn** command, and **tbl** command, respectively, regardless of the command-line options.

> **Note:** Use the **-o***List* flag of the **troff** command to specify ranges of pages to be output. Calling the **mant** command with one or more of the **-c** flag, **-e** flag, **-t** flag, and **-** (minus) flags together with the **-o***List* flag of the **troff** command, give a `broken pipe` message if the last page of the document is not specified by the *List* variable. This broken pipe message is not an indication of any problem and can be ignored.

The **mant** command, unlike the **troff** command, automatically pipes its output to a specific postprocessor, according to the following flags, environment variable, or default setting unless specifically requested not to do so:

Item	Description
-z	Indicates that no postprocessors are used.
-T*Name*	Prepares the output for the printing device specified by the *Name* variable.
TYPESETTER	Specifies a particular printing device for the system environment.
default	Sends to **ibm3816**.

Flags, other than the ones in the following list, are passed to the **troff** command or to the macro package, as appropriate. All flags must be displayed before the specified file names.

Flags

All flags must appear before the specified file names.

Item	Description
-a	Calls the **-a** flag of the **troff** command.
-c	Preprocesses the input files with the **cw** command.
-e	Preprocesses the input files with the **eqn** command.
-M *Media*	Specifies a paper size in order to determine the amount of imageable area on the paper. Valid values for the *Media* variable are: **A4** Specifies a paper size of 8.3 X 11.7 inches (210 X 297 mm). **A5** Specifies a paper size of 5.83 X 8.27 inches (148 X 210 mm). **B5** Specifies a paper size of 6.9 X 9.8 inches (176 X 250 mm). **EXEC** Specifies a paper size of 7.25 X 10.5 inches (184.2 X 266.7 mm). **LEGAL** Specifies a paper size of 8.5 X 14 inches (215.9 X 355.6 mm). **LETTER** Specifies a paper size of 8.5 X 11 inches (215.9 X 279.4 mm). This is the default value. **Note:** The *Media* variable is not case-sensitive.
-t	Preprocesses the input files with the **tbl** command.
-z	Prepares the output without the postprocessor.
-T*Name*	Prepares the output for the specified printing device. Possible *Name* variables are: **ibm3812** 3812 Pageprinter II. **ibm3816** 3816 Pageprinter. **hplj** Hewlett-Packard LaserJet II. **ibm5587G** 5587-G01 Kanji Printer multi-byte language support. **psc** PostScript printer. **X100** AIXwindows display.

Item	Description
-	Forces input to be read from standard input.

mark Command

Purpose

Creates, modifies, and displays message sequences.

Syntax

mark [**+***Folder*] [**-list**] [**-sequence** *Name* [*Messages*...] [**-add** | **-delete**] [**-zero** | **-nozero**] [**-public** | **-nopublic**]]

Description

The **mark** command creates, deletes, adds, and lists messages in a sequence. The **mark** command by default lists all of the sequences and their messages for the current folder. If you use the **-add** or **-delete** flag, you must also use the **-sequence** flag. When all messages are deleted from a sequence, the **mark** command removes the sequence name from the folder.

To create a new sequence, enter the **-sequence** flag with the name of the sequence you want to create. The **mark** command creates the sequence starting with the current message. By default, the **mark** command places the sequence in the current folder. If you specify a folder, that folder becomes the current folder.

Flags

Item	Description
-add	Adds messages to a sequence. The **-add** flag is the default. If you do not specify a message, the **mark** command uses the current message. **Note:** You can only use this flag with the **-sequence** flag.
-delete	Deletes messages from a sequence. If you do not specify a message, the current message is deleted by default. **Note:** You can only use this flag with the **-sequence** flag.
+*Folder*	Specifies the folder to examine.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-list	Displays the messages in a sequence. By default, the **-list** flag displays all the sequence names and messages defined for the current folder. To see a specific sequence, use the **-sequence** flag with the **-list** flag.
-nopublic	Restricts a sequence to your usage. The **-nopublic** flag does not restrict the messages in the sequence, only the sequence itself. This option is the default if the folder is write-protected from other users.
-nozero	Modifies the sequence by adding or deleting only the specified messages. This flag is the default.
-public	Makes a sequence available to other users. The **-public** flag does not make protected messages available, only the sequence itself. This flag is the default if the folder is not write-protected from other users.

Item	Description
-sequence *Name*	Specifies a sequence for the **-list**, **-add**, and **-delete** flags. You cannot use **new** as a sequence name.
-zero	Clears a sequence of all messages except the current message. When the **-delete** flag is also specified, the **-zero** flag places all of the messages from the folder into the sequence before deleting any messages.
Messages	Specifies messages in a sequence. You can specify more than one message at a time. Messages are identified with following references: ***Number*** Number of the message **all** All the messages in a folder **cur or . (period)** Current message (the default) **first** First message in a folder **last** Last message in a folder **new** New messages in a folder **next** Message following the current message **prev** Message preceding the current message If the **-list** flag is used, the default for the *Messages* parameter is **all**. Otherwise, the default is the current message.

Profile Entries

The following entry is found in the *UserMHDirectory***/context** file:

Item	Description
`Current-Folder:`	Specifies the default current folder.

The following entry is found in the **$HOME/.mh_profile** file:

Item	Description
`Path:`	Specifies the MH directory.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To see the list of all sequences defined for the current folder, enter:

```
mark
```

The system displays a message similar to the following:

```
cur: 94
test: 1-3 7 9
```

In this example, message 94 is the current message number in the current folder. The message sequence called `test` includes message numbers 1, 2, 3, 7, and 9.

2. To see the list of all the sequences defined for the `meetings` folder, enter:

   ```
   mark  +meetings
   ```

   The system displays a message similar to the following:

   ```
   cur: 5
   dates: 12 15 19
   ```

3. To create a new message sequence called `schedule` in the current folder, enter:

   ```
   mark  -sequence schedule
   ```

   The system displays the shell prompt to indicate that the `schedule` sequence was created. By default, the system adds the current message to the new sequence.

4. To delete message `10` from the `schedule` sequence, enter:

   ```
   mark  -sequence schedule 10  -delete
   ```

Files

Item	Description
$HOME/.mh_profile	Specifies the MH user profile.
/usr/bin/mark	Contains the **mark** command.

mesg Command

Purpose

Permits or refuses write messages.

Syntax

mesg [**n** | **y**]

Description

The **mesg** command controls whether other users on the system can send messages to you with either the **write** command or the **talk** command. Called without arguments, the **mesg** command displays the current workstation message-permission setting.

The shell startup process permits messages by default. You can override this default action by including the line `mesg  n` in your **$HOME/.profile** file. A user with root user authority can send write messages to any workstation, regardless of its message permission setting. Message permission has no effect on messages delivered through the electronic mail system.

If you add `mesg  y` to your **$HOME/.profile**, you will be able to receive messages from other users via the **write** command or the **talk** command.

If you add `mesg  n` to your **$HOME/.profile**, you will not be able to receive messages from other users using the **write** command or the **talk** command.

Flags

Item	Description
n	Allows only the root user the permission to send messages to your workstation. Use this form of the command to avoid having others clutter your display with incoming messages.
y	Allows all workstations on the local network the permission to send messages to your workstation.

Exit Status

This command returns the following exit values:

Item	Description
0	Receiving messages is allowed.
1	Receiving messages is not allowed.
>1	An error occurred.

Examples

1. To allow only the root user the permission to send messages to your workstation, enter:

   ```
   mesg n
   ```

2. To allow everyone the permission to send messages to your workstation, enter:

   ```
   mesg y
   ```

3. To display what your current message-permission setting is, enter:

   ```
   mesg
   ```

   Information similar to the following is displayed:

   ```
   is y
   ```

   In the previous example, the current message-permission setting is y (allowing all users on the local network the permission to send messages to your workstation). If you change the message-permission setting to n (allowing only the root user the permission to send messages to your workstation), information similar to the following is displayed:

   ```
   is n
   ```

Files

Item	Description
/dev/tty*	Supports the controlling terminal interface.
$HOME/.profile	Controls startup processes and daemons.

mhl Command

Purpose

Produces formatted listings of messages.

Syntax

mhl [**-form** *FormFile*] [**-folder** **+***Folder*] [**-moreproc** *Command* | **-nomoreproc** [**-bell** | **-nobell**] [**-clear** | **-noclear**]] [**-length** *Number*] [**-width** *Number*]

Description

The **mhl** command creates formatted lists of messages. The command is usually started through the `showproc:` profile entry or through the **-showproc** flag in other MH commands. When displaying messages, the **mhl** command uses the directions listed in the format file. If you specify more than one message, the **mhl** command provides a prompt before displaying each screen of messages.

If the **-nomoreproc** flag is specified, the **mhl** command prompts the user to press the Return key (the Ctrl-D key sequence is also acceptable) to see the next message. To stop the current message output and receive a prompt for the next message, press the Ctrl-D key sequence. Press the QUIT key sequence to stop the command output.

> **Note:** To use the **mhl** command, you must make the folder you wish to work with the current directory.

Flags

Item	Description
-bell	Produces a bell at the end of each page. When the **-nomoreproc** flag is specified or the `moreproc:` profile entry is defined, but empty, the **-bell** flag is the default.
-clear	Clears the screen after each page when the output device is a display. The **mhl** command uses the **$TERM** environment variable to determine the type of display. When the output device is not a display, the **-clear** flag inserts a form feed character at the end of each message. This flag affects the **mhl** command only if the `moreproc:` profile entry is defined and empty.
-folder **+***Folder*	Identifies the folder to be used for the **mhl.format** file's `MessageName:` entry. The default is the value of the **$mhfolder** environment variable.
-form *FormFile*	Specifies a file containing an alternate output format. The default format is described in the *UserMHDirectory***/mhl.format** file. If this file does not exist, the **mhl** command uses the system default format described in the **/etc/mh/mhl.format** file.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-length *Number*	Sets the screen length for the output. The default is the value indicated by the **$TERM** environment variable. If that value is not appropriate, the default is 40 lines.
-moreproc *Command*	Uses the value of the *Command* variable instead of the value of the `moreproc:` entry specified in the **$HOME/.mh_profile** file.
-nobell	Suppresses the bell at the end of each page. This flag affects the **mhl** command only if the output device is a display, the **-nomoreproc** flag is used, or the `moreproc:` profile entry is defined and empty.

Item	Description
-noclear	Prevents clearing of the screen at the end of each page when the output device is a display. When the output device is not a display, the **-clear** flag does not insert a form-feed character at the end of each message. This flag is the default when the **-moreproc** flag is used or the `moreproc:` entry is defined and is empty.
-nomoreproc	Sets the `moreproc:` entry as an empty value.
-width *Number*	Sets the screen width for the output. The default is the value indicated by the **$TERM** environment variable. If that value is not appropriate, the default is 80 characters.

Profile Entries

The following entry is found in the *UserMHDirectory***/.mh_profile** file:

Item	Description
`moreproc:`	Specifies the interactive program for communicating with the user.

Examples

1. To list message 5 in the **inbox** folder, change the directory to **inbox**:

   ```
   cd /home/mickey/Mail/inbox
   ```

   Then enter:

   ```
   /usr/lib/mh/mhl 5
   ```

   A display similar to the following appears:

   ```
   --- Using template MHL.FORMAT ---
   Date:

   To:
   cc:

   From:
   Subject:

   Message Text
   ```

2. To display more than one message, enter:

   ```
   /usr/lib/mh/mhl 5 6 7
   ```

Files

Item	Description
$HOME/.mh_profile	Contains the MH user profile.
/etc/mh/mhl.format	Defines the default MH message template.
*UserMHDirectory***/mhl.format**	Specifies a user's default message template. (If it exists, it overrides the default MH message template.)
/usr/lib/mh/mhl	Contains the **mhl** command.

mhmail Command

Purpose

Sends or receives mail.

Syntax

mhmail *User* ... [**-cc** *User* ...] [**-from** *User* ...] [**-subject** "*String*"] [**-body** "*String*"]

Description

The **mhmail** command composes, sends, and files messages. To file a message, enter the **mhmail** command without any flags. The default folder is **$HOME/inbox**.

If you specify one or more user addresses with the *User* parameter, the **mhmail** command accepts text from your terminal and composes a message. You can end the message text by pressing the Ctrl-D key sequence. The **mhmail** command sends a copy of the message to each specified address.

Flags

Item	Description
-body "*String*"	Sends a message with the specified string as the body. You must enclose the string in quotes. When you specify the **-body** flag, the **mhmail** command does not accept text from the terminal.
-cc *User...*	Sends a copy of the message to the specified users. The **mhmail** command puts the addresses in the `cc:` field.
-from *User...*	Places the specified user address in the `From:` field of the message.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-subject "*String*"	Places the specified text string in the `Subject:` field of the message.

Examples

1. To receive new mail and file it into the default mail folder, **$USER/Mail/inbox**, enter:

   ```
   mhmail
   ```

   The system displays a message similar to the following:

   ```
   Incorporating new mail into inbox...
   65+  04/08 jim@athena.a   Meeting        <<The meeting will
   66   04/08 jim@athena.a   Schedule       <<Schedule change
   ```

   In this example, two messages are filed in the `inbox` file. The subject of the first message is `Meeting`, and the first line starts with the words `The meeting will`. The subject of the second message is `Schedule`, and the first line starts with the words `Schedule change`.

2. To send a message regarding a schedule change to user `jamie` on system `venus`, enter:

   ```
   mhmail jamie@venus  -subject "Schedule Change"
   ```

   The system waits for you to enter the text of the message. After completing the last line of the text, press the Enter key and then the Ctrl-D key sequence to send the message.

Files

Item	Description
/var/spool/Mail/$USER	Defines the location of the mail drop.
/usr/bin/mhmail	Contains the **mhmail** command.

mhpath Command

Purpose

Prints full path names of messages and folders.

Syntax

mhpath [**+***Folder*] [*Messages* [,*Messages*] ...]

Description

The **mhpath** command lists the path names of folders and messages. By default, the command lists the path name of the current folder.

Flags

Item	Description
+*Folder*	Specifies which folder path to list.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.

Item	Description
Messages	Specifies the messages for which you want to list path names. The *Messages* parameter can specify several messages, a range of messages, or a single message. Use the following references to specify messages. ***Number*** Number of the message. When specifying multiple messages, separate each message number with a comma. When specifying a range of messages, separate the upper and lower ends of the range with a hyphen. **Note:** You cannot use the **new** variable when specifying a range. ***Sequence*** A group of messages specified by the user. Recognized values include: **all** All the messages in a folder. **cur or . (period)** Current message. **first** First message in a folder. **last** Last message in a folder. **new** Path name that the system will assign to the next message that is incorporated. **next** Message following the current message. **prev** Message immediately before the current message.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Current-Folder:`	Sets the default current folder.
`Path:`	Specifies a user's MH directory.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Examples

1. To list the path name of the current folder, enter:

   ```
   mhpath
   ```

   The system responds with a message similar to the following:

   ```
   /home/tom/Mail/inbox
   ```

2. To list the path names for messages 2 through 4 in the **source** folder, enter:

```
mhpath  +source 2-4
```

The system responds with a message similar to the following:

```
/home/tom/Mail/source/2
/home/tom/Mail/source/3
/home/tom/Mail/source/4
```

3. To list the path name the system will assign to the next message added to the current folder, enter:

```
mhpath new
```

The system responds with a message similar to the following:

```
/home/tom/Mail/source/5
```

In this example, the next message will be message 5 in user `tom`'s current folder, `/home/tom/Mail/source`.

Files

Item	Description
$HOME/.mh_profile	Defines the user's MH profile.
/usr/bin/mhpath	Contains the **mhpath** command.

migratelp Command

Purpose

Moves allocated logical partition from one physical partition to another physical partition on a different physical volume.

Syntax

migratelp *LVname*/*LPartnumber*[/*Copynumber*] *DestPV*[/*PPartNumber*]

Description

The **migratelp** moves the specified logical partition *LPartnumber* of the logical volume *LVname* to the *DestPV* physical volume. If the destination physical partition *PPartNumber* is specified it will be used, otherwise a destination partition is selected using the intra region policy of the logical volume. By default the first mirror copy of the logical partition in question is migrated. A value of 1, 2 or 3 can be specified for *Copynumber* to migrate a particular mirror copy.

Note:

1. You must consider the partition usage, reported by **lvmstat**, on the other active concurrent nodes in case of a concurrent volume group.
2. Strictness and upper bound settings are not enforced when using `migratelp`.
3. Running this command on an active, firmware-assisted, dump logical volume temporarily changes the location of the dump device to **/dev/sysdumpnull**. After you have successfully migrated the logical volume, this command calls the following command to set the firmware-assisted, dump logical volume to the original logical volume.

```
sysdumpdev -P
```

The **migratelp** command fails to migrate partitions of striped logical volumes.

Security

To use **migratelp**, you must have root user authority.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To move the first logical partitions of logical volume lv00 to hdisk1, type:

```
migratelp lv00/1 hdisk1
```

2. To move second mirror copy of the third logical partitions of logical volume hd2 to hdisk5, type:

```
migratelp hd2/3/2 hdisk5
```

3. To move third mirror copy of the 25th logical partitions of logical volume testlv to 100th partition of hdisk7, type:

```
migratelp testlv/25/3 hdisk7/100
```

Files

Item	Description
/usr/sbin	Directory where the **migratelp** resides.

migratepv Command

Purpose

Moves allocated physical partitions from one physical volume to one or more other physical volumes.

Syntax

migratepv [**-i**] [**-l** *LogicalVolume*] *SourcePhysicalVolume DestinationPhysicalVolume*...

Description

The **migratepv** command moves allocated physical partitions and the data they contain from the *SourcePhysicalVolume* to one or more other physical volumes. To limit the transfer to specific physical volumes, use the names of one or more physical volumes in the *DestinationPhysicalVolume* parameter; otherwise, all the physical volumes in the volume group are available for the transfer. All physical volumes must be within the same volume group. The specified source physical volume cannot be included in the list of *DestinationPhysicalVolume* parameters.

Note:

1. To use this command, you must either have `root` user authority or be a member of the **system** group.
2. The **migratepv** command is not allowed on a snapshot volume group or a volume group that has a snapshot volume group.
3. Running this command on a physical volume that has an active, firmware-assisted, dump logical volume temporarily changes the location of the dump device to **/dev/sysdumpnull**. After you have

successfully migrated the logical volume, this command calls the following command to set the firmware-assisted, dump logical volume to the original logical volume.

```
sysdumpdev -P
```

The allocation of the new physical partitions follows the policies defined for the logical volumes that contain the physical partitions being moved.

When you migrate a physical volume, the boot logical volume must remain intact. Two contiguous physical partitions and the new boot image must be built on the new boot logical volume.

If you specify a logical volume that contains the boot image, the **migratepv -l** command attempts to find enough contiguous partitions on one of the target physical volumes. If the migration is successful, the **migratepv** command prints a message that recommends the user run the **bosboot** command to indicate a change in the boot device. The attempted migration fails if the **migratepv -l** command is unable to find enough contiguous space to satisfy the request.

Note: All Logical Volume Manager migrate functions work by creating a mirror of the logical volumes involved, then resynchronizing the logical volumes. The original logical volume is then removed. If the **migratepv** command is used to move a logical volume containing the primary dump device, the system will not have an accessible primary dump device during the execution of the command. Therefore, a dump taken during this execution may fail. To avoid this, reassign the primary dump device using the **sysdumpdev** command or ensure there is a secondary dump device defined before using **migratepv**.

You can use the System Management Interface Tool (SMIT) **smit migratepv** fast path to run this command.

Flags

Item	Description
-i	Reads the *DestinationPhysicalVolume* parameter from standard input.
-l *LogicalVolume*	Moves only the physical partitions allocated to the specified logical volume and located on the specified source physical volume.

Examples

1. To move physical partitions from `hdisk1` to `hdisk6` and `hdisk7`, enter:

   ```
   migratepv hdisk1 hdisk6 hdisk7
   ```

   Physical partitions are moved from one physical volume to two others within the same volume group.
2. To move physical partitions in logical volume `lv02` from `hdisk1` to `hdisk6`, enter:

   ```
   migratepv -l lv02 hdisk1 hdisk6
   ```

   Only those physical partitions contained in `lv02` are moved from one physical volume to another.

Files

Item	Description
/usr/sbin	Directory where the **migratepv** command resides.
/tmp	Directory where the temporary files are stored while the command is running.

migwpar Command

Purpose

After an operating system migration of a global system to version 7, the **migwpar** command is used to migrate a workload partition (WPAR) that was created on the version 6 global system, to version 7.

The **migwpar** command can also be used to migrate from a 5.2 or 5.3 versioned WPAR to a native version 7 WPAR.

The **migwpar** command can also be used to enable a 5.2 or 5.3 versioned WPAR to work after the global system is migrated to a new operating system level. The versioned WPAR remains at the previous level.

Syntax

migwpar [**-d**<*software_source*>] [-V]{ **-A** | **-f** <*wparNamesFile*> | [**-C**] **wpar_name** }

Description

After an operating system migration of a global system to version 7, the **migwpar** command is used to migrate a workload partition (WPAR) that was created on the version 6 global system, to version 7. Software that no longer exists (it might be replaced by a different software package) on the global system, is removed.

Ensure that all software on the global system has been migrated before you begin to migrate the WPARs. The use of the pre_migration script before the global system migration, and the post_migration script after the migration, will provide data that can be used to verify the migration, such as listing software that will be removed during the migration, and software that did not migrate.

The **-C** flag is used to migrate from a 5.2 or 5.3 versioned WPAR to a native version 7 WPAR. The **-C** flag is incompatible with the list flags (**-A** or **-f**). When you migrate a versioned WPAR, the **-d** software_source information is mandatory.

The **-V** flag can be used to enable a versioned WPAR after the global system is migrated. This option maintains the WPAR at its current level and allows the WPAR to work on the newly migrated global system. This functionality is available on IBM AIX 7.2 with Technology Level 2 or later. Only `rootvg` versioned WPARs require enablement by using the **-V** option.

Note: It is highly recommended to back up the WPAR before you begin to migrate.

A log of all actions of the **migwpar** command is saved in the `/var/adm/ras/migwpar.log` file. The output of the actual software migration of each WPAR is saved in `/var/adm/ras/devinst.log` within the WPAR.

Flags

Item	Description
-A	Migrates all migratable WPARs
-f *wparNamesFile*	Migrates the list of WPARs contained in the file wparNamesFile, one per line.
-C *wparName*	Migrates the specified 5.2 or 5.3 versioned WPAR.
-d *software_source*	Specifies the installation location that is used for the detached WPAR migration. **Note:** The install (or update) images in the specified location must be the same as the ones used to install (or update) the global system.
-V	Enables a WPAR after the global system is migrated. The **-V** option is not compatible with the **-C** or **-d** flags.

Security

Access control: Only the root user can run this command.

Examples

1. After the base operating system of GLOBAL has been migrated to version 7 to migrate a single WPAR, wpar1 to version 7, enter the following command:

```
# migwpar wpar1
```

2. To migrate a detached WPAR, wpar2 by using install images from **/images**, enter the following command:

```
# migwpar -d /images wpar2
```

3. To migrate all shared WPARs, enter the following command:

```
migwpar -A
```

4. To migrate all detached WPARs by using install images in **/images**, enter the following command:

```
migwpar -A -d /images
```

5. To migrate a 5.2 versioned WPAR wpar_52 by using install images in **/images**, enter the following command:

```
migwpar -d /images -C wpar_52
```

6. To enable all versioned WPARs on a system after a global system is migrated, enter the following command:

```
migwpar -VA
```

mirrorvg Command

Purpose

Mirrors all the logical volumes that exist on a given volume group.

Syntax

mirrorvg [**-S** | **-s**] [**-Q**] [**-c** *copies*] [**-m**] [**-p** *copyn=mirrorpool*] *volumegroup* [*physicalvolume ...*]

Description

The **mirrorvg** command takes all the logical volumes on a given volume group and mirrors those logical volumes. This same functionality may also be accomplished manually if you execute the **mklvcopy** command for each individual logical volume in a volume group. As with **mklvcopy**, the target physical drives to be mirrored with data must already be members of the volume group. To add disks to a volume group, run the **extendvg** command.

By default, **mirrorvg** attempts to mirror the logical volumes onto any of the disks in a volume group. If you wish to control which drives are used for mirroring, you must include the list of disks in the input parameters, *physicalvolume.* Mirror strictness is enforced. Additionally, **mirrorvg** mirrors the logical volumes, using the default settings of the logical volume being mirrored. If you wish to violate mirror strictness or affect the policy by which the mirror is created, you must execute the mirroring of all logical volumes manually with the **mklvcopy** command.

When **mirrorvg** is executed, the default behavior of the command requires that the synchronization of the mirrors must complete before the command returns to the user. If you wish to avoid the delay, use the **-S** or **-s** option. Additionally, the default value of 2 copies is always used. To specify a value other than 2, use the **-c** option.

Restrictions:

- To use this command, you must either have root user authority or be a member of the **system** group.
- You cannot use the **mirrorvg** command on a snapshot volume group.
- You cannot use the **mirrorvg** command on a volume group that has an active firmware assisted dump logical volume.

Attention: The **mirrorvg** command may take a significant amount of time before completing because of complex error checking, the amount of logical volumes to mirror in a volume group, and the time is takes to synchronize the new mirrored logical volumes.

You can use the System Management Interface Tool (SMIT) **smit mirrorvg** fast path to run this command.

Flags

Item	Description
-c *copies*	Specifies the minimum number of copies that each logical volume must have after the **mirrorvg** command has finished executing. It may be possible, through the independent use of **mklvcopy**, that some logical volumes may have more than the minimum number specified after the **mirrorvg** command has executed. Minimum value is 2 and 3 is the maximum value. A value of 1 is ignored.
-m *exact map*	Allows mirroring of logical volumes in the exact physical partition order that the original copy is ordered. This option requires you to specify a `PhysicalVolume(s)` where the exact map copy should be placed. If the space is insufficient for an exact mapping, then the command will fail. You should add new drives or pick a different set of drives that will satisfy an exact logical volume mapping of the entire volume group. The designated disks must be equal to or exceed the size of the drives which are to be exactly mirrored, regardless of if the entire disk is used. Also, if any logical volume to be mirrored is already mirrored, this command will fail.
-p *copyn*=*mirrorpool*	Assigns mirror pools to the copies being created. A mirror pool is assigned to a copy using the *copyn*=*mirrorpool* parameter. Specify a mirror pool for each copy. To specify more than one *copyn*=*mirrorpool* pair, provide multiple **-p** *copyn*=*mirrorpool* flags.
-Q *Quorum Keep*	By default in **mirrorvg**, when a volume group's contents becomes mirrored, volume group quorum is disabled. If the user wishes to keep the volume group quorum requirement after mirroring is complete, this option should be used in the command. For later quorum changes, refer to the **chvg** command.
-S *Background Sync*	Returns the **mirrorvg** command immediately and starts a background **syncvg** of the volume group. With this option, it is not obvious when the mirrors have completely finished their synchronization. However, as portions of the mirrors become synchronized, they are immediately used by the operating system in mirror usage.
-s *Disable Sync*	Returns the **mirrorvg** command immediately without performing any type of mirror synchronization. If this option is used, the mirror may exist for a logical volume but is not used by the operating system until it has been synchronized with the **syncvg** command.

The following is a description of **rootvg**:

Item	Description
rootvg mirroring	When the **rootvg** mirroring has completed, you must perform two additional tasks: **bosboot** and **bootlist**. The **bosboot** command is required to customize the bootrec of the newly mirrored drive. The **bootlist** command needs to be performed to instruct the system which disk and order you prefer the mirrored boot process to start.
non-rootvg mirroring	When this volume group has been mirrored, the default command causes Quorum to be deactivated.
rootvg and non-rootvg mirroring	The system dump devices, primary and secondary, should not be mirrored. In some systems, the paging device and the dump device are the same device. However, most users want the paging device mirrored. When **mirrorvg** detects that a dump device and the paging device are the same, the logical volume will be mirrored automatically. If **mirrorvg** detects that the dump and paging device are different logical volumes, the paging device is automatically mirrored, but the dump logical volume is not. The dump device can be queried and modified with the **sysdumpdev** command.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Examples

1. To triply mirror a volume group, enter:

   ```
   mirrorvg -c 3 workvg
   ```

   The logical partitions in the logical volumes held on `workvg` now have three copies.
2. To get default mirroring of rootvg, enter:

   ```
   mirrorvg rootvg
   ```

   `rootvg` now has two copies.
3. To replace a bad disk drive in a mirrored volume group, enter

   ```
   unmirrorvg workvg hdisk7
   reducevg workvg hdisk7
   rmdev -l hdisk7 -d
   replace the disk drive, let the drive be renamed hdisk7
   extendvg workvg hdisk7
   mirrorvg workvg
   ```

   Note: By default in this example, **mirrorvg** will try to create 2 copies for logical volumes in `workvg`. It will try to create the new mirrors onto the replaced disk drive. However, if the original system had been triply mirrored, there may be no new mirrors created onto `hdisk7`, as other copies may already exist for the logical volumes.
4. To sync the newly created mirrors in the background, enter:

   ```
   mirrorvg -S -c 3 workvg
   ```

5. To create a second and third copy of the logical volumes within `datavg`, where the physical partition maps on each disk match each other exactly, enter:

```
mirrorvg -m -c 3 datavg hdisk2 hdisk3
```

The logical partitions in the logical volumes held on `datavg` now have three copies.

Files

Item	Description
/usr/sbin	Directory where the **mirrorvg** command resides.

mirscan Command

Purpose

Search for and correct physical partitions that are stale or unable to perform I/O operations.

Syntax

`mirscan` `-v` *vgname* | `-l` *lvname* | `-p` *pvname* | `-r` *reverse_pvname* [`-a`] [`-o`] [`-q` *nblks*] [`-c` *lvcopy*] [`-s` *strictness*] [`-u` *upperbound*]

Description

The `mirscan` command examines each allocated partition on the specified device. A report is generated that lists whether the partition is stale or fresh, and lists whether it is capable of performing I/O operations. The LVM device driver is queried to determine whether the partition is stale or fresh. Regardless of whether the partition is stale or fresh, it is read to determine whether it is capable of performing I/O operations. By default the entire partition is read, but if the `-q` flag is specified, the *nblks* value determines how much of the partition will be read. If the `-a` flag is not specified, the report is printed and execution ends after all partitions are read.

If the `-a` flag is used, corrective action is taken after all the partitions have been examined. Stale partitions will be synced. If a partition is not capable of performing I/O, `mirscan` attempts to trigger bad block relocation or hardware relocation with a forced sync operation, which should write a good copy of the data to the block that is incapable of performing I/O operations. If the partition is still unreadable, the `mirscan` command attempts to migrate that partition to a new location. By default, the new location that is selected adheres to the strictness and upperbound policies for the logical volume that contains the partition. Using the `-s` flag causes the strictness value specified on the command line to override the natural strictness value of the logical volume that contains the partition. Similarly, using the `-u` flag causes the upperbound value specified on the command line to override the natural upperbound value of the logical volume that contains the partition.

The `mirscan` command prints (to standard output) a status report for the partitions scanned. If the `-a` flag is specified, the `mirscan` command also prints (to standard output) a status report containing each corrective action that is taken. If the `-o` flag is specified, the report will be in colon-separated output format. If the `-o` flag is not specified, the default behavior is to print the report in human-readable format.

Partitions on nonmirrored logical volumes are scanned and included in all reports, but no sync or migration operation is possible for such partitions. Partitions on striped logical volumes can be synced but cannot be migrated. Partitions on paging devices cannot be migrated, because this would result in a system hang if the `mirscan` process were to be paged out. Partitions on the boot logical volume cannot be migrated. Partitions on an active firmware-assisted dump logical volume cannot be migrated. An informative error message is generated in the corrective action report for each of the preceding cases.

By default, the `mirscan` command does not take any lock on the volume group. This should allow the `mirscan` command to run in the background without interfering with other `lvm` commands. If the `-a` flag is specified and there are partitions that need to be migrated, the volume group is locked, all the migration operations are performed, and the volume group lock is released. Therefore, if the `-a` flag is specified, the impact to other `lvm` commands is minimized because the volume group is only locked during the migration operations, which are all performed at once just before the end of execution.

Flags

Item	Description
-a	Specifies that corrective action should be taken.
-c *lvcopy*	Identifies a particular copy of the logical volume. The `-c` flag can only be specified in conjunction with the `-l` flag. The `-c` flag is ignored if it is used in conjunction with the `-p`, `-r`, or `-v` flag.
-l *lvname*	Specifies the logical volume to be scanned.
-o	Specifies colon-separated output format should be used for the report. If this option is not used, the default behavior is to print a report in human-readable format.
-p *pvname*	Specifies the physical volume to be scanned.
-q *nblks*	Specifies which portions of the partition should be read. If the *nblks* value is `0`, only the first, middle, and last 512 bytes of each partition are read to determine whether the partition is capable of performing I/O operations. A nonzero *nblks* value indicates that only the first *nblks* 512 byte blocks of each partition should be read to determine whether the partition is capable of performing I/O operations. If the `-q` flag is not specified, the entire partition is read.
-r *reverse_pvname*	Specifies that any partitions in the volume group should be scanned if they do not reside on *pvname* but they do have a mirror copy on *pvname*. This could be run prior to removing *pvname* from the system, in case *pvname* somehow has the last good copy of a partition.
-s *strictness* (y, n, s)	Specifies a strictness value that should override the natural strictness value. Legal values are `y`, `n`, and `s`, where `y` enables strictness, `n` disables strictness, and `s` enables "superstrictness." By default, when `mirscan` has to perform a migration operation on a partition it will adhere to the natural strictness value of the logical volume that contains that partition. If the `-s` flag is used, the override strictness value will be used. If the `-s` flag is used in conjunction with the `-p`, `-r`, or `-v` flags, the override strictness value could override the natural strictness of multiple logical volumes.
-u *upperbound*	Specifies an upperbound value that should override the natural upperbound value. The upperbound value should be between `1` and the total number of physical volumes in the volume group. By default, when `mirscan` has to perform a migration operation on a partition it will adhere to the natural upperbound value of the logical volume that contains the partition. If the `-u` flag is used, the override upperbound value will be used. If the `-u` flag is used in conjunction with the `-p`, `-r`, or `-v` flags, the override upperbound value could override the natural upperbound value of multiple logical volumes.
-v *vgname*	Specifies the volume group to be scanned.

Exit Status

An exit code of `0` indicates that `mirscan` was able to complete its execution and was able to correct any error conditions that were encountered along the way. An exit code of `1` indicates that `mirscan` was able to complete its execution, but it was unable to correct every error that it found; further corrective action

is still required. For example, if corrective actions would be required but the `-a` flag was not specified, an exit code of 1 is used. An exit code of 2 indicates that `mirscan` was unable to complete its execution. For example, if the target device is not listed in the ODM, an exit code of 2 is used.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To scan logical volume lv33, report the status of each partition, and have every block of each partition read to determine whether it is capable of performing I/O operations, type:

   ```
   mirscan -l lv33
   ```

2. To scan logical volume lv33, report the status of each partition, and have only the first two blocks of each partition read to determine whether it is capable of performing I/O operations, type:

   ```
   mirscan -l lv33 -q 2
   ```

3. To scan logical volume lv33, report the status of each partition, sync any stale partitions found, and migrate any partitions that are not capable of performing I/O operations, type:

   ```
   mirscan -l lv33 -a
   ```

4. To scan every allocated logical partition on hdisk4 and report the status of each partition, type:

   ```
   mirscan -p hdisk4
   ```

5. To find every allocated partition in the volume group that resides on hdisk4, and scan and report the status of all partitions that do not reside on hdisk4 but are mirror copies of a partition that resides on hdisk4, type:

   ```
   mirscan -r hdisk4
   ```

   This would be useful to run before removing hdisk4 from the system.

6. To scan volume group vg05, report the status of each allocated partition, and have the first, middle, and last 512 bytes of each partition read to determine whether that partition is capable of performing I/O operations, type:

   ```
   mirscan -v vg05 -q 0
   ```

Restrictions

Unmirrored partitions and striped partitions are not eligible for migration. Partitions on paging devices will not be migrated by mirror scan because it would result in a system hang if the `mirscan` process happened to get paged out. Partitions from the boot logical volume cannot be migrated.

Location

/usr/sbin/mirscan

Standard Output

Each line in the report corresponds to an operation on a physical partition. There are 4 types of operation that `mirscan` can perform. A *scan operation* determines whether the partition is synced and whether

it is capable of performing I/O operations. A *resync operation* is a corrective action performed on stale partitions that attempts to return them to synced state. A *force resync operation* is a corrective action performed on partitions that are not capable of performing I/O operations, in an attempt to trigger bad block relocation or hardware relocation. At the end of the force resync operation, the partition is read again to determine whether it is capable of performing I/O operations. A *migration operation* is a corrective action performed on partitions that are not capable of performing I/O operations, in an attempt to move the data to a physical location that is capable of performing I/O.

The default format for the reports contains the following column headings. If the *-o* flag is specified, no header is displayed and the output report is printed in colon-separated output format. The columns and their meanings are as follows:

Item	Description
OP	The valid values for this field are s, r, f, and m. A value of s signifies a scan operation. A value of r signifies a resync operation. A value of f signifies a force resync operation, which is performed in an effort to trigger bad block relocation or hardware relocation. A value of m signifies a migration operation.
STATUS	The valid values for this field are SUCCESS or FAILURE. For a scan operation, FAILURE is indicated if the partition being scanned is stale or incapable of performing I/O. For a resync operation, FAILURE is indicated if the partition was not synchronized. For a force resync operation, FAILURE is indicated if the partition is still incapable of performing I/O operations. For a migration operation, FAILURE is indicated if the migration operation was not completed.
PVNAME	Identifies the name of the physical volume where the partition being operated on resides. For a migration operation, PVNAME refers to the source physical volume and TARGETPV refers to the destination physical volume.
PP	Identifies the physical partition number of the partition being operated on. The first partition on a particular physical volume has a PP value of 1, not 0.
SYNC	The valid values for this field are synced or stale. The value indicated refers to the state of the partition after the operation has been completed. For example, if a resync operation succeeds, a value of synced will be displayed.
IOFAIL	The valid values for this field are yes or no. The value indicated refers to the state of the partition after the operation has been completed. For example, if a migration operation succeeds then a value of no is displayed to indicate that the partition no longer has a problem performing I/O operations.
LVNAME	Identifies the name of the logical volume where the partition being operated on resides.
LP	Identifies the logical partition number of the partition being operated on. The first partition on a particular logical volume has an LP value of 1, not 0.
CP	Identifies the logical copy number of the partition being operated on. The first logical copy of a logical volume has a CP value of 1, not 0.
TARGETPV	Identifies the name of the physical volume that was used as the target for a migration operation. For any type of operation other than a migration operation, this field is left blank.
TARGETPP	Identifies the physical partition number of the partition that was used as the target for a migration operation. For any type of operation other than a migration operation, this field is left blank. The first partition on a particular physical volume has a TARGETPP value of 1, not 0.

mkauth Command

Purpose

Creates a new user-defined authorization.

Syntax

mkauth [**-R** *load_module*] [*Attribute* **=** *Value* ...] *Name*

Description

The **mkauth** command creates a new user-defined authorization in the authorization database. You can create authorization hierarchies by using a dot (.) in the *Name* parameter to create an authorization of the form *ParentAuth*.*SubParentAuth*.*SubSubParentAuth*.... All parent elements in the *Name* parameter must already exist in the authorization database before the new authorization is created. The maximum number of parent elements that you can use to create an authorization is 8.

If the system is configured to use multiple domains for the authorization database, the new authorization is created in the first domain specified by the **secorder** attribute in the authorizations stanza of the **/etc/nscontrol.conf** file. Use the **-R** flag to create an authorization in a specific domain.

Authorization attributes can be set at creation time through the *Attribute* **=** *Value* parameter. Every authorization that you create must have a value for the **id** authorization attribute. If you do not specify the value using the **mkauth** command, the command automatically generates a unique ID for the authorization. If you specify an ID, the value must be unique and greater than 10000.

Restriction: Authorization IDs less than 10000 are reserved for system-defined authorizations

.

When the system is operating in enhanced Role Based Access Control (RBAC) mode, modifications made to the authorization database are not used for security considerations until the database is sent to the kernel security tables using the **setkst** command. Authorizations created in the authorization database can be assigned to roles immediately, but do not take effect until the kernel security tables is updated.

Flags

Item	Description
-R *load_module*	Specifies the loadable module to use for authorization creation.

Parameters

Item	Description
Attribute **=** *Value*	Initializes an authorization attribute. Refer to the **chauth** command for valid attributes and values.

Item	Description
Name	Specifies a unique authorization name string. **Restrictions on Creating Authorization Names:** The *Name* parameter that you specify must be unique, and can be a maximum of 63 single-byte printable characters. Although the **mkauth** command supports multibyte authorization names, authorization names to characters are restricted within the POSIX portable file name character set. The authorization name that you specify cannot begin with `aix.` because that is the designated top-level parent for system-defined authorizations and the **mkauth** command only creates user-defined authorizations. Authorization names must not begin with a dash (-), a plus sign (+), an at sign (@), a tilde (~), or contain any space, tab or newline characters. You cannot use the keywords **ALL**, **default**, **ALLOW_OWNER**, **ALLOW_GROUP**, **ALLOW_ALL**, or an asterisk (*) as an authorization name. Additionally, do not use any of the following characters within an authorization string: • : (colon) • " (quotation mark) • # (number sign) • , (comma) • = (equal sign) • \ (backslash) • / (forward slash) • ? (question mark) • ' (single quotation mark) • ` (grave accent)

Security

The **mkauth** command is a privileged command. You must assume a role that has the following authorization to run the command successfully.

Item	Description
aix.security.auth.create	Required to run the command.

Files Accessed

Item	Description
File	Mode
/etc/security/authorizations	rw

Examples

1. To create a top-level authorization `custom` and have the **mkauth** command assign an appropriate `ID` value, use the following command:

```
mkauth custom
```

2. To create a `custom.test` child authorization and assign an ID and default description, use the following command:

```
mkauth id=12000 dfltmsg="Test Authorization" custom.test
```

3. To create the `custom` authorization in LDAP, use the following command:

```
mkauth -R LDAP custom
```

mkboot Command

Purpose

Creates the boot image, the boot record, and the service record. This command is *not* a user-level command and is NOT supported.

Syntax

mkboot **-d** *Device* [**-b**] [**-D**] [**-c**] [**-h**] [**-i**] [**-I**] [**-l** *LVDev*] { **-k** *Kernel* | **-e** *Expander* } [**-L**] [**-s**] [**-r**] [**-p** *Offset*] [**-w**] **-f** *FileSystem*

Description

The **mkboot** command combines a kernel and file system into a boot image. The resulting image is written to standard out. It is copied to a boot device with the appropriate boot record information. The boot image can be made compressed or uncompressed and with or without a boot record at the beginning of the image. An image created for a tape is compressed with the boot record at the start of the image file. A disk boot image may be created without compression and has no boot record. The boot record is written to the first sector of the disk. The record contains information about the size and location of the image after it is written to the boot logical volume on that disk.

If the boot logical volume is mirrored, the **mkboot** command not only writes the boot image to each copy of the boot logical volume but also writes a boot record to each physical disk comprising the mirror. As long as the **mkboot** command is able to update at least one of the copies of a mirrored boot logical volume, no error is returned. To enable booting from each copy of a mirrored boot logical volume, each of the physical disks must be specified using the **bootlist** command.

The **mkboot** command is usually called by the **bosboot** command. However, you can run the **mkboot** command a second time to put expand code at the beginning of a compressed boot image.

Flags

Item	Description
-b	Zeros out save-base fields. This flag is optional.
-d *Device*	Specifies the device required for the IPL record. This flag is required.
-c	Zeros out the boot record on the device. This flag is optional.
-D	Loads the low-level debugger at boot time.
-e *Expander*	Specifies kernel expansion code to create a compressed boot image file. Either the **-e** flag or the **-k** flag must be specified.
-f *FileSystem*	Specifies the boot file system. This flag is required.
-h	Prevents the **mkboot** command from updating the boot header. This flag is optional.

Item	Description
-i	Writes the normal portion of the boot record.
-I (uppercase i)	Invokes the low-level debugger at boot time.
-k *Kernel*	Specifies the kernel in the boot image. Either the **-k** flag or the **-e** flag must be specified.
-l (lowercase L) *LVDev*	Specifies the logical volume device that contains the loadable boot code.
-L	Enables lock instrumentation for MP systems. This flag has no effect on systems that are not using the MP kernel.
-p *Offset*	Specifies the address to use as `boot_pr_start` field in the boot record. This flag is used in creating the CD-ROM boot image. This flag is optional.
-r	Creates an image that is read-only storage (ROS) emulation code.
-s	Writes the service portion of the boot record.
-w	Outputs first two blocks of boot logical volume before the boot image. This flag is applicable to disk boot image only.

Security

Access Control: Only the root user can read and execute this command.

Examples

1. To create an uncompressed boot image, using the kernel **/usr/lib/boot/unix** and the **/tmp/bootfs** file system for the device **/dev/hdisk0**, enter

   ```
   mkboot -d /dev/hdisk0 -k /usr/lib/boot/unix -f /tmp/bootfs \
   -b -i -s > /tmp/boot.image
   ```

2. To clear the boot record but leave the PVID for disk **hdisk0**, enter:

   ```
   mkboot -d /dev/hdisk0 -c
   ```

3. Although the **mkboot** command combines a kernel and a random access memory (RAM) file system to create one boot image, you can run the **mkboot** command a second time to put expand code at the beginning of a compressed boot image. For example, enter:

   ```
   mkboot -b -d /dev/rmt0 -k unix -f ramfs | compress > /tmp/image
   mkboot -b -i -s -d /dev/rmt0 -k bootexpand -f /tmp/image \
   > bootfile
   ```

   for a bootable tape where:

Item	Description
`unix`	Specifies the kernel.
`ramfs`	Specifies the RAM disk file system.
`compress`	Specifies the compression or compact routine.
`bootexpand`	Specifies the expansion or kernel uncompact routine.

Files

Item	Description
/usr/include/sys/bootrecord.h	Specifies the structure of the boot record.

mkC2admin Command

Purpose

Configure a system to operate in C2 Security Mode.

Syntax

mkC2admin { [**-m**] | [**-a** *address*] *hostname* }

Description

The **mkC2admin** command initializes the security directories for use in a C2 System configuration. The distributed database directories are created and symbolic links initialized. When a system is being configured as the Administrative Host (using the **-m** flag), an additional file system is created to hold the master copies of the administrative database files. Those files are stored in the directory **/etc/data.master** which has a logical volume name of **hd10sec**.

The administrative database files are divided into three categories. Those files that must be shared, those files that optionally may be shared, and those files that may not be shared. Optionally sharable files are described in the file **/etc/security/files.config**. That file consists of multiple lines of the format:

```
[y|n]|filename
```

and is editable by the administrator. To select an optionally sharable filename, the administrator sets the first field to the value **y**. To make an optionally sharable file be unshared, the field is set to the value **n**. All hosts in the C2 System must have an identical **/etc/security/files.config** file.

The system *hostname* must be defined in the **/etc/hosts** file at the time this command is run. If not, the IP address of the new C2 System Administrative Host may be provided with the **-a** option, and an entry will be added to **/etc/hosts**.

Flags

Item	Description
-a *address*	Use address as the IP address of *hostname*.
-m	Configure the host as the administrative master.

Parameters

Item	Description
hostname	Specifies the hostname.

Exit Status

0
The system has been properly configured to operate in the C2 mode.

1
The system was not installed with the C2 option.

2
The system could not be successfully configured to operate in C2 mode.

3
The system was previously configured to operate in C2 mode without having first been unconfigured.

Files

Item	Description
/usr/sbin/mkC2admin	Contains the mkC2admin command.

mkcatdefs Command

Purpose

Preprocesses a message source file.

Syntax

mkcatdefs *SymbolName SourceFile* ... [**-h**]

Description

The **mkcatdefs** command preprocesses a message source file for input to the **gencat** command.

The *SourceFile* message file contains symbolic identifiers. The **mkcatdefs** command produces the *SymbolName*_**msg.h** file, containing statements that equate symbolic identifiers with the set numbers and message ID numbers assigned by the **mkcatdefs** command.

The **mkcatdefs** command creates two outputs. The first is a header file called *SymbolName*_**msg.h**. You must include this *SymbolName*_**msg.h** file in your application program to associate the symbolic names to the set and message numbers assigned by the **mkcatdefs** command.

The **mkcatdefs** command sends message source data, with numbers instead of symbolic identifiers, to standard output. This output is suitable as input to the **gencat** command. You can use the **mkcatdefs** command output as input to the **gencat** command in the following ways:

- Use the **mkcatdefs** command with a **>** (redirection symbol) to write the new message source to a file. Use this file as input to the **gencat** command.
- Pipe the **mkcatdefs** command output file directly to the **gencat** command.
- Use the **runcat** command rather than the **mkcatdefs** command. The **runcat** command automatically sends the message source file through the **mkcatdefs** command and then pipes the file to the **gencat** command.

After running the **mkcatdefs** command, you can use symbolic names in an application to refer to messages.

Flags

Item	Description
-h	Suppresses the generation of a *SymbolName*_**msg.h** file. This flag must be the last argument to the **mkcatdefs** command.

Examples

To process the `symb.msg` message source file and redirect the output to the `symb.src` file, enter:

```
mkcatdefs symb symb.msg > symb.src
```

The generated `symb_msg.h` file looks similar to the following:

```
#ifdef      _H_SYMB_MSG
#define     _H_SYMB_MSG
#include <limits.h>
#include <nl_types.h>
#define      MF_SYMB     "symb.cat"
/* The following was generated from symb.src. */
/* definitions for set MSFAC */
#define      SYM_FORM     1
#define      SYM_LEN      2
#define      MSG_H  6
#endif
```

The **mkcatdefs** command also creates the `symb.src` message catalog source file for the **gencat** command with numbers assigned to the symbolic identifiers:

```
$quote " Use double quotation marks to delimit message text
$delset 1
$set 1
1       "Symbolic identifiers can only contain alphanumeric \
characters or the _ (underscore character)\n"
2       "Symbolic identifiers cannot be more than 65 \
characters long\n"
5       "You can mix symbolic identifiers and numbers\n"
$quote
6       remember to include the "msg_h" file in your program
```

The assigned message numbers are noncontiguous because the source file contained a specific number. The **mkcatdefs** program always assigns the previous number plus 1 to a symbolic identifier.

> **Note:** The **mkcatdefs** command inserts a **$delset** command before a **$set** command in the output message source file. This means you cannot add, delete, or replace single messages in an existing catalog when piping to the **gencat** command. You must enter all messages in the set.

Files

Item	Description
/usr/bin/mkcatdefs	Contains the **mkcatdefs** command.

mkCCadmin Command

Purpose

Configure a system to operate in Common Criteria enabled Security Mode.

Syntax

mkCCadmin { [**-m**] | [**-a** *address*] *hostname* }

Description

The **mkCCadmin** command initializes the security directories for use in a Common Criteria enabled System configuration. The distributed database directories are created and symbolic links initialized. When a system is being configured as the Administrative Host (using the **-m** flag), an additional file

system is created to hold the master copies of the administrative database files. Those files are stored in the directory **/etc/data.master** which has a logical volume name of **hd10sec**.

The administrative database files are divided into three categories. Those files that must be shared, those files that optionally may be shared, and those files that may not be shared. Optionally sharable files are described in the file **/etc/security/files.config**. That file consists of multiple lines of the format:

```
[y|n]|filename
```

and is editable by the administrator. To select an optionally sharable filename, the administrator sets the first field to the value **y**. To make an optionally sharable file be unshared, the field is set to the value **n**. All hosts in the Common Criteria enabled System must have an identical **/etc/security/files.config** file.

The system *hostname* must be defined in the **/etc/hosts** file at the time this command is run. If not, the IP address of the new Common Criteria enabled System Administrative Host may be provided with the **-a** option, and an entry will be added to **/etc/hosts**.

Flags

Item	Description
-a *address*	Use address as the IP address of *hostname*.
-m	Configure the host as the administrative master.

Parameters

Item	Description
hostname	Specifies the hostname.

Exit Status

0
The system has been properly configured to operate in the Common Criteria enabled mode.

1
The system was not installed with the Common Criteria enabled option.

2
The system could not be successfully configured to operate in Common Criteria enabled mode.

3
The system was previously configured to operate in Common Criteria enabled mode without having first been unconfigured.

Files

Item	Description
/usr/sbin/mkCCadmin	Contains the mkCCadmin command.

mkcd Command

Purpose

Creates multi-volume CDs from a **mksysb**, **savevg**, or **savewpar** backup image.

Syntax

mkcd **-r** *directory* | **-d** *cd_device* | **-S** [**-m** *mksysb_image* | **-M** *mksysb_target* | **-s** *savevg_image* | **-v** *savevg_volume_group* | **-w** *savewpar_image* | **-W** *wparname*] [**-C** *cd_fs_dir*] [**-I** *cd_image_dir*] [**-V** *cdfs_volume_group*] [**-B**] [**-p** *pkg_source_dir*] [**-R** | **-S**] [**-i** *image.data*] [**-u** *bosinst.data*] [**-f** *wparspecificationfile*] [**-e**] [**-P**] [**-l** *package_list*] [**-L**][**-b** *bundle_file*] [**-z** *custom_file*] [**-D**] [**-U**] [**-Y**] [**-n**] [**-a**] [**-A**] [**-c**] [**-Z**] [**-G** |**-N**] [**-x** file] [**-T**]

Description

The **mkcd** command creates a system backup image (**mksysb**) to CD-Recordable (CD-R) or DVD-Recordable (DVD-R, DVD-RAM) from the system **rootvg** or from a previously created **mksysb** image. It creates a volume group backup image (**savevg**) to CD-R from a user-specified volume group or from a previously created **savevg** image. It also creates the backup image of a workload partition (**savewpar**) to CD or DVD from a user-specified workload partition or from a previously created **savewpar** image.

Note: If the system has a **multibos** environment where both instances are mounted, the only way to restore the backup is by using the **alt_disk_mksysb** command.

For DVD media, system backups that are made with the **mkcd** command have a limitation in that they expect the media to be 4.7 GB or larger per side. The **mkcd** command does not process the next volume until it writes over 4 GB on the current volume, thus the use of smaller media would result in corruption when you go beyond the capacity of the media.

When a bootable backup of a root volume group is created, the boot image reflects the currently running kernel. If the current kernel is the 64-bit kernel, the backup boot image is also 64 bit, and it boots 64-bit systems only. If the current kernel is a 32-bit kernel, the backup boot image is 32 bit, and it can boot both 32-bit and 64-bit systems.

With the **mkcd** command, you can create bootable and non-bootable CDs in Rock Ridge (ISO9660) or UDF (Universal Disk Format) format.

See the **-L** flag for details about creating DVD-sized images. What applies to CDs also applies to DVDs, except where noted.

Note: The functionality that is required to create Rock Ridge format CD images and to write the CD image to the CD-R, DVD-R or DVD-RAM device is not part of the **mkcd** command. You must supply additional code to **mkcd** command to do these tasks. The code is called by using shell scripts and then linked to **/usr/sbin/mkrr_fs** (for creating the Rock Ridge format image) and **/usr/sbin/burn_cd** (for writing to the CD-R device). Both links are called from the **mkcd** command.

Some sample shell scripts are included for different vendor-specific routines. You can find these scripts in **/usr/samples/oem_cdwriters**.

If you do not supply any file systems or directories as command parameters, the **mkcd** command creates the necessary file systems and removes them when the command finishes executing. File systems that you supply are checked for adequate space and write access.

Note:

1. While the **mkcd** command is running, ensure that system activity is minimal.
2. If the **mkcd** command creates file systems in the backup volume group, they are excluded from the backup.

If you want to create multi-volume CDs because the volume group image does not fit on one CD, the **mkcd** command gives instructions for CD replacement and removal until all the volumes have been created.

Flags

Item	Description
-a	Does not back up extended attributes or NFS4 ACLs.
-A	Backs up DMAPI file system files.

Item	Description
-b *bundle_file*	Gives the full path name of the file that contains a list of filesets to be installed after the **mksysb** is restored. This file is copied to **./usr/sys/inst.data/user_bundles/***bundle_file* in the CD file system and also copied to RAM in case the CD is unmounted. The file would be listed as BUNDLES=**/../usr/sys/inst.data/user_bundles/***bundle_file* in the **bosinst.data** file.
-B	Prevents the **mkcd** command from adding boot images (non-bootable CD) to the CD. Use this flag if you want to create a **mksysb** CD that you will not boot. Before you install the non-bootable **mksysb** CD, you must boot a same level (V.R.M.) product CD. The **mkcd** command defaults to creating a bootable CD for the machine type of the source system. See Notes for details.
-c	Does not compress or pack files as they are backed up.
-C *cd_fs_dir*	Specifies the file system that is used to create the CD file system structure, which must have at least 645 MB of available disk space (up to 4.38 GB for DVD sized images). The CD image consumes only as much room as is necessary to contain all the data on the CD. If you do not specify the **-C** flag and the **/mkcd/cd_fs** directory exists, the **mkcd** command uses that directory. If you do not specify the **-C** flag and the **/mkcd/cd_fs** directory does not exist, the **mkcd** command creates the file system **/mkcd/cd_fs** and removes it when the command finishes running. The command creates the file system in the volume group that is indicated with the **-V** flag, or **rootvg** if that flag is not used. Each time that you invoke the **mkcd** command, a unique subdirectory (that uses the process id) is created under the **/mkcd/cd_fs** directory, or in the directory that is specified with the **-C** flag. **Note:** If performing DVD sized backups, the file systems must be *large file enabled*. This type of backup also requires setting the file **ulimit** size to *unlimited*.
-d *cd_device*	Indicates the CD-R, DVD-R, or DVD-RAM device (`/dev/cd1`, for instance). This flag is required unless you use the **-S** flag.
-D	Turns on the debug output information feature. The default is no debug output.
-e	Excludes the files and directories from the backup image that is listed in the **/etc/exclude.***volume_group* file. You cannot use this flag with the **-m** or **-s** flags.
-f *wparspecificationfile*	Specifies the user-supplied WPAR specification file. This specification file of workload partition takes precedence over the **wpar.spec** file in the **savewpar** image. If you do not use the **-f** flag, the **mkcd** command restores the **wpar.spec** file from the specified **savewpar** image, or generates a new **wpar.spec** file during the creation of **savewpar**.

Item	Description
-i *image.data*	Specifies the user-supplied *image.data* file. This data file takes precedence over the **image.data** file in the **mksysb** image. If you do not give the **-i** flag, then **mkcd** restores the **image.data** from the specified **mksysb** image, or generates a new **image.data** file during the creation of **mksysb**. **Note:** The **-i** flag cannot be used to specify a user-supplied *vgname.data* file for use with a **savevg** image.
-I *cd_image_dir*	Specifies the directory or file system where the final CD images are stored before they are written to the CD-R, DVD-R, or DVD-RAM device. If this flag is not used, **mkcd** uses the **/mkcd/***cd_images* directory if it already exists. If not, the command creates the **/mkcd/***cd_images* file system in the volume group that is given with the **-V** flag, or in **rootvg** if that flag is not used. If the **mkcd** command creates the file system, it is removed upon command completion, unless either the **-R** or **-S** flag is used. If the **-R** or **-S** flag is used, consideration must be made for adequate file system, directory, or disk space, especially when you create multi-volume CDs. The CD image consumes only as much room as is necessary to contain all the data on the CD. **Note:** If performing DVD sized backups, the file systems must be *large file enabled*. This type of backup also requires setting the file **ulimit** size to *unlimited*.
-l *package_list*	Specifies the file that contains a list of additional packages you want copied to the **./usr/lpp/inst.images** directory of the CD file system. The images are copied from the location that is named with the **-p** flag. If you use the **-l** flag, you must also use the **-p** flag.
-L	Creates final CD images that are DVD sized (up to 4.38 GB).
-m *mksysb_image*	Specifies a previously created **mksysb** image. If you do not specify the **-m** flag, the **mkcd** command calls **mksysb**. For more information about where the **mksysb** image is placed, see the **-M** flag.
-M *mksysb_target*	States the directory or file system where the **mksysb** or **savevg** image is stored if a previously created backup is not given with the **-m** or **-s** flags. If the **-M** flag is not used and a **mksysb** or **savevg image** is not provided, the **mkcd** command verifies that **/mkcd/***mksysb_image* exists. If the directory does not exist, then the **mkcd** command creates a separate file system, **/mkcd/***mksysb_image*, where the **mksysb** or **savevg** images are temporarily stored. The command creates the file system in the volume group that is given with the **-V** flag, or in **rootvg** if that flag is not used. **Note:** If performing DVD sized backups, the file systems must be *large file enabled*. This also requires setting the file **ulimit** size to *unlimited*.

Item	Description
-n	Backs up user volume group information and administration data files. This backs up files such as **/tmp/vgdata/vgname/vgname.data** and map files, if any exist. This flag does not back up user data files. This backup can be used to create a user volume group without restoring user data files. This action cannot be done to rootvg.
-N	Includes file systems that belong to a workload partition (WPAR) in the defined state in the system backup. **Note:** To be included in the backup, all file systems that belong to a WPAR in the defined state must be in the rootvg volume group.
-p *pkg_source_dir*	Names the directory or device that contains device and kernel package images. The device must be a CD device (for example, `/dev/cd0`). If you use the same CD-R, DVD-R, or DVD-RAM device that you gave with the **-d** flag, the product CD media must be inserted into the CD-R drive first. The **mkcd** command then prompts you to insert the writeable CD before the actual CD creation.
-P	Creates physical partition mapping during the **mksysb** or **savevg** creation. You cannot use this flag with the **-m** or **-s** flags.
-r *directory*	Indicates existing directory structure to burn onto a CD or DVD. This flag makes a CD image that is a copy of the specified directory structure.
-R	Prevents the **mkcd** command from removing the final CD images. **mkcd** defaults by removing everything that it creates when it finishes running. The **-R** flag allows multiple CD image sets to be stored, or for CD creation (burn) to occur on another system. If multiple volumes are needed, the final images are uniquely named by using the process ID and volume suffixes.
-s *savevg_image*	Indicates a previously created **savevg** image. All `savevg` backup images are nonbootable. See Notes for details.
-S	Stops the **mkcd** command before it writes to the CD-R, DVD-R, or DVD-RAM without removing the final CD images. The **-S** flag allows multiple CD sets to be created, or for CDs to be created on another system. The images remain in the directory marked by the **-I** flag, or in the **/mkcd/***cd_images* directory if the **-I** flag is not used. If multiple volumes are required, the final images are uniquely named by using the process ID and volume suffixes.

Item	Description
-T	Creates backup by using snapshots. This command applies only to JFS2 file systems. When you specify the **-T** flag to use snapshots for creating a volume group backup, external JFS2 snapshots are created. Snapshots allow for a point-in-time image of a JFS2 file system and thus, do not require a system to be put into a temporarily inactive state. The size of the snapshot is 2% - 15% of the size of the file system. The snapshot logical volumes are removed when back up is complete. However, snapshots are not removed if a file system already has other snapshots. Additionally, if a file system has internal snapshots, external snapshots cannot be created and thus, snapshots are not used for creating the backup of the file system. The use of the **-T** flag does not affect any JFS file systems that are present in the volume group that is being backed up. These file systems are backed up in the same manner as done previously.
-u *bosinst.data*	Specifies the user-supplied *bosinst.data* file. This data file takes precedence over the **bosinst.data** file in the **mksysb** image. If you do not give the **-u** flag, then the **mkcd** command restores **bosinst.data** from the specified **mksysb** image, or generates a new **bosinst.data** file during the creation of **mksysb**.
-U	Creates a **UDF** (Universal Disk Format) file system on DVD-RAM media. It does not require the amount of free space that is needed to create Rock Ridge format backups. It does not need the **/mkcd/cd_fs** and **/mkcd/cd_images** file systems. Therefore, the only temporary disk space it needs is to create the backup image that will be copied to the media. This means that the **-I** and **-C** flags do not apply to the **-U** flag. Because the backup is copied to the media, images cannot be created and burned later or on another system. So, the **-R** flag and **-S** flag do not apply when you use the **-U** flag. You must specify a device to write to with the **-d** flag. The **-U** flag does not use the **/usr/sbin/mkrr_fs** or **/usr/sbin/burn_cd** file systems.
-v *savevg_volume_group*	Denotes the volume group to be backed up using the **savevg** command. All savevg backup images are nonbootable. See Notes for details. For more information about where the **savevg** image is placed, see the **-M** flag.
-V *cdfs_volume_group*	Indicates the volume group that is used when you create the file systems needed for the **mkcd** command. If the **-V** flag is not given and a file system is needed but not there (because it was not supplied with other flags), then **rootvg** is the default volume group for creating the file systems. If the **mkcd** command creates the file systems in the backup volume group, those file systems are not included as part of the backup image. **mkcd**-created file systems are removed upon completion of the command.
-w *savewpar_image*	Indicates a previously created **savewpar** image.
-W *wparname*	Denotes the workload partition to be backed up using the **savewpar** command.
-Y	Accepts licenses.

Item	Description
-z *custom_file*	States the full path name of the file to be copied to the root directory of the CD file system. This file could be a customization script that is specified in the **bosinst.data** file, such as `CUSTOMIZATION_FILE=filename`. For example: If the file `my_script` is in **/tmp** on the machine where the **mkcd** command is running, then enter `-z/tmp/my_script` and specify `CUSTOMIZATION_FILE=my_script`. The code copies the script to the root directory of the RAM file system before it runs.
-Z	Specifies that the Encrypted file system (EFS) information for all the files, directories, and file systems is not backed up.
-G	Excludes WPAR file systems from the system backup. The flag is not valid with **-N** flag.
-x *file*	Excludes the file systems that are listed in the file from the system backup. File system mount points must be listed one per line.

Note: Use care when you exclude file systems as a resulting backup can be unusable for system restoration.

Note:

1. If you are creating a non-bootable CD (by using the **-B** flag), you cannot use the **-p** or **-l** flags.
2. If you are creating a non-bootable CD with a **savevg** image (by using the **-s** or **-v** flags), you cannot use the **-p**, **-l**, **-u**, **-i**, **-z**, or **-b** flags.

Examples

1. To generate a bootable system backup to the CD-R device named `/dev/cd1`, enter the following command:

   ```
   mkcd -d /dev/cd1
   ```

2. To generate a system backup to the DVD-R or DVD-RAM device named `/dev/cd1`, enter the following command:

   ```
   mkcd -d /dev/cd1 -L
   ```

3. To generate a non-bootable volume group backup of the volume group `myvg` to `/dev/cd1`, enter the following command:

   ```
   mkcd -d /dev/cd1 -v myv
   ```

   Note: All **savevg** backup images are non-bootable.

4. To generate a non-bootable backup of the workload partition `mywpar` to **/dev/cd1**, enter the following command:

   ```
   mkcd -d /dev/cd1 -W mywpar
   ```

   Note: All **savewpar** backup images are not bootable.

5. To generate a non-bootable backup of the workload partition `mywpar` to **/dev/cd1** from the previously generated **savewpar** image **/wparbackups/mywpar.bff**, enter the following command:

   ```
   mkcd -d /dev/cd1 -w /wparbackups/mywpar.bff
   ```

6. To create a CD or DVD that duplicates an existing directory structure such as:

```
/mycd/a
/mycd/b/d
/mycd/c/f/g
```

enter the following command:

```
mkcd -r /mycd -d /dev/cd1
```

After you mount with `mount -o ro /dev/cd1 /mnt`, **cd** to `/mnt`; a `find . -print` command displays:

```
./a
./b
./b/d
./c
./c/f
./c/f/g
```

Files

Item	Description
/usr/bin/mkcd	Contains the **mkcd** command.

mkcfsmnt Command

Purpose

Mounts a CacheFS directory.

Syntax

mkcfsmnt **-d** *PathName* **-t** { **nfs** | **cdrom**}] [**-h** *RemoteHost*] [**-p** { *RemoteDirectory* | *LocalDeviceName* }] [**-c** *CacheDirectory*] [**-o** *MountOptions*] [**-b** *BackingFileSystem*] [**-I** | **-B** | **-N**]

Description

The **mkcfsmnt** command constructs an entry that will be appended to the **/etc/filesystems** file, thus making a file system available for use as a cache file system. If the mount is to be permanent, this entry will remain. If the mount is temporary, the flags will be used directly for the **mount** command. CacheFS file systems are used to cache accesses to backing file systems. Backing file systems are generally NFS mounts.

Flags

Item	Description
-d *PathName*	Specifies the mount point for the cache directory.
-t	Selects file systems to be cached. **nfs** Specifies that the CacheFS file system is backed by an NFS mount. **cdrom** Specifies that the CacheFS file system is backed by a CDROM file system. (Currently not supported.)

Item	Description
-h *RemoteHost*	Specifies the NFS server that is exporting the directory.
-p *RemoteDirectory*	Specifies the directory that is mounted on the path name specified. This is commonly a remote file system that will be mounted via NFS or a local device name in the case of CDROM (Currently not supported.)
-c *CacheDirectory*	Specifies the location of the CacheFS file system. This must have been previously created by execution of the **cfsadmin** command.
-d *RemoteDirectory*	Specifies the directory that is mounted on the path name specified.
-o *MountOptions*	Specifies a comma-separated string of mount options that are dependent on the backing file system type. For instance, if it is NFS, the options would be those typically specified by the **-o** Options string to mount. See the **mount** command documentation for the acceptable values.
-b *BackingFileSystem*	Specifies a backing file system if it is already mounted. If this is not specified, then the command will do the mount itself on a temporary mount point. If this is not specified, then RemoteHost and RemoteDirectory must be specified.
-I	Causes an entry to be added to the **/etc/filesystems** file. The directory is not mounted.
-B	Adds an entry to the **/etc/filesystems** file and attempts to mount the file system. This flag is the default.
-N	Mounts the directory with the options specified, but does not modify the **/etc/filesystems** file.

Examples

To specify a CacheFS mount, type:

```
/usr/sbin/mkcfsmnt -t nfs -d /usr/share/man -p /usr/share/man -h host1 -c
/cache/cache1 -o ro, intr -N
```

In this example, the **mkcfsmnt** command caches the remote directory **/usr/share/man** that resides on `host1` on the local **/usr/share/man** directory. The cache is kept in **/cache/cache1**,

which was created with the **cfsadmin** command. CacheFS takes care of doing the NFS backing mount, because the **-b** flag has not been specified.

```
/usr/sbin/mkcfsmnt -t nfs -d /usr/share/man -p /usr/share/man -h host1 -c /cache/cache1
 -b /backs/man -o ro, intr -N
```

In this example, the **mkcfsmnt** command caches the remote directory **/usr/share/man** residing on `host1` on the local **/usr/share/man** directory. The cache is kept in **/cache/cache1**,

which was created with the **cfsadmin** command. The backing file system has already been mounted on **/backs/man**.

Files

Item	Description
/etc/filesystems	Lists the remote file systems to be mounted during the system restart.

mkcifscred Command

Purpose

Adds CIFS credentials to the `/etc/cifs_fs/cifscred` file to allow future mounting of CIFS shares with stored credentials.

Syntax

`mkcifscred -h` *RemoteHost* `-u` *user* [`-p` *password*]

Description

The `mkcifscred` command takes a server and user name as input, and prompts for a password. The password is encrypted, and the credentials are stored in the `cifscred` file. If the password is not passed in with the `-p` option when mounting to a CIFS server, the credentials are either retrieved from the `cifscred` file, or, if the credentials do not exist in `cifscred`, the password is prompted for and read in as hidden input.

The credentials are stored as a server/user/password set. Multiple sets of credentials for the same server are permitted with different user names. Multiple sets with the same user name on different servers are also permitted.

Flags

Item	Description
`-h` *RemoteHost*	Specifies the name of the remote host (CIFS server). This can be provided as a host name, an IP address, or a fully qualified domain name.
`-p` *password*	Specifies the password for a particular user on a particular remote host.
`-u` *user*	Specifies the user name whose credentials are being defined for access to the given remote host.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Examples

1. To add credentials for `user1` to mount on `server1`, enter:

```
mkcifscred -h server1 -u user1
```

Location

`/usr/sbin/mkcifscred`

Files

Item	Description
`/etc/cifs_fs/cifscred`	Stores the CIFS credentials.

mkcifsmnt Command

Purpose

Adds a CIFS mount to the /etc/filesystems file and performs the mount.

Syntax

mkcifsmnt -f *MountPoint* -d *RemoteShare* -h *RemoteHost* -c *user* [-p *password*] [-m *MountTypeName*] [-A|-a] [-I|-B|-N] [-t {rw|ro}] [-u *uid*] [-g *gid*] [-x *fmode*] [-w *wrkgrp*]

Description

The mkcifsmnt command constructs a CIFS entry that is appended to the /etc/filesystems file. It then attempts to mount the CIFS file system. Its options are parsed and prepared to be passed into the crfs command, which actually adds the CIFS entry to /etc/filesystems.

Flags

Item	Description
-a	Specifies that the /etc/filesystems entry for this file system should not be automatically mounted at system restart. This is the default.
-A	Specifies that the /etc/filesystems entry for this file system should be automatically mounted at system restart.
-B	Specifies that the entry should be added to the /etc/filesystems and that it should be mounted at system restart.
-c *user*	Specifies user name used to gain access to the CIFS share.
-d *RemoteShare*	Specifies the share name on the CIFS server that should be mounted.
-f *MountPoint*	Specifies the path name over which the CIFS share should be mounted.
-g *gid*	Specifies the GID that is assigned to files in the mount. The default is 0.
-h *RemoteHost*	Specifies the name of the remote host (CIFS server). This can be provided as a host name, an IP address, or as a fully qualified domain name.
-I	Specifies that the entry should be added to the /etc/filesystems file, but should not be mounted.
-m *MountTypeName*	Defines the mount type that will be added to the /etc/filesystems file, which allows for mounting all file systems of a specific type using the -t option of the mount command. By default, no type value will be added to /etc/filesystems.
-N	Mounts the CIFS share with the options specified, but does not modify the /etc/filesystems file.

Item	Description
-p *password*	Specifies the password used to grant access to the specific user on the specific server. The specific credentials (server/user/password) are added to the `cifscred` file (the password will be encrypted). If the `-p` option is not specified, and the credentials do not already exist in the `cifscred` file, the command line prompts the user to provide the password, and the credentials will be added to the `cifscred` file. If the server/user credentials already exist in the `cifscred` file, this option is ignored, and the existing credentials are used for mounting.
`-t` {rw|ro}	Specifies whether file system should be mounted as read-only. The default is read-write (rw).
`-u` *uid*	Specifies the UID that is assigned to files in the mount. The default is 0.
`-x` *fmode*	Specifies the owner, group, and other permission bits assigned to files in the mount. The default is 755.
`-w` *wrkgrp*	Specifies the domain that should be used to authenticate the user during mount. If this option is not used, authentication is handled locally by the CIFS server.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

You must have root authority to run this command.

Examples

1. To add a mount over `/mnt` to `share1` on `server1`, and then authenticate as `user1`, enter:

```
mkcifsmnt -f /mnt -d share1 -h server1 -c user1
```

Location

`/usr/sbin/mkcifsmnt`

Files

Item	Description
`/etc/cifs_fs/cifscred`	Stores the CIFS credentials.
`/etc/filesystems`	Stores the CIFS entry.

mkcimreg Command

Purpose

Registers Common Information Model (CIM) classes and Common Manageability Programming Interface (CMPI) providers with RMC.

Syntax

To register a class:

`mkcimreg` [`-I` *include_directory*...] [`-f`] [`-h`] *definition_file*...

To register a provider:

`mkcimreg` [`-I` *include_directory*...] [`-p` *provider_directory*] [`-h`] *registration_file*...

To compile the CIM schema:

`mkcimreg` [`-I` *include_directory*...] `-b` *schema_path* [`-h`]

Description

The `mkcimreg` command registers Common Information Model (CIM) classes and Common Manageability Programming Interface (CMPI) providers with the resource monitoring and control (RMC) subsystem. You can specify one or more class definition files or provider registration files with this command. Use the `-I` flag to add directories to the search path. The output from `mkcimreg` includes the names of the files that the CIM resource manager needs for working with CIM classes.

Registering classes

If you upgrade a class using the `-f` flag (that is, if the class definition has changed somehow), you must re-register all classes that are subclasses of the upgraded class so that the changes introduced into the new class propagate to its subclasses. This must be done in "descending" order, because changes propagate from parent to child. The hierarchy is:

```
------------------
Hardware_Component
------------------
        ↑
        |
------------------
Hardware_Adapter
------------------
        ↑
        |
------------------
Hardware_Ethernet
------------------
```

If, for example, `Hardware_Component` is upgraded using `mkcimreg -f`, `Hardware_Adapter` and then `Hardware_Ethernet` must both be registered afterward, in that order.

After you register any classes:

You must restart RMC.

Restarting RMC

As the final step in the CIM class registration process, the RMC subsystem must be restarted. The sequence of commands to run follows:

1. To shut down the RMC subsystem, enter:

```
/opt/rsct/bin/rmcctrl -k
```

When you shut down RMC:

Any RMC-dependent resource monitoring that is in place at the time of shutdown is deactivated. Environments that rely on RMC or any of its resource managers for high availability or other critical system functions may become temporarily disabled.

2. Wait until the following command lists the status of `ctrmc` as "inoperative":

```
lssrc -s ctrmc
```

3. Shut down the CIM resource manager and confirm it has been stopped:

```
stopsrc -s IBM.CIMRM
lssrc -s IBM.CIMRM
```

4. To restart the RMC subsystem, enter:

```
/opt/rsct/bin/rmcctrl -A
```

Registering providers

The `-p` flag indicates that the registration file on the command line contains provider registration information. The provider library's directory is expected as this flag's parameter. Provider library names follow the CMPI/Pegasus convention of appending `lib` to the beginning of the `ProviderName` property. For example, the provider with the property `ProviderName=Linux_Processor` is searched for in the `ProviderDirectory` under the name `libLinux_Processor.so`. Auxiliary libraries required by providers that are not explicitly declared in the registration file must be either in the directory supplied on the command line, or in a standard system directory such as `/usr/lib` or `/lib`.

Compiling a schema

Version 2.9 of the CIM schema is shipped with the CIM resource manager. Use the `-b` flag if you want to upgrade to a higher version. The schema file (`CIM_Schema`*version*`.mof`) must be passed as the parameter to this flag. This file contains the entire CIM schema, usually in the form of a series of `#include` statements that bring in other schema MOF files.

After a CIM schema is compiled with the `-b` flag, `mkcimreg` will not need further access to the schema managed object format (MOF) files. User classes that are registered by `mkcimreg` against previous versions of the CIM schema need to be re-registered, so changes from the new version of the schema are reflected in any derived classes.

Flags

-I ***include_directory...***
Specifies one or more additional directories to be searched.

-f
Overwrites any existing class registration data with the definitions that are provided in the class definition files.

-p ***provider_directory***
Specifies a path to the provider library.

-b ***schema_path***
Compiles the CIM schema file.

-h
Writes the command's usage statement to standard output.

Parameters

definition_file...
Specifies one or more class definition files.

registration_file...
Specifies one or more provider registration files.

Security

This command requires `root` authority.

Exit Status

0
The command has run successfully.

1
An internal command error occurred.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
A class registration error occurred.

Restrictions

You cannot register a class that derives from a class that has not yet been registered.

Implementation Specifics

This command is part of the `rsct.exp.cimrm` fileset, in the `rsct.exp` package on the AIX Expansion Pack.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output.

Standard Error

When the `-T` flag is specified, this command's trace messages are written to standard error.

Examples

1. To register the `Linux_ComputerSystem` CIM class if the class definition file is located in the `$CIMDEFS` directory, enter:

   ```
   mkcimreg $CIMDEFS/Linux_ComputerSystem.mof
   ```

   You must also register the CMPI provider for this class.
2. To register a CMPI provider when the registration file is located in the `$CIMDEFS` directory and the provider library is in the `$CMPIHOME` directory, enter:

   ```
   mkcimreg -p $CMPIHOME $CIMDEFS/Linux_ComputerSystemRegistration.mof
   ```
3. To compile Version 2.12 of the CIM schema, enter:

   ```
   mkcimreg -I $SCHEMA_DIR -b CIM_Schema2.12.mof
   ```

$SCHEMA_DIR, which indicates a search path for schema MOF files, is not required, but could help mkcimreg find the required MOF files if they are not in the current working directory from which the command is run.

Location

/opt/rsct/bin/mkcimreg

mkclass Command

Purpose

Create a Workload Management class.

Syntax

mkclass [**-a** *Attribute*=*Value ...*] [**-c** | **-m** | **-b** | -v | **-C** | **-B** | **-P** | **-T** | -V | **-L** | **-A** *KeyWord*=*Value*] [**-d** *Config_Dir*] [**-S** *SuperClass*] *Name*

Description

The **mkclass** command creates a superclass or a subclass identified by the *Name* parameter. The class must not already exist. The *Name* parameter can contain only uppercase and lowercase letters, numbers, and underscores. The name is in the format *supername* or *subname* (with the **-S** *supername* flag) or *supername***.***subname*. The *supername* and *subname* parameters are each limited to 16 characters in length. The names **Default**, **System**, and **Shared** are reserved. They refer to predefined classes. Any *Attribute*=*Value* or *KeyWord*=**Value** argument initializes the specified attribute or resource limit. See "Attributes" on page 2388 for more information. To set the process total limits (the limits that apply to each process of the class), use one or more of the options **-C** (totalCPU), **-B** (totalDiskIO), **-A** (totalConnectTime), or -v (totalVirtualMemoryLimit), with the keyword value of `hardmax`. To set the class total limits (the limits that apply to the whole class), use one or more of the options **-P** (totalProcesses), **-T** (totalThreads), **-L** (totalLogins), or -V (totalVirtualMemoryLimit) with the keyword value of `hardmax`. To reset any total limit, use - for *Value*. Process, class, or both total limits may be disabled when starting or updating the WLM (see **wlmcntrl** command).

Normally, **mkclass** adds the class and its attributes in the relevant WLM property files, and the modifications is applied to the in-core class definitions (active classes) only after an update of WLM using the **wlmcntrl** command.

If an empty string is passed as the configuration name (*Config_dir*) with the **-d** flag, the class is created only in the WLM in-core data structures, and no property file is updated, making the new class temporary (the change is lost if WLM is stopped and restarted or the system is rebooted).

Note: This command cannot apply to a set of time-based configurations (do not specify a set with the **-d** flag). If the current configuration is a set, the **-d** flag must be given to indicate which regular configuration the command should apply to.

Attributes

The following attributes can be changed:

Class properties:

Item	Description
tier	Specifies the **tier** value. The **tier** value for a class is the position of the class in the hierarchy of resource limitation desirability for all classes. A class with a lower **tier** value is more favored. The **tier** value ranges from 0 through 9 (the default is 0).

Item	Description
inheritance	If the **inheritance** attribute is set to **yes**, the children of processes in this class remain in the class upon **exec** regardless of the automatic assignment rules in effect. If the **inheritance** attribute is set to **no**, the assignment rules apply normally. The default if not specified is **no**.
localshm	Indicates whether memory segments that are accessed by processes in different classes remain local to the class they were initially assigned to or if they go to the **Shared** class. You can specify a value of **Yes** or **No**. If not specified, the default is **No**.
authuser	Specifies the user name of the user who is allowed to assign processes to this class. The default when the attribute is not specified is **root**.
authgroup	Specifies the group name of the group of users that is allowed to assign processes to this class. There is no default value.
rset	Specifies the name of a resource set that the processes in the class have access to. By default, the class has access to all resources on the system.
vmenforce	Specifies whether all processes or only the offending processes in the class need to be terminated when the class hits the maximum VM limit. You can specify the value of `class` or `proc`. The default value is `proc`.
delshm	Specifies whether the shared segments will be deleted when the last process referencing them ends because virtual memory is exceeded. You can specify the value of `yes` or `no`. The default value is `no`.
adminuser	Specifies the user name of the user who is allowed to administer the subclasses of this superclass. This attribute is valid only for superclasses. The default, when the attribute is not specified, is a null string, and in this case, only **root** users can administer the subclasses. **Note:** If the **adminuser** or **admingroup** attribute is changed for a superclass that belongs to the running configuration (or to a configuration of the running set), a global WLM update should be performed to reflect these changes to the in-core configuration, elsewhere, updates that are restricted to superclass by such a user might fail due to lack of authority.
admingroup	Specifies the group name of the group of users that is allowed to administer the subclasses of this superclass. This attribute is valid only for superclasses. The default value, when the attribute is not specified, is a null string, meaning that no group can administer the subclasses. **Note:** If the **adminuser** or **admingroup** attribute is changed for a superclass that belongs to the running configuration (or to a configuration of the running set), a global WLM update should be performed to reflect these changes to the in-core configuration, elsewhere, updates that are restricted to superclass by such a user might fail due to lack of authority.

Class limits and shares for CPU, memory, or disk I/O resource:

Item	Description
min	Specifies the minimum percentage of the resource that must be made available when requested, expressed as a percentage of the total resource available in the system. Possible values range from 0 through 100 (the default is 0).

Item	Description
shares	Specifies the maximum ratio of the resource that can be made available if there is contention. This parameter is expressed in shares of the total resource available in the system. The actual ratio of the resource is dynamically computed, proportionally to the shares of all active classes. If a class has no running process, its shares are excluded from the computation. The shares are arbitrary numbers ranging from 1 through 65535. If shares is specified as a hyphen (-), the class is always considered on target and its utilization for this resource is not regulated by WLM, but the minimum and maximum limits if any still apply. This is the default if the shares for a resource are not specified.
softmax	Specifies the maximum percentage of the resource that can be made available, when there is contention. Possible values range from 1 through 100 (the default is 100). A class can exceed its soft maximum for a given resource if there is no contention on the resource.
hardmax	Specifies the maximum percentage of the resource that can be made available, even if there is no contention. Possible values range from 1 through 100 (the default is 100). Specifying a value different from the default value of 100 for memory can result in some memory pages remaining unused, while some processes in the class use more.
max	Specifies the maximum percentage of the resource that can be made available, even if there is no contention. Possible values range from 1 through 100 (the default is 100). Specifying a value different from the default value of 100 for memory can result in some memory pages remaining unused, while some processes in the class use more.

Note: The default values for a class can be read using the **lsclass -D** command and can be changed by manually editing the property files **classes**, **shares**, or **limits** to add a default stanza. For more information about these files, see the *Files Reference*.

Class description:

Item	Description
description	The class description text can be composed of any ASCII character, except colons (**:**) and commas (**,**).

Note: This command is not supported when executed within a workload partition.

Flags

Item	Description
-A hardmax=***Value***	Sets the maximum amount of time a login session in the class can stay active. Value is specified as an integer, possibly appending the unit (s for seconds, m for minutes, h for hours, d for days, and w for weeks, default is seconds). As a user approaches this connection time limit, WLM will send a warning message to the session terminal. When the limit is reached, the user will be notified and the session leader will be sent the **SIGTERM** signal, and after a short grace period, the session will be terminated (**SIGKILL**).
-B hardmax=*Value*	Sets the total amount of disk I/Os allowed for each process in the class. Value is specified as an integer, possibly appending the unit (KB for kilobytes, MB for megabytes, TB for terabytes, PB for petabytes, and EB for exabytes, default is kilobytes). After a process has used this amount of disk I/Os, the process will be sent the **SIGTERM** signal, and after a grace period, it will be killed (**SIGKILL**).
-C hardmax=*Value*	Sets the total amount of CPU time allowed for each process in the class. Value is specified as an integer, possibly appending the unit (s for seconds, m for minutes, h for hours, d for days, and w for weeks, default is seconds). After a process has used this amount of time, the process will be sent the **SIGTERM** signal, and after a grace period, it will be killed (**SIGKILL**).

Item	Description
-d *Config_Dir*	Use **/etc/wlm/***Config_Dir* as an alternate directory for the properties files. When this flag is not used, **mkclass** uses the configuration files in the directory pointed to by **/etc/wlm/current**. If an empty string is passed as the configuration name (`-d ""`) the new class is created only in the WLM in-core data structures and no configuration file is modified.
-L hardmax=*Value*	Sets the total number of login sessions simultaneously available in the class. If a user tries to log onto the system and the login shell would end up in a class that has reached the total logins limit, the login operation will fail.
-P hardmax=*Value*	Sets the maximum number of processes allowed in the class. If an operation would result in a new process entering the class when the class has this many processes in it, the operation will fail.
-S *SuperClass*	Specifies the name of the superclass when creating a subclass. There are two ways of creating the subclass **Sub** of superclass **Super**: 1. Specify the full name of the subclass as **Super.Sub** for *Name* and not use **-S** 2. Specify the **-S** flag to give the superclass name and use the short name for the subclass: `mkclass options -S Super  Sub`
-T hardmax=*Value*	Sets the maximum number of threads allowed in the class. If an operation would result in a new thread entering the class when the class has this many processes in it, the operation will fail. The total thread limit must be at least as large as the total process limit for a class. If a class has a total thread limit but no total process limit specified, the total process limit will be set to the total thread limit.
`-v hardmax=`*Value*	Specifies the virtual memory limit allowed per process in the specified class. The maximum amount of virtual memory allowed per process is (2^31)-1 for 32-bit kernels and (2^63)-1 for 64-bit kernels.
`-V hardmax=`*Value*	Specifies the virtual memory allowed for the specified class. The maximum amount of virtual memory allowed per process is (2^31)-1 for 32-bit kernels and (2^63)-1 for 64-bit kernels.

Security

Access control: Only the root user can create a superclass. Only root or authorized users whose user ID or group ID matches the user name or group name specified in the attributes **adminuser** and **admingroup** of a superclass can create a subclass of this superclass.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
classes	Contains the names and definitions of the classes.
limits	Contains the resource limits enforced each class.
shares	Contains the resource shares attributed to each class.

mkclient Command

Purpose

Uncomments the entry in the **/etc/rc.nfs** file for the **ypbind** daemon and starts the **ypbind** daemon to configure a client.

Syntax

/usr/sbin/mkclient [**-I** | **-B** | **-N**] [**-S** *server*]

Description

The **mkclient** command uncomments the entry to the **/etc/rc.nfs** file to start the **ypbind** daemon to configure a client. The **mkclient** command starts the **ypbind** daemon by using the appropriate System Resource Controller (SRC) command.

You can use the System Management Interface Tool (SMIT) **smit mkclient** fast path to run this command.

Flags

Item	Description
-I	Uncomments the entry for starting the **ypbind** daemon to the **/etc/rc.nfs** file. This entry causes the **ypbind** daemon to start during the next system restart.
-B	Uncomments the entry to the **/etc/rc.nfs** file and starts the **ypbind** daemon. This flag is the default.
-N	Causes the **startsrc** command to start the **ypbind** daemon. This flag does not affect the **/etc/rc.nfs** file.
-S	Specifies which NIS *server* to use instead of broadcasting for one. This option must be used when no NIS server exists on the networks directly connected to the client machine.

Examples

To modify the **/etc/rc.nfs** file so that the **ypbind** daemon is started on the next system restart, enter:

```
mkclient -I
```

Files

Item	Description
/var/yp/domainname directory	Contains the NIS maps for the NIS domain.
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.

mkcluster Command

Purpose

To create a single-site cluster.

Syntax

mkcluster **-r** *reposdev* [**-S** *sitename* { [`cle_uuid=`*UUID*,`cle_globid=`*id*,`cle_prio=`*prio*] }] [**-m** *node* { [`cle_ip=`*addr*,`cle_uuid=`*UUID*,`cle_globid=`*id*] }] [,...] [**-d** *shareddisk* [,...]] [**-n** *clustername*] [**-s** *multi_cast_addr*] [**-c** *capability* [,...]] [-v] [**-b** backupdisk [,...]] [**-p** *comdisk*]

Description

The **mkcluster** command creates a cluster. A cluster is a collection of nodes and disks.

Each node that is added to the cluster must have common storage area network (SAN) storage devices that are zoned appropriately. The SAN storage devices are used for the cluster repository disk and for any clustered shared disks.

A multicast address is used for cluster communications between the nodes in the cluster. If any network considerations need to be reviewed before creating a cluster, consult your cluster systems administrator.

Flags

Item	Description
-b *backupdisk* [,...]	Specifies a comma-separated list of SAN shared storage device such as hdisk5 and hdisk6. These disks are used as the backup for the central repository of the cluster. When the central repository is inaccessible, the disk from the list is used as a replacement. These devices must be accessible from all nodes in the site.
-c *capability* [,...]	Specifies a comma-separated list of capabilities that the cluster requires upon creation. If no capabilities are specified, the **mkcluster** command allows for the possibility that some nodes have older AIX software that is not capable of supporting newer CAA capabilities. In that case, the cluster is created in such a way that it is compatible with nodes that run older AIX software. After the cluster is created, and it is determined that all nodes can support newer CAA capabilities, the cluster automatically allows those capabilities to be used. The **-c** flag merely allows the specified capabilities to be used without first determining that all nodes are able to support it. In some situations, it is necessary for newer CAA capabilities to be enabled immediately. For example, if one or more nodes do not have IPv4 connectivity to all of the other cluster nodes, then it is necessary to specify that IPv6 capability must be enabled during cluster creation, to allow those nodes that have only IPv6 connectivity to join the cluster. All nodes must be online and able to join the cluster, to determine that the cluster can support newer capabilities. If a cluster must be created while one or more of the nodes are powered off, then specifying that the capability is needed during cluster creation allows the capability to be used before all of the nodes are able to join the cluster. However, you must make sure that all of the nodes have an AIX software level that can support that capability. Otherwise, any nodes that have older AIX software that cannot support the capability are not allowed to join the cluster. The **-c** flag supports the following capability keywords: • ipv6: IPv6 connectivity is required because some nodes do not have IPv4 connectivity to the rest of the cluster, or IPv6 functionality is needed before all nodes are able to join the cluster. • site: The cluster needs to allow one or more sites to be defined before all nodes are able to join the cluster. • auto_repos_replace: The created cluster can process and maintain the backup repository disks. • 4kdisk: The cluster must be able to support a 4k-block disk as repository and backup repository disk.

Item	Description
-c *unicast* or *multicast*	Specifies the type of communication mode that is used by CAA to transfer CAA heartbeats and other protocol messages. If the *unicast* option is specified, CAA uses unicasting to transfer the protocol messages. If the *multicast* option is specified, CAA uses multicasting to transfer the protocol messages. If no option is specified, CAA uses the default multicast communication mode.
-p *comdisk*	Specifies a SAN shared storage device such as `hdisk5` and `hdisk6`. These disks are used by the shared storage pool cluster for inter-node communication when the network is down.
-r *reposdev*	Specifies the name of the SAN shared storage device that is used as the central repository for the cluster configuration data, such as *hdisk10*. This device must be accessible from all gateway nodes in the site. It is required that this device is a minimum of 1 GB, and is backed up by a redundant and highly available SAN configuration.
-S *sitename*	Specifies the name of the local site. If not specified, a default site with the name LOCAL is created. Currently, a cluster can support only 2 sites. To create a second site, use the **chcluster** command. The following site information can be specified: • `cle_uuid`: The site UUID, which is acknowledged as unique across the cluster. If not specified, the site UUID is automatically generated. • `cle_globid`: The short ID of site, which must be a unique unsigned number greater than zero. If not specified, the site short ID is automatically generated. The following site attribute can be specified: • `cle_prio`: The priority of a site. A lower value indicates a higher priority. The priority is used in the context of synchronizing the repository metadata. If two sites split and the repository data becomes out of sync, then the data from the site with higher priority must be copied over to the site with lower priority.
-m *node*[,...]	Lists the comma-separated host names or IP addresses for nodes that are members of the cluster. The local host must be included in the list. If the **-m** flag is not used, the local host is implied, causing a one-node local cluster to be created. The following node information can be specified: • `cle_uuid`: The node UUID, which is acknowledged as unique across the cluster. If not specified, the node UUID is automatically generated. • `cle_globid`: The short ID of node, which must be a unique unsigned number greater than zero. If not specified, the node short ID is automatically generated. The following node attributes can be specified: • `cle_ip`: The nodes gateway address (in case the cluster spans across multiple sites). Typically, this attribute is an address through which this node can be reached from an external node. This address can be specified in either IP version 4 or version 6 format.

Item	Description
-d *shareddisk*[,...]	Specifies a comma-separated list of shared storage area network (SAN) devices, such as *hdisk12, hdisk34,* to be incorporated into the cluster configuration. Specified devices must not be open when the **mkcluster** command is run.
-n *clustername*	Sets the name of the cluster that is being created. If no name is specified when you run the **mkcluster** command, a default of **CL_hostname** is used, where *hostname* is the name of the local host. You can retrieve the name of the local host by running the **gethostname()** function.
-s *multi_cast_addr*	Sets the multicast address of the cluster that is created. This address is used for internal communication within the cluster. Only a multicast address in IPv4 format is accepted. If an IPv6 multicast address is needed, it must be generated from the IPv4 address. If the **-s** flag is not specified when you first run the **mkcluster** command, the necessary multicast addresses are automatically generated.
-v	Specifies the verbose mode.

Examples

1. To create a cluster of one node and use the default values, enter the following command:

   ```
   mkcluster -r hdisk
   ```

   The output is a cluster that is name CL_*myhostname* with a single node in the cluster. The multicast address is automatically generated and no shared disks are created for this cluster. The

   ```
   mkcluster -r hdisk1
   ```

   repository is set up on *hdisk1* and this disk cannot be used by the node for any other purpose. The repository device is dedicated as the cluster repository disk.
2. To create a multinode cluster, enter the following command:

   ```
   mkcluster -n mycluster -m nodeA,nodeB,nodeC
   ```

   The output is a cluster of three nodes and uses the default values. The output also creates a cluster with the specified name and the multicast address is automatically created. Three disks are created as shared clustered disks for this cluster. The repository device is set up on *hdisk1*, and it cannot be used by any of the nodes for any other purpose. The repository device is now dedicated to being the cluster repository disk. A volume group of cvg is created for the cluster repository disk and these logical volumes are used exclusively by the clustering subsystem.
3. To create a cluster that is capable of IPv6 and sites, enter the following command:

   ```
   mkcluster -n mycluster -m nodeA,nodeB,nodeC -r hdisk1 -c ipv6, site
   ```

   This command creates a cluster of three nodes that are immediately capable of using IPv6 networks and having sites that are defined. The cluster is named *mycluster*, and the local site and multicast address are automatically created with default values. If any IPv6 networks are configured, they can be used for cluster communication. The repository device is set up on hdisk1, and it cannot be used by any of the nodes for any other purpose. The repository device is now dedicated to being a cluster repository disk.
4. To create a cluster with one site named *mysite*, enter the following command:

   ```
   mkcluster -n mycluster -S mysite -m nodeA,nodeB,nodeC -r hdisk1 -d hdisk10,hdisk11,hdisk12
   ```

   The output is a single-site cluster of three nodes that uses the default attribute values for all sites and all nodes. The cluster is named *mycluster*, and the local site is named *mysite*. The multicast address is automatically created. Three disks are created as shared clustered disks for the local site. The

repository device is set up on *hdisk1*, and it cannot be used by any of the nodes for any other purpose. The repository device is now dedicated to being a cluster repository disk.

5. To create a cluster with one site named *mysite*, specifying site and node information, enter the following command:

```
mkcluster -n mycluster -S mysite{cle_uuid=0551c722-92fe-11e1-97b0-1aae1ed14715,
cle_globid=5,cle_prio=2}
-m nodeA,nodeB,nodeC{cle_uuid=e4ad47bc-92fd-11e1-8486-1aae1ed14715}
-r hdisk1 -d hdisk10,hdisk11,hdisk12
```

   The output is a single-site cluster of three nodes. Nodes nodeA and nodeB have automatically generated UUIDs, while nodeC has a UUID of e4ad47bc-92fd-11e1-8486-1aae1ed14715. The cluster is named *mycluster*, and the local site is named *mysite* and has a UUID of 0551c722-92fe-11e1-97b0-1aae1ed14715, a short ID of 5, and a priority of 2. The multicast address is automatically created. Three disks are created as shared clustered disks for the local site. The repository device is set up on *hdisk1*, and it cannot be used by any of the nodes for any other purpose. The repository device is now dedicated to being a cluster repository disk.

6. To create a multinode unicast cluster with one site named *mycluster*, append **-c** *unicast* to the **mkcluster** command. The multicast cluster in example 2 can be made unicast by entering the following command:

```
mkcluster -r hdisk10 -m nodeA,nodeB,nodeC -n mycluster -r hdisk1
-d hdisk10,hdisk11,hdisk12 -c unicast
```

7. To create a cluster that uses backup disks that are added later, enter the following command:

```
mkcluster -n mycluster -n nodeA,nodeB -r hdisk1 -c auto_repos_replace
```

8. To create a cluster and populate the backup repository disk list, enter the following command:

```
mkcluster -n mycluster -n nodeA,nodeB -r hdisk1 -b hdisk5,hdisk6
```

Messages

CLUST_LVL_NO_4K
"1035-346 %1$s: The current effective cluster level does not support 4k-block disks.\n"

DISK_INFO_ERR
"1035-347 %1$s: Unable to get disk information for %2$s.\n"

mkcomg Command

Purpose

Creates a new communication group definition for a peer domain.

Syntax

mkcomg [-s *sensitivity*] [-p *period*] [**-g** *grace*] [-t *priority*] [-x b | r | br] [**-N** *UseForNodeMembership*] [-e *NIM_path*] [-m *NIM_parameters*] [**-M** *media_type*] [**-i** {**h** | **n**}**:***interface1*[**:***node1*][**,***interface2*[**:***node2*]...] | **-S** {**h** | **n**}:**"***interface_selection_string***"** [**-6**] [-h] [-TV] *communication_group*

Description

The mkcomg command creates a new communication group definition for an online peer domain with the name specified by the *communication_group* parameter. The communication group is used to define heartbeat rings for use by topology services and to define the tunables for each heartbeat ring. The communication group determines which devices are used for heartbeating in the peer domain. There can be more than one communication group in a peer domain.

The `mkcomg` command must be run on a node that is currently online in the peer domain where the communication group is to be defined. More than half of the nodes must be online to create a new communication group for the domain.

The `-e` and `-m` flags are used to set the network interface module (NIM) path and parameters. The NIM path is the path to the NIM that supports the adapter types used in the communication group. The NIM parameters are passed to NIM when it is started. If `-m` is not specified, the parameters predefined by topology services are used.

The communication group can be assigned to one or more interface resources. Use the `-i` flag to assign the communication group to a specific interface resource name. The interface resource can be limited to one on a particular node. An interface resource can also be specified using the `-S` flag and a selection string. This is used when specifying the interface resource name is not sufficient. The `-i` and `-S` flags cannot be used together. The `chcomg` command can also be used to assign a communication group to an interface resource.

Flags

-s *sensitivity*
Specifies the heartbeat sensitivity. This is the number of missed heartbeats that constitute a failure. The sensitivity value is an integer greater than or equal to 2. The default value is 4.

-p *period*
Specifies the amount of time between heartbeats. The period is specified in seconds and is significant to milliseconds. It can be specified as an integer or as a floating-point number.

-g *grace*
Specifies the grace period that is used when heartbeats are no longer received. When a heartbeat is missed, an Internet Control Message Protocol (ICMP) echo packet is sent to the failed node. If the echo is returned, the grace period is initiated.

The grace period is specified in seconds and is significant to milliseconds. It can be specified as an integer, a floating-point number, or one of these values:

0
Specifies that the grace period is disabled.

-1 | D
Specifies that the topology services subsystem controls the grace period. This is the default.

-t *priority*
Specifies the priority. This value indicates the importance of this communication group with respect to others. It is used to order the heartbeat rings. The lower the number means the higher the priority. The highest priority is 1. The default value is 1 for IP networks and 255 for RS232 networks.

-x b | r | br
Excludes controls for heartbeat mechanisms. This flag indicates that one or more controls for heartbeat mechanisms should not be used even if the underlying media support it. The following features can be excluded:

b
Specifies that the broadcast feature should not be used even if the underlying media support it. If `-x b` is not specified, the broadcast feature will be used if the underlying media support it.

r
Specifies that the source routing feature should not be used even if the underlying media support it. If `-x r` is not specified, the source routing feature will be used if the underlying media support it.

To exclude more than one control, specify the feature characters consecutively: `-x br`.

-N *UseForNodeMembership*
Specifies whether group services will use the communication group in calculating node membership. Sets the **UseForNodeMembership** persistent resource attribute for the communication group resource. Valid values are:

0
Indicates that, regardless of the results of liveness checks run on **NetworkInterface** resources that are members of this communication group, group services will not use those results in calculating whether the node owning the interfaces is online.

1
Indicates that group services will use the results of liveness checks run on the **NetworkInterface** resources in calculating the online state of their owning nodes.

-e *NIM_path*
Specifies the network interface module (NIM) path name. This character string specifies the path name to the NIM that supports the adapter types in the communication group.

-m *NIM_parameters*
Specifies the NIM start parameters. This character string is passed to the NIM when starting it.

-M *media_type*
Specifies the type of interfaces that make up *communication_group*. Valid values are:

0
Indicates that *communication_group* consists of interface resources other than IP or disk.

1
Indicates that *communication_group* consists of IPv4 or IPv6 interface resources.

If the **-M** flag is not specified, this is the default.

2
Indicates that *communication_group* consists of disk interface resources.

-i {h | n}:*interface1*[:*node1*] [,*interface2*[:*node2*]...
Assigns *communication_group* to one or more heartbeat or network interface resources and, optionally, to the nodes where these resources can be found. Specify **-i h** for heartbeat interface resources or **-i n** for network interface resources.

By default, the **-i n** flag adds network interface resources that have IPv4 addresses to *communication_group*. If the **-6** flag is specified, the **-i n** flag will add network interface resources that have IPv6 addresses to *communication_group*.

If `-i` is specified, `-S` cannot be specified.

-S {h | n}:"*network_selection_string*"
Assigns *communication_group* to the heartbeat or network interface that is specified by *interface_selection_string*. Specify **-S h** for heartbeat interfaces or **-S n** for network interfaces.

By default, the **-S n** flag adds network interface resources that have IPv4 addresses to *communication_group*. If the **-6** flag is specified, the **-S n** flag will add network interface resources that have IPv6 addresses to *communication_group*.

If `-S` is specified, `-i` cannot be specified.

-6
Specifies that IPv6 addresses represented as resources on each interface have their communication group changed to the one specified. IPv4 addresses represented as resources on the interfaces would be unaffected.

By default (without **-6** specified), the inverse is true. Only IPv4 addresses represented as resources on the interface would have their communication group changed.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

communication_group
Specifies the name of the new communication group that is to be created for the online peer domain. The name can contain any printable character.

Security

The user of the `mkcomg` command needs write permission for the `IBM.CommunicationGroup` resource class. Write permission for the `IBM.NetworkInterface` resource class is required to set the communication group for a network interface resource. By default, `root` on any node in the peer domain has read and write access to these resource classes through the configuration resource manager.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is defined and online to the peer domain where the communication group is to be defined.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the `-f "-"` or `-F "-"` flag is specified, this command reads one or more node names from standard input.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To define the communication group ComGrp1 for the peer domain ApplDomain and nodeA is defined and online to ApplDomain, run this command on nodeA:

```
mkcomg ComGrp1
```

2. To define the communication group ComGrp1 for the peer domain ApplDomain, using a sensitivity of 1 and period of 3, and nodeA is defined and online to ApplDomain, run this command on nodeA:

```
mkcomg -s 1 -p 3 ComGrp1
```

3. To define the communication group ComGrp1 for the peer domain ApplDomain, not using broadcast, using a priority of 3, and nodeA is defined and online to ApplDomain, run this command on nodeA:

```
mkcomg -x b -t 3 ComGrp1
```

4. To define the communication group ComGrp1 for the peer domain ApplDomain, not using broadcast, not using source routing, and nodeA is defined and online to ApplDomain, run the following command on nodeA:

```
mkcomg -x br ComGrp1
```

5. To define the communication group ComGrp1 for the peer domain ApplDomain, using a NIM path of /opt/rsct/bin/hats_nim, NIM parameters -l 5 to set the logging level, and nodeA is defined and online to ApplDomain, run this command on nodeA:

```
mkcomg -e /opt/rsct/bin/hats_nim -m "-l 5" ComGrp1
```

6. To define the communication group **ComGrp1** for **ApplDomain** and assign **ComGrp1** to the heartbeat interface resource named **hbi0** on **nodeC**, run this command on **nodeA**:

```
mkcomg -i h:hbi0:nodeC ComGrp1
```

7. To define the communication group ComGrp1 for the peer domain ApplDomain, assign ComGrp1 to the network interface resource named eth0 on nodeB, and nodeA is defined and online to ApplDomain, run this command on nodeA:

```
mkcomg -i n:eth0:nodeB ComGrp1
```

8. To define the communication group **ComGrp1** for **ApplDomain** and assign **ComGrp1** to the heartbeat interface resource that uses the subnet 9.345.67.812, run this command on **nodeA**:

```
mkcomg -S h:"Subnet == 9.345.67.812" ComGrp1
```

9. To define the communication group ComGrp1 for the peer domain ApplDomain, assign ComGrp1 to the network interface resource that uses the subnet 9.123.45.678, and nodeA is defined and online to ApplDomain, run this command on nodeA:

```
mkcomg -S n:"Subnet == 9.123.45.678" ComGrp1
```

10. To define the communication group **ComGrp1** for **ApplDomain**, using a period of 500 milliseconds, run this command on **nodeA**:

```
mkcomg -p 0.5 ComGrp1
```

Location

/opt/rsct/bin/mkcomg

mkcondition Command

Purpose

Creates a new condition definition which can be monitored.

Syntax

mkcondition -r *resource_class* **-e "***event_expression***"** [**-E "***rearm_expression***"**] [**-d "***event_description***"**] [**-D "***rearm_description***"**] [**-b** *interval*[**,***max_events*][**,***retention_period*][**,***max_totalsize*]] [**-m l** | **m** | **p**] [**-n** *node_name1*[**,***node_name2*...]] [**-p** *node_name*] [**--qnotoggle** | *--qtoggle*] [**-s "***selection_string***"**] [**-S c** | **w** | **i**] [**-g 0** | **1** | **2**] [**-h**] [**-TV**] *condition*

mkcondition -c *existing_condition*[**:***node_name*] [**-r** *resource_class*] [**-e "***event_expression***"**] [**-E "***rearm_expression***"**] [**-d "***event_description***"**] [**-D "***rearm_description***"**] [**-b** *interval*[**,***max_events*] [**,***retention_period*][**,***max_totalsize*]] [**-m l** | **m** | **p**] [**-n** *node_name1*[**,***node_name2*...]] [**-p** *node_name*] [**--qnotoggle** | *--qtoggle*] [**-s "***selection_string***"**] [**-S c** | **w** | **i**] [**-g 0** | **1** | **2**] [**-h**] [**-TV**] *condition*

Description

The **mkcondition** command creates a new condition with the name specified by the condition parameter. The condition is used to monitor a resource for the occurrence of the condition (or event). Use the **mkresponse** command to define one or more responses to an event. You can then link the conditions to the responses using the **mkcondresp** command, or you can use the **startcondresp** command to link the responses and start monitoring.

Using the **-b** flag, multiple events can be batched or grouped together and passed to a response. The grouping of events is by the time span in which they occur. In addition, the grouping can be done such that a specified maximum number of events are grouped within the time span. A response that handles batched events must be defined as supporting batched events.

In a cluster environment, use the `-p` flag to specify the node in the domain that is to contain the condition definition. If you are using `mkcondition` on the management server and you want the condition to be defined on the management server, do *not* specify the `-p` flag. If the `-p` flag is not specified, the condition is defined on the local node. If the node where the condition will be defined is:

- in a cluster of nodes, the condition can monitor resources on more than one node. Use the `-n` flag to specify the nodes on which the condition will be monitored.
- the management server in a management domain, a management scope (`-m`) of local (`l`) or management domain (`m`) can be specified to indicate how the condition applies. The selection string will be evaluated using the entire management domain when management scope is set to the management domain and the node is the management server.
- a managed node in a management domain, only a management scope (`-m`) of local (`l`) can be used.
- in a peer domain, a management scope (`-m`) of peer domain (`p`) or local (`l`) can be used to indicate how the condition and the selection string apply.
- in both a management domain and a peer domain, a management scope (`-m`) of management domain (`m`), peer domain (`p`), or local (`l`) can be used to indicate how the condition and its selection string apply.

To lock a condition so it cannot be modified or removed, use the `chcondition` command (with its `-L` flag).

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM `nodegrp` command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference.*

Flags

-b *interval*[,*max_events*][,*retention_period*][,*max_totalsize*]
Specifies one or more batching-related attributes. Use commas to separate the attribute values. Do not insert any spaces between the values or the commas.

interval specifies that the events are to be batched together for the indicated interval. Batching continues until no events are generated for an interval. Use an interval of 0 to turn batching off.

max_events specifies that the events are to be batched together until the *max_events* number of events are generated. The interval restarts if the *max_events* number of events is reached before the interval expires.

retention_period specifies the retention period in hours. The batched event file is saved for the time specified as the retention period. Once this time is reached, the file is automatically deleted.

max_totalsize specifies the total size for the batched event file in megabytes (MB). The batched event file is saved until this size is reached, Once the size is reached, the file is automatically deleted.

max_events, *retention_period*, and *max_totalsize* cannot be specified unless interval is greater than 0.

When *interval* is greater than 0 and *max_events* is 0, no maximum number of events is used.

If *retention_period* and *max_totalsize* are both specified, the batched event file is saved until the specified time or size is reached, whichever occurs first.

If you want to change one, two, or three attribute values, you must specify a valid value or an empty field for any attributes that precede the value you want to change. You do not have to specify any values for attributes that follow the value you want to change. For example, if you only need to change the retention period, you need to specify values for interval and *max_events* as well. You can provide an empty field if an attribute does not need to be changed. To change the retention period to 36 hours without changing the values of interval and *max_events*, enter:

```
mkcondition -c existing_condition -b ,,36
```

-c *existing_condition*[:*node_name*]
Copies an existing condition. The existing condition is defined on *node_name*. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable. If any other flags are specified, update the new condition as indicated by the flags. Links with responses are not copied.

-d "*event_description*"
Describes the event expression.

-D "*rearm_description*"
Describes the rearm expression.

-e "*event_expression*"
Specifies an *event expression,* which determines when an event occurs. An event expression consists of a dynamic attribute or a persistent attribute of *resource_class*, a mathematical comparison symbol (or <, for example), and a constant. When this expression evaluates to TRUE, an event is generated.

-E "*rearm_expression*"
Specifies a rearm expression. After *event_expression* has evaluated to True and an event is generated, the rearm expression determines when monitoring for the event expression will begin again. Typically, the rearm expression prevents multiple events from being generated for the same event evaluation. The rearm expression consists of dynamic attributes or persistent attributes of *resource_class*,

mathematical comparison symbols (> or <, for example), logical operators (|| or &&), constants, and an optional qualifier.

--g 0 | 1 | 2
Specifies granularity levels that control audit logging for the condition. The levels of granularity are:

0
Enables audit logging. ERRM writes all activities to the audit log. This is the default.

1
Enables error logging only. ERRM writes only in case of errors to the audit log.

2
Disables audit logging. ERRM does not write any records to the audit log.

-m l | **m** | **p**
Specifies the management scope to which the condition applies. The management scope determines how the condition is registered and how the selection string is evaluated. The scope can be different from the current configuration, but monitoring cannot be started until an appropriate scope is selected. The valid values are:

l
Specifies *local* scope. This is the default. The condition applies only to the local node (the node where the condition is defined; see the `-p` flag). Only the local node is used in evaluating the selection string.

m
Specifies *management domain* scope. The condition applies to the management domain in which the node where the condition is defined belongs (see the `-p` flag). All nodes in the management domain are used in evaluating the selection string. The node where the condition is defined must be the management server in order to use management domain scope.

p
Specifies *peer domain* scope. The condition applies to the peer domain in which the node where the condition is defined belongs (see the `-p` flag). All nodes in the peer domain are used in evaluating the selection string.

-n *node_name1*[,*node_name2*...]
Specifies the host name for a node (or a list of host names separated by commas for multiple nodes) where this condition will be monitored. Node group names can also be specified, which are expanded into a list of node names.

You must specify the **-m** flag with a value of **m** or **p** if you want to use the **-n** flag. This way, you can monitor conditions on specific nodes instead of the entire domain.

The host name does not have to be online in the current configuration, but once the condition is monitored, the condition will be in error if the node does not exist. The condition will remain in error until the node is valid.

-p *node_name*
Specifies the name of the node where the condition is defined. This is used in a cluster environment and the node name is the name by which the node is known in the domain. The default *node_name* is the local node on which the command runs. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

If you are using `mkcondition` on the management server and you want the condition to be defined on the management server, do *not* specify the `-p` flag.

--qnotoggle
Specifies that monitoring does not toggle between the event expression and the rearm expression, but instead the event expression is always evaluated.

--qtoggle
Specifies that monitoring toggles between the event expression and the rearm expression.

-r *resource_class*
Specifies the resource class to be monitored by this condition. You can display the resource class names using the `lsrsrcdef` command.

-s "*selection_string*"
Specifies a selection string that is applied to all of the *resource_class* attributes to determine which resources should be monitored by the *event_expression*. The default is to monitor all resources within the *resource_class*. The resources used to evaluate the selection string is determined by the management scope (the `-m` flag). The selection string must be enclosed within double or single quotation marks. For information on how to specify selection strings, see the *RSCT: Administration Guide* .

-S c | **w** | **i**
Specifies the severity of the event:

c
Critical

w
Warning

i
Informational (the default)

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

condition
The *condition* name is a character string that identifies the condition. If the name contains spaces, it must be enclosed in quotation marks. A name cannot consist of all spaces, be null, or contain embedded double quotation marks.

Security

The user needs write permission for the `IBM.Condition` resource class to run `mkcondition`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To define a condition with the name "FileSystem space used" to check for percentage of space used greater than 90% and to rearm when the percentage is back down below 85%, enter:

```
mkcondition -r IBM.FileSystem  \
-e "PercentTotUsed > 90" -E "PercentTotUsed < 85" \
"FileSystem space used"
```

2. To define a condition with the name "tmp space used" to check for percentage of space used greater than 90% for **/tmp** and to rearm when the percentage is back down below 85%, including comments, enter:

```
mkcondition -r IBM.FileSystem  \
-e "PercentTotUsed > 90" -E "PercentTotUsed < 85" \
-d "Generate event when tmp > 90% full"  \
```

```
-D "Restart monitoring tmp again after back down < 85% full"\
-s 'Name=="/tmp"'  "tmp space used"
```

3. To define a condition with the name "Space used" as a copy of "FileSystem space used", enter:

```
mkcondition -c "FileSystem space used"  "Space used"
```

4. To define a condition with the name "var space used" as a copy of "tmp space used", but change the selection to **/var**, enter:

```
mkcondition -c "tmp space used" -s 'Name=="/var"' \
"var space used"
```

5. To define a condition with the name "vmstat is running" to monitor when user `joe` is running the `vmstat` program in a 64-bit environment, enter:

```
mkcondition -r "IBM.Program" \
-e "Processes.CurPidCount > 0" -E "Processes.CurPidCount  <= 0"  \
-d "Generate event when user starts vmstat" \
-D "Restart monitoring when vmstat is terminated" \
-s ProgramName == \"vmstat64\" && Filter==\"ruser==\\\"joe\\\"\"" \
-S "i" -m "l" "vmstat is running"
```

6. To define a condition with the name "myscript terminated" to monitor when a script has ended, enter:

```
mkcondition -r "IBM.Program" \
-e "Processes.CurPidCount <= 0" -E "Processes.CurPidCount  > 0"  \
-d "Generate event when myscript is down" \
-D "Rearm the event when myscript is running" \
-s ProgramName == \"ksh\" && Filter == 'args[1]==\"/home/joe/myscript\"'" \
-m "l" "myscript terminated"
```

   In this example, `args` represents the array of argument strings that was passed to `main`. Because this is an array, `args[1]` references the first argument after the program name. Use the `ps -el` command to determine the `ProgramName`. See the `lsrsrcdef` command for more information.

7. To batch together a maximum of 20 events at a time that come from a sensor named **DBInit** in 60-second intervals, enter:

```
mkcondition -r "IBM.Sensor" \
-e "Int32 < 0" -E "Int32 > 0"  -b 60,20 \
-s "Name == \"DBInit\""  "DBInit Sensor"
```

8. To define a condition with the name tmp space used to check for percentage of space used greater than 90% for **/tmp** for at least seven out of the last 10 observations, including comments, enter:

```
mkcondition -r IBM.FileSystem \
-e "PercentTotUsed > 90 __QUAL_COUNT(7,10)" \
-d "Generate event when tmp > 90% full for 7 out of 10 last
\observations" \ -s 'Name=="/tmp"' "tmp space used"
```

9. To define a condition with the name adapter stability to check for adapter status that has changed four times within one minute, including comments, enter:

```
mkcondition -r IBM.NetworkInterface \
-e "OpState != OpState@P __QUAL_RATE(4,60)" \
-d "Generate event when OpState is changed 4 times within 1 minute" \
"adapter stability"
```

10. To define a condition for a batched event called tmp space used to check the percentage of space used by **/tmp** that is greater then 90%, with a batch interval of 5 and a batch event file retention period of 72 hours, enter:

```
mkcondition -r IBM.FileSystem -e "PercentTotUsed > 90" -b 5,,72 "tmp space used"
```

11. To define a condition called tmp space used to check that percentage of space used by **/tmp** that is greater then 90%, with audit logging enabled only in case of errors, enter:

```
mkcondition -r IBM.FileSystem -e "PercentTotUsed > 90" -g 1 "tmp space used"
```

These examples apply to management domains:

1. To define a condition with the name "FileSystem space used" to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor all nodes in the domain, run this command on the management server:

```
 mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -m d "FileSystem space used"
```

2. To define a condition with the name "FileSystem space used" to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor nodes nodeA and nodeB in the domain, run this command on the management server:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -n nodeA,nodeB -m p \
"FileSystem space used"
```

3. To define a condition with the name "nodeB FileSystem space used" on nodeB to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor the condition with local scope, run this command on the management server:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -m l -p nodeB \
"nodeB FileSystem space used"
```

4. To define a condition with the name "local FileSystem space used" to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor the local node, run this command on a managed node:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -m l "local FileSystem space used"
```

These examples apply to peer domains:

1. To define a condition on nodeA with the name "FileSystem space used" to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor all nodes in the domain, run this command:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -m p -p nodeA "FileSystem space used"
```

2. To define a condition on nodeC with the name "FileSystem space used" to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor nodes nodeA and nodeB in the domain, run this command:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -n nodeA,nodeB -m p -p nodeC \
"FileSystem space used"
```

3. To define a condition with the name "local FileSystem space used" on nodeB to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor the local node only, run this command:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -m l -p nodeB "local FileSystem space used"
```

Location

/opt/rsct/bin/mkcondition

mkcondresp Command

Purpose

Creates a link between a condition and one or more responses.

Syntax

`mkcondresp` [`-h`] [`-TV`] *condition*[`:`*node_name*] *response1* [*response2*...]

Description

The `mkcondresp` command creates a link between a condition and one or more responses. A link between a condition and a response is called a *condition/response association.* This command creates one or more condition/response associations; it does not start monitoring. In a cluster environment, the condition and the response must be defined on the same node. You can start monitoring for this condition and its linked responses later using the `startcondresp` command.

To lock a condition/response association, use the `-L` flag of the `rmcondresp`, `startcondresp`, or `stopcondresp` command.

Flags

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

condition
Specifies the name of the condition to be linked to the response. The condition is always specified first.

node_name
Specifies the node in the domain where the condition is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

***response1* [*response2*...]**
Specifies one or more response names. All responses are linked to *condition*.

Security

The user needs write permission for the `IBM.Association` resource class to run `mkcondresp`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To link the condition "FileSystem space used" to the response "Broadcast event on-shift", run this command:

```
mkcondresp "FileSystem space used" "Broadcast event on-
shift"
```

2. To link the condition "FileSystem space used" to the responses "Broadcast event on-shift" and "E-mail root anytime", run this command:

```
mkcondresp "FileSystem space used" "Broadcast event on-shift" "E-mail root
anytime"
```

These examples apply to management domains:

1. To link the condition "FileSystem space used" on the management server to the response "Broadcast event on-shift" (also on the management server), run this command on the management server:

```
mkcondresp "FileSystem space used" "Broadcast event on-
shift"
```

2. To link the condition "FileSystem space used" on the management server to the response "Broadcastevent on-shift", run this command on one of the nodes in the domain:

```
mkcondresp "FileSystem space used":nodeA "Broadcast event on-
shift"
```

This example applies to peer domains:

1. To link the condition "FileSystem space used" on node nodeA to the response "Broadcastevent on-shift" (also on nodeA), run this command on one of the nodes in the domain:

```
mkcondresp "FileSystem space used":nodeA "Broadcast event on-
shift"
```

Location

/opt/rsct/bin/mkcondresp

mkcosi Command

Purpose

Makes a Common Operating System Image (COSI) for use with thin servers.

Syntax

mkcosi **-s** *Source* [**-l** *Location*] [**-S** *Server*] [**-v**] *COSI*

Description

The `mkcosi` command creates a Common Operating System Image (COSI). A COSI is a repository that contains all the necessary software to bring a thin server up to a functional state. The `mkcosi` command takes a source (`-s` *Source*) containing installable images and attempts to install those software images into a specific location (`-l` *Location*). If the `-S` *Server* is specified, the COSI image is stored on that particular server. The result is an OS image that can be used by thin servers as its boot image and operating system.

This command is dependent upon the `bos.sysmgt.nim.master` fileset being present on the system. When this command is executed for the first time, the machine executing the command is configured as a NIM master. The `mkcosi` command uses the `nim_master_setup` command to configure the machine as a NIM master. The `-S` parameter must point to a machine that is managed by the caller of the `mkcosi` command.

Flags

Item	Description
`-l`	Specifies the full path name to a location for storing the COSI.
`-S` *Server*	Specifies the name of the machine where the COSI image resides.
`-s` *Source*	Specifies the source of installable images to be used in creating the COSI. The source can be an `lpp_source`, a device with installable media, a directory to installable images, or a remote location to installable images.
`-v`	Enables verbose debug output when the `mkcosi` command runs.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

Access Control: You must have root authority to run the `mkcosi` command.

Examples

1. To define a COSI named `cosi1` from a CD-ROM `cd0`, and to store it at the location `/export/cosi1`, enter the following command:

```
mkcosi -s cd0 -l /export/cosi1 cosi1
```

Location

`/usr/sbin/mkcosi`

Files

Item	Description
`/etc/niminfo`	Contains variables used by NIM.

mkdev Command

Purpose

Adds a device to the system.

Syntax

mkdev { **-c** *Class* **-s** *Subclass* **-t** *Type* } [**-l** *Name*] [**-a** *Attribute*=*Value*] ... [**-d** | **-S** | **-R**] [**-f** *File*] [**-h**] [**-p** *ParentName*] [**-q**] [**-w** *ConnectionLocation*]

mkdev -l *Name* [**-h**] [**-q**] [**-S**]

Description

Attention: To protect the Configuration Database, the **mkdev** command is not interruptible. Stopping this command before it is complete could result in a corrupted database.

The **mkdev** command performs the following:

- Defines and makes available a device with the given device class (**-c** *Class* flag), type (**-t** *Type* flag), subclass (**-s** *Subclass* flag), connection location (**-w** *ConnectionLocation* flag), and the device logical name of the parent (**-p** *ParentName* flag)
- Makes available the previously defined device specified by the given device logical name (**-l** *Name* flag).

You can use any combination of the **-c**, **-s**, and **-t** flags you need to uniquely identify the predefined device.

If you specify the **-d** flag, the **mkdev** command only defines the device. If you specify the **-S** flag, the **mkdev** command brings the device to the Stopped state, if this state is supported, and does not make the device available. If you do not specify either the **-d** flag or the **-S** flag, the **mkdev** command makes the device available.

If you specify the **-R** flag, the **mkdev** command configures any previously-defined parents of the specified device that are not already configured. The **-R** flag is not compatible with the **-d** and **-S** flags.

By using the **-l** flag with the **-c**, **-s**, and **-t** flags, you can specify the name of the device. If you do not use the **-l** flag, a name will be automatically generated and assigned. Not all devices support user-supplied names.

Note: Queue device names must begin with an alphabetic character.

When using the **mkdev** command, you can supply the flags either on the command line or in the specified **-f** *File* flag.

You can use the System Management Interface Tool (SMIT) **smit mkdev** fast path to run this command.

Flags

Item	Description
-a *Attribute*=*Value*	Specifies the device attribute-value pairs to be used instead of the defaults. The *Attribute*=*Value* variable can be used to specify one attribute value pair or multiple attribute value pairs for one **-a** flag. Multiple attribute-value pairs must be enclosed in quotation marks with a blank space between the pairs. For example, entering `-a Attribute=Value` lists one attribute value pair per flag, while entering `-a 'Attribute1=Value1 Attribute2=Value2'` lists more than one attribute value pair. This flag cannot be used with the **-l** flag unless the **-c**, **-s**, and **-t** flags are also used.
-c *Class*	Specifies the device class.
-d	Defines the device in the Customized Devices object class. If you specify the **-d** flag, the **mkdev** command does not make the device available. This flag cannot be used with the **-S** flag.
-f *File*	Reads the necessary flags from the *File* parameter.
-h	Displays the command usage message.
-l *Name*	Specifies the predefined device, indicated by the *Name* variable, in the Customized Devices object class when not used with the **-c**, **-s**, and **-t** flags. The **-a**, **-p**, and **-w** flags cannot be used in this case. Queue device names must begin with an alphabetic character.

Item	Description
-p *ParentName*	Specifies the device name, indicated by the *ParentName* variable, that you want assigned to the device when it is used with the **-c**, **-s**, and **-t** flags. Not all devices support this feature. This flag cannot be used with the **-l** flag unless the **-c**, **-s**, and **-t** flags are also used.
-q	Suppresses the command output messages from standard output and standard error.
-R	Configures any parents of the device that are not already configured. This flag cannot be used with the **-d** and **-S** flags.
-S	Prevents the device from being set to the Available state. This flag is only meaningful for those devices that support the Stopped state. This flag cannot be used with the **-d** flag.
-s *Subclass*	Specifies the subclass, indicated by the *Subclass* variable, of the device.
-t *Type*	Specifies the device type from the Predefined Devices object class.
-w *ConnectionLocation*	Specifies the connection location, indicated by the *ConnectionLocation* variable, on the parent. This flag cannot be used with the **-l** flag unless the **-c**, **-s**, and **-t** flags are also used.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Privilege Control: Only the root user and members of the system group should have execute (x) access to this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Auditing Events:

Event	Information
DEV_Create	Method name, parameters
DEV_Configure	Errors
DEV_Start	Device name
DEV_Change	Parameters

Examples

1. To define (but not configure) a 4.0 GB 4mm Tape Drive connected to the `scsi0` SCSI adapter and using SCSI ID 5 and LUN of 0, type the following:

```
mkdev -d -c tape -t4mm2gb -s scsi -p scsi0 -w 5,0
```

The system displays a message similar to the following:

```
rmt4 defined
```

2. To make the predefined `rmt0` tape device available to use, type the following:

```
mkdev -l rmt0
```

The system displays a message similar to the following:

```
rmt0 available
```

3. To define and configure an RS-232 tty device connected to port 0 on the IBM 8-Port EIA-232/RS-422A (PCI) Adapter with the speed attribute set to 19200, and other attributes set from the `foo` file, type the following:

```
mkdev -t tty -s rs232 -p sa3 -w 0 -a speed=19200 -f foo
```

The system displays a message similar to the following:

```
tty0 available
```

Files

Item	Description
/usr/sbin/mkdev	Contains the **mkdev** command.

mkdir Command

Purpose

Creates one or more new directories.

Syntax

mkdir [**-e**] [**-m** *Mode*] [**-p**] *Directory ...*

Description

The **mkdir** command creates one or more new directories specified by the *Directory* parameter. Each new directory contains the standard entries **.** (dot) and **..** (dot-dot). You can specify the permissions for the new directories with the **-m** *Mode* flag. You can use the **umask** subroutine to set the default mode for the **mkdir** command.

The owner-ID and group-ID of the new directories are set to the process's effective user-ID and group-ID, respectively. The setgid bit setting is inherited from the parent directory. To change the setgid bit, you can either specify the **-m** *Mode* flag or issue the **chmod** command after the creation of the directory.

Note: To make a new directory you must have write permission in the parent directory.

Flags

Item	Description
-e	Creates directories with encryption inheritance.

Item	Description
-m *Mode*	Sets the permission bits for the newly-created directories to the value specified by the *Mode* variable. The *Mode* variable takes the same values as the *Mode* parameter for the **chmod** command, either in symbolic or numeric form. When you specify the **-m** flag using symbolic format, the op characters + (plus) and - (minus) are interpreted relative to the assumed permission setting a=rwx. The + adds permissions to the default mode, and the - deletes permissions from the default mode. Refer to the **chmod**command for a complete description of permission bits and formats.
-p	Creates missing intermediate path name directories. If the **-p** flag is not specified, the parent directory of each-newly created directory must already exist. Intermediate directories are created through the automatic invocation of the following **mkdir** commands: `mkdir -p -m $(umask -S),u+wx $(dirname Directory) &&` `mkdir [-m Mode] Directory` where the `[-m Mode]` represents any option supplied with your original invocation of the **mkdir** command. The **mkdir** command ignores any *Directory* parameter that names an existing directory. No error is issued.

Exit Status

This command returns the following exit values:

Item	Description
0	All the specified directories were created successfully, or the **-p** option was specified and all the specified directories now exist.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To create a new directory called `Test` in the current working directory, enter:

   ```
   mkdir Test
   ```

   The `Test` directory is created with default permissions.
2. To create a new directory called `Test` with rwxr-xr-x permissions in the previously created `/home/demo/sub1` directory, enter:

   ```
   mkdir -m 755 /home/demo/sub1/Test
   ```
3. To create a new directory called `Test` with default permissions in the `/home/demo/sub2` directory, enter:

```
mkdir -p /home/demo/sub2/Test
```

The **-p** flag creates the /home, /home/demo, and /home/demo/sub2 directories if they do not already exist.

Files

Item	Description
/usr/bin/mkdir	Contains the **mkdir** command.

mkdirhier Command

Purpose

Creates a hierarchy of directories or a single directory.

Syntax

mkdirhier *Directory* ...

Description

The **mkdirhier** command creates the specified directories. Unlike the **mkdir** command, if any of the parent directories of the specified directory do not exist, the **mkdirhier** command creates those directories as well as the specified directory.

Example

To create a directory named **foo2** or to create a hierarchy of directories named **foo, foo1**, and **foo2**, enter:

```
mkdirhier ~/foo/foo1/foo2
```

If **foo** and **foo1** already exist then the command creates **foo2**. However, if none of them exist then the command creates all three new directories.

mkdom Command

Purpose

Creates a new domain.

Syntax

mkdom [**-R** *load_module*] [*Attribute* **=** *Value* ...] *Name*

Description

The **mkdom** command creates a new domain in the domain database. The domain attributes can be set during the domain creation phase by using the *Attribute* **=** *Value* parameter.

When the system is operating in the enhanced Role Based Access Control (RBAC) mode, modifications made to the domain database are not used for security considerations until the database is sent to the kernel security tables by using the **setkst** command.

Note: The domain id value can be lesser than or equal to 1024. The **mkdom** command enables you to create 1024 domains on the system.

If the system is configured to use multiple authentication load modules for the role-based access control (RBAC) domain database, the new RBAC domain is created in the first load module specified by the `secorder` attribute in the `domains` stanza of the `/etc/nscontrol.conf` file. Use the **-R** flag to create an RBAC domain in a specific authentication load module.

Flags

Item	Description
-R *load_module*	Specifies the loadable module that is to be used when you create an RBAC domain.

Parameters

Item	Description
Attribute **=** *Value*	Initializes a domain attribute. See the **chdom** command for valid attributes and values.
Name	Specifies a unique domain name string.

Restrictions on creating domain names: The *Name* parameter specified must be unique and is limited to a 63 single-byte printable character. While the **mkdom** command supports multibyte domain names, it is recommended that you restrict domain names to characters within the POSIX portable file name character set. Domain names must not begin with a - (dash), + (plus sign), @ (at sign), or ~ (tilde) and must not contain any space, tab, or new-line characters. You cannot use the keywords ALL, default, ALLOW_OWNER, ALLOW_GROUP, ALLOW_ALL or * as a domain name. Additionally, do not use any of the following characters within a domain string:

Item	Description
:	Colon
"	Double quotation mark
#	Number sign
,	Comma
=	Equal sign
\	Backslash
/	Forward slash
?	Question mark
'	Single quotation marks
`	Grave accent

Security

The **mkdom** command is a privileged command. Callers of the command must have activated a role that has the following authorization to run the command successfully.

Item	Description
aix.security.domains.create	Required to run the command.

Files Accessed

Item	Description
File	Mode
/etc/security/domains	rw

Examples

1. To create a domain `hrdom` and to have the **mkdom** command assign an appropriate ID value, enter the following command:

```
mkdom hrdom
```

2. To create a custom domain in Lightweight Directory Access Protocol (LDAP), enter the following command:

```
mkdom -R LDAP custom
```

mkdvd Command

Purpose

Creates multi-volume DVDs from a **mksysb**, **savevg**, or **savewpar** backup image.

Syntax

mkdvd **-r** *directory* | **-d** *dvddevice* | **-S** [**-m** *mksysbimage* | **-M** *mksysbtarget* | **-s** *savevgimage* | **-v** *savevgvolumegroup* | **-w** *savewpar_image* | **-W** *wparname*] [**-C** *cdfsdir*] [**-I** *cdimagedir*] [**-V** *dvdfsvolumegroup*] [**-B**] [**-p** *pkgsourcedir*] [**-R** | **-S**] [**-i** *image.data*] [**-u** *bosinst.data*] [**-f** *wparspecificationfile*] [**-e**] [**-P**] [**-l** *packagelist*] [**-b** *bundlefile*] [**-z** *customfile*] [**-D**] [**-U**] [**-Y**] [**-n**] [**-a**] [**-A**] [**-c**] [-Z] [**-G** | **-N**] [**-x** *file*] [**-T**]

Description

The **mkdvd** command creates a system backup image (**mksysb**) to DVD-Recordable (DVD-R, DVD-RAM) from the system **rootvg** or from a previously created **mksysb** image. It creates a volume group backup image (**savevg**) to DVD from a user-specified volume group or from a previously created **savevg** image. It also creates the backup image of a workload partition (**savewpar**) to DVD from a user-specified workload partition or from a previously created **savewpar** image.

Note: If the system has a **multibos** environment where both instances are mounted, you can restore the backup only by using the **alt_disk_mksysb** command.

For DVD media, system backups that are made with the **mkdvd** command have a limitation in that they expect the media to be 4.7 GB or larger per side. The **mkdvd** command does not process the next volume until it writes over 4 GB on the current volume, thus the use of smaller media would result in corruption when you go beyond the media capacity.

When a bootable backup of a root volume group is created, the boot image reflects the currently running kernel. If the current kernel is the 64-bit kernel, the backup boot image is also 64 bit, and it boots 64-bit systems only. If the current kernel is a 32-bit kernel, the backup boot image is 32 bit, and it can boot both 32-bit and 64-bit systems.

With the **mkdvd** command, you can create bootable and non-bootable DVDs in Rock Ridge (ISO9660) or UDF (Universal Disk Format) format.

Note: The functionality that is required to create Rock Ridge format DVD images and to write the DVD image to the DVD-RAM device is not part of the **mkdvd** command. You must supply additional code to

the **mkdvd** command to do these tasks. You can call the code by using shell scripts and then link it to **/usr/sbin/mkrr_fs** (for creating the Rock Ridge format image) and **/usr/sbin/burn_cd** (for writing to the DVD device). Both links are called from the **mkdvd** command.

Some sample shell scripts are included for different vendor-specific routines. You can find these scripts in **/usr/samples/oem_cdwriters**.

If you do not supply any file systems or directories as command parameters, the **mkdvd** command creates the necessary file systems and removes them when the command finishes running. File systems that you supply are checked for adequate space and write access.

Note: If the **mkdvd** command creates file systems in the backup volume group, they are excluded from the backup.

If you must create multi-volume DVDs because the volume group image does not fit on one DVD, the **mkdvd** command provides instructions for DVD replacement and removal until all the volumes have been created.

Flags

Item	Description
-a	Does not back up extended attributes or NFS4 ACLs.
-A	Backs up DMAPI file system files.
-b *bundlefile*	Gives the full path name of the file that contains a list of filesets to be installed after the **mksysb** is restored. This file is copied to **./usr/sys/inst.data/user_bundles/***bundle_file* in the DVD file system and also copied to RAM in case the DVD is unmounted. The file would be listed as BUNDLES=**/../usr/sys/inst.data/user_bundles/***bundlefile* in the **bosinst.data** file.
-B	Prevents **mkdvd** from adding boot images (non-bootable DVD) to the DVD. Use this flag if you create a **mksysb** DVD that you will not boot. Before you install the non-bootable **mksysb** DVD, you must boot a same level (V.R.M.) product media. The **mkdvd** command defaults to creating a bootable DVD for the machine type of the source system. For more information, see the Notes section.
-c	Does not compress or pack files as they are backed up.
-C *cdfsdir*	Specifies the file system that is used to create the DVD file system structure, which must have up to 4.38 GB for DVD sized images. The DVD image consumes only as much room as is necessary to contain all the data on the DVD. If you do not specify the **-C** flag and the **/mkcd/cd_fs** directory exists, the **mkdvd** command uses that directory. If you do not specify the **-C** flag and the **/mkcd/cd_fs** directory does not exist, the **mkdvd** command creates the file system **/mkcd/cd_fs** and removes it when the command finishes running. The command creates the file system in the volume group that is indicated with the **-V** flag, or **rootvg** if that flag is not used. Each time that you invoke the **mkdvd** command, a unique subdirectory (by using the process id) is created under the **/mkcd/cd_fs** directory, or in the directory that is specified with the **-C** flag. **Note:** If performing DVD sized backups, the file systems must be *large file enabled*. This type of backup also requires setting the file **ulimit** size to *unlimited*.
-d *dvddevice*	Indicates the DVD-R or DVD-RAM device (`/dev/cd1`, for instance). This flag is required unless you use the **-S** flag.

Item	Description
-D	Turns on the debug output information feature. The default is no debug output.
-e	Excludes the files and directories from the backup image that is listed in the **/etc/exclude.***volume_group* file. You cannot use this flag with the **-m** or **-s** flags.
-f *wparspecificationfile*	Specifies the user-supplied WPAR specification file. This specification file of workload partition takes precedence over the **wpar.spec** file in the **savewpar** image. If you do not use the **-f** flag, the **mkdvd** command restores the **wpar.spec** from the specified **savewpar** image, or generates a new **wpar.spec** file during the creation of **savewpar**.
-i *image.data*	Specifies the user-supplied *image.data* file. This data file takes precedence over the **image.data** file in the **mksysb** image. If you do not specify the **-i** flag, then the **mkdvd** command restores the **image.data** from the given **mksysb** image, or generates a new **image.data** file during the creation of **mksysb**. **Note:** The **-i** flag cannot be used to specify a user-supplied *vgname.data* file for use with a **savevg** image.
-I *cdimagesdir*	Specifies the directory or file system where the final DVD images are stored before they are written to the DVD-R or DVD-RAM device. If this flag is not used, the **mkdvd** command uses the **/mkcd/*cd_images*** directory if it already exists. If not, the command creates the **/mkcd/*cd_images*** file system in the volume group that is given with the **-V** flag, or in **rootvg** if that flag is not used. If the **mkdvd** command creates the file system, it is removed upon command completion, unless either the **-R** or **-S** flag is used. If the **-R** or **-S** flag is used, consideration must be made for adequate file system, directory, or disk space, especially when you create multi-volume DVDs. The DVD image consumes only as much room as is necessary to contain all the data on the DVD. **Note:** If performing DVD sized backups, the file systems must be *large file enabled*. This type of backup also requires setting the file **ulimit** size to *unlimited*.
-l *packagelist*	Specifies the file that contains a list of additional packages you want copied to the **./usr/lpp/inst.images** directory of the DVD file system. The images are copied from the location that is named with the **-p** flag. If you use the **-l** flag, you must also use the **-p** flag.
-m *mksysbimage*	Specifies a previously created **mksysb** image. If you do not specify the **-m** flag, the **mkdvd** command calls **mksysb**. For more information about where the **mksysb** image is placed, see the **-M** flag.

Item	Description
-M *mksysbtarget*	States the directory or file system where the **mksysb** or **savevg** image is stored if a previously created backup is not given with the **-m** or **-s** flags. If the **-M** flag is not used and a **mksysb** or **savevg image** is not provided, the **mkdvd** command verifies that **/mkcd/***mksysbimage* exists. If the directory does not exist, then the **mkdvd** command creates a separate file system, **/mkcd/***mksysbimage*, where the **mksysb** or **savevg** images are temporarily stored. The command creates the file system in the volume group that is given with the **-V** flag, or in **rootvg** if that flag is not used. **Note:** If performing DVD sized backups, the file systems must be *large file enabled*. This type of backup also requires setting the file **ulimit** size to *unlimited*.
-n	Backs up user volume group information and administration data files. This flag backs up files such as **/tmp/vgdata/vgname/vgname.data** and map files, if any exist. This flag does not back up user data files. This backup can be used to create a user volume group without restoring user data files. This action cannot be done to rootvg.
-N	Includes file systems that belong to a workload partition (WPAR) in the defined state in the system backup. **Note:** To be included in the backup, all file systems that belong to a WPAR in the defined state must be in the rootvg volume group.
-p *pkgsourcedir*	Names the directory or device that contains device and kernel package images. The device must be a CD or DVD device (for example, `/dev/cd0`). If you use the same DVD-R or DVD-RAM device that you gave with the `-d` flag, the product media must be inserted into the drive first. The `mkdvd` command then prompts you to insert the writable DVD before the actual DVD creation.
-P	Creates physical partition mapping during the **mksysb** or **savevg** creation. You cannot use this flag with the **-m** or **-s** flags.
-r *directory*	Indicates existing directory structure to burn onto a DVD. This flag makes a DVD image that is a copy of the specified directory structure.
-R	Prevents the **mkdvd** command from removing the final DVD images. The **mkdvd** command defaults by removing everything that it creates when it finishes running. The **-R** flag allows multiple DVD image sets to be stored, or for DVD creation (burn) to occur on another system. If multiple volumes are needed, the final images are uniquely named by using the process ID and volume suffixes.
-s *savevgimage*	Indicates a previously created **savevg** image. See Notes for details.
-S	Stops the **mkdvd** command before it writes to the DVD-R or DVD-RAM without removing the final DVD images. The **-S** flag allows multiple DVD sets to be created, or for DVDs to be created on another system. The images remain in the directory marked by the **-I** flag, or in the **/mkcd/*cd_images*** directory if the **-I** flag is not used. If multiple volumes are required, the final images are uniquely named by using the process ID and volume suffixes.

Item	Description
-T	Creates backup by using snapshots. This command applies only to JFS2 file systems. When you specify the **-T** flag to use snapshots for creating a volume group backup, external JFS2 snapshots are created. Snapshots allow for a point-in-time image of a JFS2 file system and thus, do not require a system to be put into a temporarily inactive state. The size of the snapshot is 2% - 15% of the size of the file system. The snapshot logical volumes are removed when back up is complete. However, snapshots are not removed if a file system already has other snapshots. Additionally, if a file system has internal snapshots, external snapshots cannot be created and thus, snapshots are not used for creating the backup of the file system. The use of the **-T** flag does not affect any JFS file systems that are present in the volume group that is being backed up. These file systems are backed up in the same manner as done previously.
-u *bosinst.data*	Specifies the user-supplied *bosinst.data* file. This data file takes precedence over the **bosinst.data** file in the **mksysb** image. If you do not specify the **-u** flag, then the **mkdvd** command restores **bosinst.data** from the specified **mksysb** image, or generates a new **bosinst.data** file during the creation of **mksysb**.
-U	Creates a **UDF** (Universal Disk Format) file system on DVD-RAM media. It does not require the amount of free space that is needed to create Rock Ridge format backups. It does not need the **/mkcd/cd_fs** and **/mkcd/cd_images** file systems. Therefore, the only temporary disk space it needs is to create the backup image that will be copied to the media. This means that the **-I** and **-C** flags do not apply to the **-U** flag. Because the backup is copied to the media, images cannot be created and burned later or on another system. So, the **-R** flag and **-S** flag do not apply when you use the **-U** flag. You must specify a device to write to with the **-d** flag. The **-U** flag does not use the **/usr/sbin/mkrr_fs** or **/usr/sbin/burn_cd** file systems.
-v *savevgvolumegroup*	Denotes the volume group to be backed up using the **savevg** command. See Notes for details. For more information about where the **savevg** image is placed, see the **-M** flag.
-V *dvdfsvolumegroup*	Indicates the volume group that is used when you create the file systems needed for the **mkdvd** command. If the **-V** flag is not given and a file system is needed but not there (because it was not supplied with other flags), then **rootvg** is the default volume group for creating the file systems. If the **mkdvd** command creates the file systems in the backup volume group, those file systems are not included as part of the backup image. The **mkdvd**-created file systems are removed upon the command's completion.
-w *savewparimage*	Indicates a previously created **savewpar** image.
-W *wparname*	Denotes the workload partition to be backed up using the **savewpar** command.
-Y	Accepts licenses.

Item	Description
-z *customsfile*	States the full path name of the file to be copied to the root directory of the DVD file system. This file can be a customization script that is specified in the **bosinst.data** file, such as `CUSTOMIZATION_FILE=filename`. For example: If the file `my_script` is in **/tmp** on the machine where **mkdvd** is running, then enter `-z/tmp/my_script` and specify `CUSTOMIZATION_FILE=my_script`. The code copies the script to the root directory of the RAM file system before it runs.
-Z	Specifies that the Encrypted file system (EFS) information for all the files, directories, and file systems is not backed up.
-G	Excludes WPAR file systems from the system backup. This flag is not valid with **-N** flag.
-x *file*	Excludes the file systems that are listed in the file from the system backup. File system mount points must be listed one per line.

Note: Use care when you exclude file systems as a resulting backup can be unusable for system restoration.

Note:

- If you are creating a non-bootable DVD (by using the **-B** flag), you cannot use the **-p** or **-l** flags.
- If you are creating a non-bootable DVD with a **savevg** image (by using the **-s** or **-v** flags), you cannot use the **-p**, **-l**, **-u**, **-i**, **-z**, or **-b** flags.

Examples

1. To generate a bootable system backup to the DVD-R device named `/dev/cd1`, enter the following command:

   ```
   mkdvd -d /dev/cd1
   ```

2. To generate a system backup to the DVD-R or DVD-RAM device named `/dev/cd1`, enter the following command:

   ```
   mkdvd -d /dev/cd1
   ```

3. To generate a non-bootable volume group backup of the volume group `myvg` to `/dev/cd1`, enter the following command:

   ```
   mkdvd -d /dev/cd1 -v myvg
   ```

   Note: All **savevg** backup images are non-bootable.

4. To generate a non-bootable backup of the workload partition `mywpar` to /dev/cd1, enter the following command:

   ```
   mkdvd -d /dev/cd1 -W mywpar
   ```

   Note: All **savewpar** backup images are not bootable.

5. To generate a non-bootable backup of the workload partition `mywpar` to **/dev/cd1** from the previously generated **savewpar** image **/wparbackups/mywpar.bff**, enter the following command:

   ```
   mkdvd -d /dev/cd1 -w /wparbackups/mywpar.bff
   ```

6. To create a DVD or DVD that duplicates an existing directory structure such as the following example:

```
/mycd/a
/mycd/b/d
/mycd/c/f/g
```

enter the following command:

```
mkdvd -r /mycd -d /dev/cd1
```

After you mount with `mount -o ro /dev/cd1 /mnt`, **cd** to `/mnt`; a `find . -print` command displays:

```
./a
./b
./b/d
./c
./c/f
./c/f/g
```

Files

Item	Description
/usr/bin/mkdvd	Contains the **mkdvd** command.

mkfifo Command

Purpose

Makes first-in-first-out (FIFO) special files.

Syntax

mkfifo [**-m** *Mode*] *File* ...

Description

The **mkfifo** command creates FIFO special files specified by the *File* parameter, in the order specified. If the **-m** *Mode* flag is not specified, the file mode of the FIFO file is the bitwise inclusive OR of the **S_IRUSR**, **S_IWUSR**, **S_IRGRP**, **S_IWGRP**, **S_IROTH**, and **S_IWOTH** permissions as modified by the file mode creation (see the **umask** command).

The **mkfifo** command functions similarly to the **mkfifo** subroutine.

Flags

Item	Description
-m *Mode*	Sets the file permission bits of the newly created FIFO file to the specified mode values. The *Mode* variable is the same as the mode operand defined for the **chmod** command. The characters + (plus sign) and - (minus sign), if used, are interpreted relative to the initial value **a=rw** (that is, having permissions of **rw-rw-rw-**).

Exit Status

This command returns the following exit values:

Item	Description
0	All the specified FIFO special files were created successfully.
>0	An error occurred.

Examples

1. To create a FIFO special file with permissions `prw-r—r—`, enter:

   ```
   mkfifo -m 644 /tmp/myfifo
   ```

   This command creates the `/tmp/myfifo` file with read/write permissions for the owner and read permission for the group and for others.
2. To create a FIFO special file using the - (minus sign) operand to set permissions of `prw-r——-`, enter:

   ```
   mkfifo -m g-w,o-rw /tmp/fifo2
   ```

   This command creates the `/tmp/fifo2` file, removing write permission for the group and all permissions for others.

   Note: If more than one file is created using the - (minus sign) operand, separate each mode specifier with a comma and no spaces.

Files

Item	Description
/usr/bin/mkfifo	Contains the **mkfifo** command.

mkfilt Command

Purpose

Activates or deactivates the filter rules.

Syntax

mkfilt **-v 4** | **6** [**-d**] [**-u**] [**-z** **P** | **D**] [**-g** *start* | *stop*] [**-i**]

Description

Use the **mkfilt** command to activate or deactivate the filter rules. This command can also be used to control the filter logging function. IPsec filter rules for this command can be configured using the `genfilt` command or IPsec smit (IP version 4 or IP version 6).

Flags

Item	Description
-v	IP version of the rules you want to activate. The value of **4** specifies IP version 4 and the value of **6** specifies IP version 6. The default (when this flag is not used) is to activate both IP version 4 and IP version 6. All the filter rules defined in the filter rule table for the IP version(s) will be activated or deactivated.
-d	Deactivates the active filter rules. This flag cannot be used with the **-u** flag.

Item	Description
-u	Activates the filter rules in the filter rule table. This flag cannot be used with the **-d** flag.
-z	Sets the action of the default filter rule to Permit (**P**) or Deny (**D**). The default filter rule is the last rule in the filter rule table that will apply to traffic that does not apply to any other filter rules in the table. Setting the action of this rule to Permit will allow all traffic that does not apply to any other filter rules. Setting this action to Deny will not allow traffic that does not apply to any other filter rules.
-g	This flag is used to either start (**start**) or stop (**stop**) the log functionality of the filter rule module.
-i	Initialization flag. This flag only applies when the **-u** flag is also used. If the **-i** flag is used, all the filter rules with an "active" status will be activated. If not used, all the filter rules in the filter rule table will be activated.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

mkfont Command

Purpose

Adds a font path name to the Object Data Manager (ODM) that is loaded by the low function terminal (LFT) at boot time.

Syntax

mkfont [*FontPathName*]

Description

The **mkfont** command adds a fully qualified font file path name to the ODM. At boot time, the LFT loads the new font and any other fonts found in the ODM. The list of font information acquired by the LFT is passed to the default display device driver. The display driver selects from this list the font that best fits the display. If a default font was selected using the **chfont** command, the device driver uses that font.

Note: This command can be run only from an LFT.

You can use the System Management Interface Tool (SMIT) **smit mkfont** fast path to run this command.

Parameter

Item	Description
FontPathName	The fully qualified pathname of a font file.

Security

The user must have root authority to execute this command.

Example

To add the font file **/usr/lpp/fonts/Rom10.snf**, enter the following command:

```
mkfont   /usr/lpp/fonts/Rom10.snf
```

Files

Item	Description
/bin/mkfont	Contains the **mkfont** command.
/usr/lpp/fonts	Contains the font directory.

mkfontdir Command

Purpose

Creates a **fonts.dir** file from a directory of font files.

Syntax

mkfontdir [*DirectoryName* ...]

Description

The **mkfontdir** command creates a **fonts.dir** file from a directory of font files. For each directory argument, the **mkfontdir** command reads all of the bitmapped font files in the directory, searching for properties named FONT or the name of the file stripped of its suffix. These are used as font names, which are written to the **fonts.dir** file in the directory along with the name of the font file. The **fonts.dir** file is then used by the X server and the Font server to determine which fonts are available.

The kinds of font files read by the **mkfontdir** command depend upon the configuration parameters and typically include the following formats:

Item	Description
Portable Compile Format	(suffix **.pcf**)
Compressed PCF	(suffix **.pcf.Z**)
Server Natural Format	(suffix **.snf**)
Compressed SNF	(suffix **.snf.Z**)
Bitmap Distribution Format	(suffix **.bdf**)
Compressed BDF	(suffix **.bdf.Z**)

If a font exists in multiple formats, the most efficient format is used (PCF format before SNF then BDF formats).

Scalable fonts are not automatically recognized by **mkfontdir**. You can contruct a **fonts.scale** file (the format is identical to that in the **fonts.dir** file) containing entries for scalable fonts. Then, when you run **mkfontdir** on a directory, it copies entries from the **fonts.scale** file in that directory into the **fonts.dir** file it constructs in that directory.

You can create the **fonts.alias** file, which can be put in any directory of the font path, to map new names to existing fonts. This file should be edited by hand. The format is two columns separated by white space, with the first column containing aliases and the second column containing font-name patterns.

When a font alias is used by an X client, the X server searches for the name it references by looking through each font directory in turn. Therefore, the aliases and the font files do not need to be in the same directory.

To embed white space in aliases or font-name patterns, enclose them in double-quotation marks. To embed double-quotation marks, or any other characters, precede each character with a \ (backslash).

```
"magic-alias with spaces" "\"font\name\"with quotes"
regular-alias                  fixed
```

If the character string **FILE_NAMES_ALIASES** stands alone on a line, each file name in the directory when stripped of its suffix (such as **.pcf** or **.pcf.Z**) is used as an alias for that font.

The X server and the Font Server look for **fonts.dir** and **fonts.alias** files in each directory in the font path each time the font path is set.

Examples

To create a **fonts.dir** file from a directory of font files, enter:

```
mkfontdir DirectoryName
```

If no directory name is specified, the **mkfontdir** command reads the current directory.

Files

Item	Description
/usr/lib/X11/fonts	Is the directory containing font files, **fonts.dir** and **fonts.alias** files.

mkfs Command

Purpose

Makes a file system.

Syntax

mkfs [**-b** *Boot*] [**-l** *Label*] [**-i** *i-Nodes*] [**-o** *Options*] [**-p** *Prototype*] [**-s** *Size*] [**-v** *VolumeLabel*] [**-V** *VfsName*] *Device*

Description

The **mkfs** command makes a new file system on a specified device. The **mkfs** command initializes the volume label, file system label, and startup block.

The *Device* parameter specifies a block device name, raw device name, or file system name. If the parameter specifies a file system name, the **mkfs** command uses this name to obtain the following parameters from the applicable stanza in the **/etc/filesystems** file, unless these parameters are entered with the **mkfs** command:

Item	Description
dev	Device name
vol	Volume ID
size	File system size
boot	Program to be installed in the startup block
vfs	Definition of the virtual file system
options	File-system implementation-specific options of the form *Keyword*, *Keyword*=*Value*

Note:

1. The file system is created with the setgid (set group ID) bit enabled. The setgid bit determines the default group permissions. All directories created under the new file system have the same default group permissions.
2. The **mkfs** command does not alter anything in a mounted file system, including the file system label. The file system label changes when you change the mount point, unless the file system is mounted.
3. For information about creating a file system on a striped logical volume, refer to "File Systems on Striped Logical Volumes" on page 2455 the **mklv** documentation.
4. To create a JFS2 file system on a logical volume, the minor number of the logical volume must be greater than 3071.

Flags

Item	Description
-b *Boot*	Names the program to be installed in block 0 of the new file system.
-i *i-Nodes*	Specifies the initial number of i-nodes on the file system. This flag is ignored when creating a journaled file system.
-l *Label*	Specifies the file system label for the new file system.
-o *Options*	Specifies a comma-separated list of virtual file system implementation-specific options.

The following options are specific to the Journaled File System (JFS):

Item	Description
-o ag={ **8** | **16** | **32** | **64** }	Specifies the allocation group size in megabytes. An allocation group is a grouping of i-nodes and disk blocks similar to BSD cylinder groups. The default ag value is 8.
-o bf={ **true** | **false** }	Specifies a large file enabled file system. See JFS and large files for more information. If you do not need a large file enabled file system, set this option to false; this is the default. Specifying **bf=true** requires a fragment size of 4096 and **compress=no**.
-o frag={ **512** | **1024** | **2048** | **4096** }	Specifies the JFS fragment size in bytes. A file system fragment is the smallest unit of disk storage that can be allocated to a file. The default fragment size is 4096 bytes.
-o compress={ **no** | **LZ** }	Specifies data compression. If you do not want data to be compressed, set this option to no. Selecting compression requires a fragment size of 2048 or less.
-o nbpi={ **512** | **1024** | **2048** | **4096** |**8192** | **16384** | **32768** | **65536** | **131072** }	Specifies the number of bytes per i-node (**nbpi**). The **nbpi** is the ratio of file system size in bytes to the total number of i-nodes. The default **nbpi** value is 4096 bytes.

Notes:

- The **ag**, **bf**, **compress**, **frag**, and **nbpi** attributes are set at file system creation and cannot be changed after the file system is successfully created. The **size** attribute defines the minimum file system size, and you cannot decrease it after the file system is created.
- The root file system (**/**) cannot be compressed.
- Some **nbpi** values and allocation group sizes are mutually exclusive. See "Understanding JFS Size Limitations" for information.

The following options are specific to the Enhanced Journaled File System:

Item	Description
-o agblksize={ 512 | 1024 | 2048 | 4096 }	Specifies the Enhanced Journaled File System (JFS2) block size in bytes. A file system block is the smallest unit of disk storage that can be allocated to a file. The default block size is 4096 bytes.
-o isnapshot={yes|no}	Specifies whether the file system can support internal snapshots. Specifying **yes** enables the file system to support internal snapshots and **v2** extended attributes. The resulting file system is not compatible with releases earlier than AIX 6.1.
-o name=*mountpoint*	Specifies the mount point for the file system.
-o log=*LVName*	Specifies the log logical volume name. The specified logical volume is the logging device for the new JFS2.
-o log=INLINE	Specifies to place the log in the logical volume with the JFS2 file system. The **INLINE** log will default to .4% of the logical volume size if logsize is not specified.
-o logsize=*Value*	Specifies the size for an **INLINE** log in MBytes. Ignored if **INLINE** log not being used. Cannot be greater than 2047 MBytes and cannot be greater than 10% of the size of the file system.
-o ea={v1 | v2}	Specifies the format to be used to store named extended attributes in the JFS2 file system. The **v2** format provides support for scalable named extended attributes as well as support for NFS4 ACLs. The **v1** format is compatible with prior releases of AIX. The default format is **v1**.
-o efs={yes|no}	Specifies encryption. Specifying **yes** enables encryption for the JFS2 file system. • If the **efs** attribute is set to **yes**, the **mkfs** command automatically creates the JFS2 file system with the extended attribute format set to v2. The **ea** attribute is not required. • If the **efs** attribute is set to **no**, the **mkfs** command creates a file system that is not encrypted.
`-o vix={yes|no}`	Specifies whether the file system can allocate i-node extents smaller than the default of 16 KB, if there are no contiguous 16 KB extents free in the file system. After a file system is enabled for small free extents, the file system cannot be accessed on AIX 5.1 or earlier releases. **yes** The file system can allocate variable-length i-node extents. This is the default value beginning with AIX 6.1. **no** The file system must use the default size of 16 KB for i-node extents. This has no effect if the file system already contains variable-length i-node extents.
`-o maxext=Value`	Specifies the maximum size of a file extent in file system blocks. A zero value implies that the JFS2 default maximum should be used. Values less than 0 or exceeding maximum supported extent size of 16777215 are invalid.

Note: The **agblksize** attribute is set at file system creation and cannot be changed after the file system is successfully created.

The **ea** attribute format is set at file system creation. The **chfs** command can be used to convert the extended attribute format from **v1** to **v2**, but the format cannot be converted back. The conversion is done

in an on-demand manner such that any extended attribute or ACL writes cause the conversion for that file object to occur.

Item	Description
-p *Prototype*	Specifies the name of the prototype file when you create a JFS file system. Options specified on the command line override attributes in the prototype file.
-s *Size*	Specifies the size of the file system. Size can be specified in units of 512-byte blocks, megabytes (suffix M must be used) or gigabytes (suffix G must be used). See JFS and JFS2 for more information.

Notes:

- The volume group in which the file system resides defines a maximum logical volume size and also limits the file system size.
- The **-s** *Size* flag specifies the minimum file size and cannot be decreased after the file system has been successfully created.
- The `maxext` attribute is ignored in older releases even if the filesystem was created with it on a later release .

Item	Description
-v *VolumeLabel*	Specifies the volume label for the new file system.
-V *VfsName*	Specifies the virtual file system (VFS) type. The VFS must have an entry in the **/etc/vfs** file.

Restriction: The **mkfs** command prevents EFS File System enablement of the following File Systems (mount points) because the security infrastructure (kernel extensions, libraries, and so on) are not available when you start the system. The following list is of known File Systems (mount points) that you cannot use:

"/"
"/usr"
"/var"
"/opt"

Security

Access Control: Only the root user or a member of the **system** group can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To specify the volume and file system name for a new file system, type:

   ```
   mkfs  -lworks  -vvol001 /dev/hd3
   ```

   This command creates an empty file system on the `/dev/hd3` device, giving it the volume serial number `vol001` and file system name `works`. The new file system occupies the entire device. The file system has a default fragment size (4096 bytes) and a default nbpi ratio (4096).
2. To create a file system with nondefault attributes, type:

   ```
   mkfs  -s 8192  -o nbpi=2048,frag=512 /dev/lv01
   ```

This command creates an empty 4 MB file system on the /dev/lv01 device with 512-byte fragments and 1 i-node for each 2048 bytes.

3. To create a large file enabled file system, type:

```
mkfs -V jfs -o nbpi=131072,bf=true,ag=64 /dev/lv01
```

This creates a large file enabled JFS file system with an allocation group size of 64 megabytes and 1 i-node for every 131072 bytes of disk. The size of the file system will be the size of the logical volume lv01.

4. To create a file system with nondefault attributes, type:

```
mkfs -s 4M -o nbpi=2048, frag=512 /dev/lv01
```

This command creates an empty 4 MB file system on the **/dev/lv01** device with 512-byte fragments and one i-node for each 2048 bytes.

5. To create a JFS2 file system which can support NFS4 ACLs, type:

```
mkfs -V jfs2 -o ea=v2 /dev/lv01
```

This command creates an empty file system on the **/dev/lv01** device with **v2** format for extended attributes.

Files

Item	Description
/etc/vfs	Contains descriptions of virtual file system types.
/etc/filesystems	Lists the known file systems and defines their characteristics.

mkgroup Command

Purpose

Creates a new group.

Syntax

mkgroup [**-R** *load_module*] [**-a**] [**-A**] [*Attribute*=*Value* ...] *Group*

Description

The **mkgroup** command creates a new group. The *Group* parameter must be a unique string (whose length is administrator-configurable by way of the chdev command) and cannot be the **ALL** or **default** keywords. By default, the **mkgroup** command creates a standard group. To create an administrative group, specify the **-a** flag. You must be the root user or a user with GroupAdmin authorization to create an administrative group.

To create a group with an alternate Identification and Authentication (I&A) mechanism, the **-R** flag can be used to specify the I&A load module used to create the group. Load modules are defined in the **/usr/lib/security/methods.cfg** file.

You can use the System Management Interface Tool (SMIT) **smit mkgroups** fast path to run this command.

The **mkgroup** command always checks the target group registry to make sure the ID for the new account is unique to the target registry. The **mkgroup** command can also be configured to check all group registries of the system using the **dist_uniqid** system attribute. The **dist_uniqid** system attribute is an

attribute of the **usw** stanza of the **/etc/security/login.cfg** file, and can be managed using the **chsec** command.

The `dist_uniqid` system attribute has the following values:

- `never` - Does not check for ID collision against the non-target registries. This is the default setting.
- `always` - Checks for ID collision against all other registries. If collision is detected between the target registry and any other registry account creation or modification fails.
- `uniqbyname` - Checks for ID collision against all other registries. Collision between registries is allowed only if the account to be created has the same name as the existing account.

Note: ID collision detection in the target registry is always enforced regardless of the `dist_uniqid` system attribute.

The **uniqbyname** system attribute setting works well against two registries. With more than two registries, and with ID collision already existing between two registries, the behavior of the **mkgroup** command is unspecified when creating a new account in a third registry using the colliding ID values. The new account creation might succeed or fail depending the order in which the registries are checked.

The check for ID collision only enforces ID uniqueness between the local registry and remote registries or between remote registries. There is no guarantee of ID uniqueness between the newly created account on the remote registry and existing local users on other systems that make use of the same remote registry. The **mkgroup** command bypasses a remote registry if the remote registry is not reachable at the time the command is run.

If Encrypted File System (EFS) is enabled on the system, the **mkgroup** command updates the **/etc/security/group** file with EFS attributes (default values are added if you do not specify the attributes on the command line). If you do not specify `efs_keystore_access=none`, the **mkgroup** command creates the group keystore if at least one of the users has a keystore.

If the **mkgroup** command returns with the return code of 3, the keystore for the group is not created, but the **mkgroup** command creates the group.

Note: You can later create the group keystore by using the **efskeymgr** command.

Restrictions on Creating Group Names

To prevent login inconsistencies, you should avoid composing group names entirely of uppercase alphabetic characters. While the **mkgroup** command supports multibyte group names, it is recommended that you restrict group names to characters with the POSIX portable filename character set.

To ensure that your user database remains uncorrupted, you must be careful when naming groups. Group names must not begin with a - (dash), + (plus sign), @ (at sign), or ~ (tilde). You cannot use the keywords **ALL** or **default** in a group name. Additionally, do not use any of the following characters within a group-name string:

Item	Description
:	Colon
"	Double quote
#	Pound sign
,	Comma
=	Equal sign
\	Back slash
/	Slash
?	Question mark
'	Single quote

Item	Description
`	Back quote

Finally, the *Name* parameter cannot contain any space, tab, or new-line characters.

Flags

Item	Description
-a	Creates an administrative group. Only the root user can use this flag.
-A	Sets the group administrator to the person who invoked the **mkgroup** command.
-R *load_module*	Specifies the loadable I&A module used to create the user.
Attribute=*Value*	Initializes a group with a specific attribute. See the **chgroup** command for more information about the group attributes.

Exit Status

This command returns the following exit values:

Item	Description
0	The command runs successfully and all requested changes are made.
>0	An error occurred. The printed error message lists further details about the type of failure.

Security

Access Control: This command should grant execute (x) access only to the root user and members of the security group. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the root user with the **setuid** (SUID) bit set.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

To get the full functionality of the command, besides the **accessauths**, the role must also have the **aix.security.group.change** authorization. If Encrypted File System (EFS) is enabled on the system, the role must also have the **aix.security.efs** authorization to create the group keystore.

Files Accessed:

Mode	File
rw	**/etc/passwd**
rw	**/etc/security/user**
rw	**/etc/security/limits**
rw	**/etc/security/environ**
rw	**/etc/group**
rw	**/etc/security/group**
r	**/usr/lib/security/mkuser.default**
x	**/usr/lib/security/mkuser.sys**

Auditing Events:

Event	Information
USER_Create	user

Limitations

Creating a group may not be supported by all loadable I&A modules. If the loadable I&A module does not support creating a group, an error is reported.

Examples

1. To create a new group account called `finance`, type:

   ```
   mkgroup finance
   ```

2. To create a new administrative group account called `payroll`, type:

   ```
   mkgroup -a payroll
   ```

   Only the root user can issue this command.

3. To create a new group account called `managers` and set yourself as the administrator, type:

   ```
   mkgroup -A managers
   ```

4. To create a new group account called `managers` and set the list of administrators to `steve` and `mike`, type:

   ```
   mkgroup adms=steve,mike managers
   ```

   The users `steve` and `mike` must already exist on the system.

5. To create a new group that is a LDAP I&A loadable module user, type:

   ```
   mkgroup -R LDAP monsters
   ```

Files

Item	Description
/usr/bin/mkgroup	Contains the **mkgroup** command.
/etc/group	Contains the basic attributes of groups.
/etc/security/group	Contains the extended attributes of groups.
/etc/passwd	Contains basic user information.
/etc/security/passwd	Contains password information.

mkhosts Command

Purpose

Generates the host table file.

Syntax

/usr/sbin/mkhosts [**-v**] *HostFile*

Description

The **mkhosts** command can be used to generate a hashed host database, using the filename specified by the *HostFile* parameter. It is not used if name resolution is performed by the **named** daemon. The host file is usually the **/etc/hosts** file, and in any case must be in the same format as the **/etc/hosts** file.

The **mkhosts** command generates database files named **hostfile.pag** and **hostfile.dir**. Updates to these files are built in a set of temporary files named **hostfile.new.pag** and **hostfile.new.dir**. The temporary files are copied into the database files only if the **hostfile.new.pag** and **hostfile.new.dir** files are built without errors.

The host file is used by one version of the **gethostbyaddr** and **gethostbyname** library routines for name resolution.

> **Note:** The version of the **gethostbyaddr** and **gethostbyname** library routines on this operating system do not support the **hostfile.pag** and **hostfile.dir** files.

After creating the host file, you can edit it to include the desired host entries.

Flags

Item	Description
-v	Lists each host as it is added to the host file specified by the *HostFile* parameter.

Examples

Use the following command to generate the **/etc/hosts.pag** and **/etc/hosts.dir** files:

```
mkhosts /etc/hosts
```

This command creates two host files called **/etc/hosts.pag** and **/etc/hosts.dir**.

Files

Item	Description
hostfile.pag	One of two files containing the real database for name resolution.
hostfile.dir	One of two files containing the real database for name resolution.
hostfile.new.pag	One of two files containing the temporary database for name resolution.
hostfile.new.dir	One of two files containing the temporary database for name resolution.

mkiba Command

Purpose

Configures an IPv4 address or IPv6 address over the InfiniBand interface.

Syntax

mkiba { **-i** *Interface* **-a** *address* | **-v** *address6* **-A** *ib_adapter* **-p** *ib_port* [**-P** *P_KEY*] [**-V** *ipv6prefix*] [**-m** *subnet_mask*] [**-S** *state*] [**-M** *mtu*] [**-q** *queue_pair_size*] [**-Q** *Q_KEY*] [**-k** *superpacket*] }

Description

The **mkiba** command sets the minimal values required for using the IPv4 address or IPv6 address over the InfiniBand interface on a host machine. These values are written to the configuration database. This

command also parses the information and verifies if the parameters are correct. If the interface is not defined, this command defines it, and calls the **chdev** command to configure it.

The following list details the functions of the **mkiba** command:

- Defining the interface name in the configuration database if it is not already defined
- Setting the host name in both the configuration database and the running machine
- Setting the IP address of the interface in the configuration database
- Setting the subnetwork mask, if applicable
- Setting the prefix length, if applicable
- Setting InfiniBand-specific parameters (such as host channel adapter (HCA), port, and so on)

You can use the **smit mkinetib** fast path in the System Management Interface Tool (SMIT) to run this command.

At least one IPv4 address or IPv6 address must be included to configure the interface.

Use the **ifconfig** ibX to check the state of the interface after the configuration.

The **ifconfig** command indicates that if the InfiniBand multicast group is pending, then there is an adapter malfunction or the physical port is down. If there is an adapter malfunction, you must perform manual HCA device driver reconfiguration to recover the InfiniBand stack. If the port is down, you must check the cabling and switches. If multicast is pending, check the InfiniBand Subnet Manager for errors. To troubleshoot the InfiniBand interface issues, use the **ibstat** command.

Flags

Item	Description
-A *ib_adapter*	Specifies the HCA. For example, iba0.
-a *address*	Sets the Internet address of the host. Specify the address in dotted decimal notation. Each network interface on the host must have a unique Internet address. For example, a standard format for setting the Internet address is 127.10.31.2.
-i *Interface*	Specifies a particular InfiniBand interface. For example, ib0.
-k *superpacket*	Superpacket is a proprietary algorithm that allows the interface to receive large MTU (Maximum Transmit Unit) packets and fragment them in the interface layer to fit in the maximum MTU of the adapter. Enabling the superpacket automatically changes the *tcp_sendspace* attribute which specifies the number of bytes of data that the sending application can buffer in the kernel before the application is blocked on a send call, and the *tcp_recvspace* attribute which specifies the number of bytes of data that the receiving system can buffer in the kernel on the receiving sockets queue, with optimized values for the superpacket use.
-M *mtu*	Specifies the MTU for the interface. The IPv4 address or IPv6 address uses this MTU to fragment the packets.
-m *subnet_mask*	Specifies the mask that the gateway must use to determine the appropriate subnetwork for routing. The subnet mask is a set of 4 bytes. The subnet mask consists of high bits (1s) corresponding to the bit positions of the network and subnetwork address, and low bits (0s) corresponding to the bit positions of the host address.
-P *P_KEY*	Specifies the partition key. The common partition keys are 0xFFFF and 0x7FFF.
-p *ib_port*	Specifies the HCA ports that you must use to configure the InfiniBand interface.

Item	Description
-q *queue_pair_size*	Specifies the size of the software queue. The range is from 256 to 32000.
-Q *Q_KEY*	Specifies the multicast qkey used to create a broadcast multicast group if there is no group previously created in the Subnet Manager. The common keys are 1, 0, and 0x1E.
-S *state*	Specifies whether the interface is active. When an interface is marked as inactive, any attempt to transmit messages through that interface fails. This action does not automatically disable routes using the interface.
-V *ipv6prefix*	Specifies the number of high-order bits used by routing protocols. The prefix is usually denoted following the IPv6 address and a slash (/). For example, the notation ff12::/16 represents a 16-bit prefix with a value of 1111111100010010.
-v *address6*	Specifies the IPv6 address. The address is a 128-bit address represented as eight 16-bit integers separated by colons. Each integer is represented by 4 hex digits. Leading zeros can be skipped, and consecutive null 16-bit integers can be replaced by two colons (one per address). For example, fe80:abcd:0000:0000:0000:0000:0260:8c2e:00a4.

Example

1. To set the required values to configure the IPv4 address over the InfiniBand interface, enter:

```
mkiba -a 192.9.200.9 -i ib0 -A iba0 -p 1 -P -1 -q 4000 -M 2044 -m 255.255.255.0
```

2. To set the required values to configure the IPv6 address over the InfiniBand interface, enter:

```
mkiba -v fe80::2:c903:1:1b40 -i ib0 -A iba0 -p 1 -P -1 -q 4000 -M 2044
```

mkinstallp Command

Purpose

Creates software packages in **installp** format.

Syntax

mkinstallp [**-d** *BaseDirectory*] [**-T** *TemplateFile*]

Description

The **mkinstallp** command allows users to create their own software packages for AIX. Packages created with the **mkinstallp** command are in **installp** format and can be installed or removed with the **installp** command.

Files to be packaged by the **mkinstallp** command must be in a directory structure such that the location of the file relative to the root build directory is the same as the destination of the file after installation. For example, if **/usr/bin/***somecommand* is to be installed through a **mkinstallp** package, the *somecommand* parameter must be in the *buildroot***/usr/bin** directory when the **mkinstallp** command is run.

After the contents of a package are located in the correct directory structure, the **mkinstallp** command prompts for basic package data. This data includes the package name, requisites, descriptions of files to be packaged, and more. The **mkinstallp** command will then generate a template file based on responses

given by the user. To prevent command-line prompting, template files can be created and edited directly by the user and passed to the **mkinstallp** command with the **-T** flag.

Flags

Item	Description
-d *BaseDirectory*	Specifies the root build directory containing the files to be packaged. If not specified, the current working directory is used.
-T *TemplateFile*	Specifies the full path name of the template file to be passed to the **mkinstallp** command. If not specified, the **mkinstallp** command prompts for package information and creates a new template file based on user responses.

Note: Do not use the **.info** directory located in the *BaseDirectory* to store a template file. The template file may be removed when you run the **mkinstallp** command.

Examples

This example demonstrates how to package the file **/usr/bin/foo** using the **/tmp/packages** directory as the root build directory.

First, create the directory structure by typing the following at the command line:

```
mkdir -p /tmp/packages/usr/bin
```

Then, type the following to create the file **/usr/bin/foo**:

```
touch /tmp/packages/usr/bin/foo
```

Then, type the following to create the package using the **mkinstallp** command:

```
mkinstallp -d /tmp/packages
```

For more examples, see the **/usr/lpp/bos/README.MKINSTALLP** file.

Files

Item	Description
/usr/sbin/mkinstallp	Contains the **mkinstallp** command.

mkiscsi Command

Purpose

Adds iSCSI target data.

Syntax

`mkiscsi -l` *AdapterName* `-g static -t` *TargetName* `-n` *PortNumber* `-i` *IPaddress* [`-p` *password*] [`-u` *UserName*]

`mkiscsi -l` *AdapterName* `-g auto -t` *TargetName* `-p` *password* [`-u` *UserName*]

`mkiscsi -l` *AdapterName* `-g` *group* `-f` *FileName*

Description

The `mkiscsi` command adds iSCSI target data to ODM. There are two categories of data stored in ODM. The first is for statically configured iSCSI targets, which require that all the relevant iSCSI target information (such as target name, IP address, and port number) are specified in order for AIX to discover them. The second category of iSCSI target data is for iSCSI target devices that can be configured automatically, but require authentication from the host (such as passwords). These two categories of iSCSI target data are associated with the `static` and `auto` groups, respectively, specified by the `-g` flag.

Flags

Item	Description
`-f` *FileName*	Specifies the filename from which iSCSI target information will be read and then placed into ODM.
`-g` *group*	Specifies which group this iSCSI target is associated with. There two valid groups are `static` and `auto`. The `static` group is for iSCSI targets that cannot be automatically discovered from this host; all relevant iSCSI target information for them (such as target name, IP address, and port number) must be specified. The `auto` group is for iSCSI targets that are automatically discovered, but require authentication information such as passwords.
`-i` *IPaddress*	Specifies the IP address of the iSCSI target.
`-l` *AdapterName*	Specifies the adapter name for the iSCSI TCP/IP Offload Engine (TOE) adapter that is attached to this iSCSI target. It can also specify the iSCSI protocol device for the iSCSI software solution device.
`-n` *NewPortNumber*	Specifies the port number on which the iSCSI target is accessed. The default port number is `3260`.
`-p` *password*	Specifies the new password for this iSCSI target.
`-t` *TargetName*	Specifies the iSCSI target name (for example, `iqn.sn9216.iscsi-hw1`).
`-u` *UserName*	Specifies the Challenge Handshake Authentication Protocol (CHAP) user name that can be used during login. This value is used only if the CHAP password is supplied. If the CHAP user name is not specified, the iSCSI initiator uses the local iSCSI Qualified Name (as specified by the **`initiator_name`** attribute of the iSCSI device) for the CHAP user name.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

The `mkiscsi` command is executable only by root.

Examples

1. To add one statically configured iSCSI target, enter the following command:

```
mkiscsi -l ics0 -g static -t iqn.sn1234.iscsi_hw1
```

2. To add all the entries from the file `/etc/iscsi/targetshw`, enter the following command:

```
mkiscsi -l ics0 -g static -f /etc/iscsi/targetshw
```

Location

`/usr/sbin/mkiscsi`

Files

Item	Description
`src/bos/usr/sbin/iscsia`	Contains the common source files from which the iSCSI commands are built.

mkitab Command

Purpose

Makes records in the **/etc/inittab** file.

Syntax

mkitab [**-i** *Identifier*] { [*Identifier*] **:** [*RunLevel*] **:** [*Action*] **:** [*Command*] }

Description

The **mkitab** command adds a record to the **/etc/inittab** file. The *Identifier***:***RunLevel***:***Action***:***Command* parameter string specifies the new entry to the **/etc/inittab** file. You can insert a record after a specific record using the **-i** *Identifier* flag. The command finds the field specified by the *Identifier* parameter and inserts the new record after the one identified by the **-i** *Identifier* flag.

Parameters

The *Identifier***:***RunLevel***:***Action***:***Command* parameter string specifies the record in the **/etc/inittab** file, as follows:

Item	Description
Identifier	A 14-character parameter that uniquely identifies an object. The *Identifier* must be unique. If the *Identifier* is not unique, the command is unsuccessful. The *Identifier* cannot be changed; if you try to change it, the command is unsuccessful.
RunLevel	A 20-character parameter defining the run levels in which the *Identifier* can be processed. Each process started by the **init** command can be assigned one or more run levels in which it can be started.

Item	Description
Action	A 20-character parameter that informs the **init** command how to process the *Command* parameter that you specify. The **init** command recognizes the following actions: **respawn** If the process identified in this record does not exist, start the process. If the process currently exists, do nothing and continue scanning the **/etc/inittab** file. **wait** When the **init** command enters the run level specified for this record, start the process and wait for it to stop. While the **init** command is in the same run level, all subsequent reads of the **/etc/inittab** file ignore this object. **once** When the **init** command enters the run level specified for this record, start the process, do not wait for it to stop and when it does stop do not restart the process. If the system enters a new run level while the process is running, the process is not restarted. **boot** Read this record only when the system boots and reads the **/etc/inittab** file. The **init** command starts the process. Do not wait for the process to stop and when it does stop, do not restart the process. The run level for this process should be the default, or it must match the run level specified by the **init** command at startup time. **bootwait** Read this record only when the system boots and reads the **/etc/inittab** file. The **init** command starts the process. Wait for it to stop, and when it does stop, do not restart the process. **powerfail** Start the process identified in this record only when the **init** command receives a **SIGPWR** power fail signal. **powerwait** Start the process identified in this record only when the **init** command receives a **SIGPWR** power fail signal, and wait until it stops before continuing to process the **/etc/inittab** file. **off** If the process identified in this record is currently running, send the warning signal **SIGTERM** and wait 20 seconds before sending the **SIGKILL** kill signal. If the process is nonexistent, ignore this line. **hold** When the process identified in this record is terminated, do not start a new one. The **hold** action can only be activated by the **phold** command. **ondemand** Functionally identical to **respawn**. If the process identified in this record does not exist, start the process. If the process currently exists, do nothing and continue scanning the **/etc/inittab** file. Specify this action to perform the **respawn** action when using **a**, **b**, or **c** run levels.

Item	Description
	initdefault A line with this action is processed only when the **init** command is originally invoked. The **init** command uses this line to determine which run level to originally enter. The command does this by taking the highest run level specified in the *RunLevel* parameter and using that as the command's initial state. If the *RunLevel* parameter is empty, its value is interpreted as 0123456789, and the **init** command enters a run level of **9**. If the **init** command does not find an **initdefault** line in the **inittab** file, it requests an initial run level from the operator at initial program load (IPL) time. **sysinit** Start the process identified in this record before the **init** command tries to access the console. For example, you might use this to initialize devices.
Command	A 1024-character field specifying the shell command.  **Attention:** To avoid possible corruption of system files, the **stdin**, **stdout**, and **stderr** files must be specified in the *Command* parameter with redirection, or they must be explicitly opened by the program being run by the command line.

Flags

Item	Description
-i *Identifier*	Specifies which record in the **/etc/inittab** file the new record follows.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To add a new record to the **/etc/inittab** file, telling the **init** command to handle a login on tty2, type:

```
mkitab "tty002:2:respawn:/usr/sbin/getty /dev/tty2"
```

2. To add a new record to the **/etc/inittab** file, telling the **init** command to execute the **/etc/rc.tcpip** file after the **/usr/sbin/srcmstr** file is started, type:

```
mkitab -i srcmstr "rctcpip:2:wait:/etc/rc.tcpip > /dev/console"
```

3. To add a new record to the **/etc/inittab** file, telling the **init** command to execute the **/etc/rc** file and send its output to the boot log, type:

```
mkitab ((rc:2:wait:/etc/rc 2>&1 | alog -tboot > /dev/console))
```

Files

Item	Description
/etc/inittab	Contains the **mkitab** command.

mkkeyserv Command

Purpose

Uncomments the entry in the **/etc/rc.nfs** file for the **keyserv** daemon and invokes the daemon by using the **startsrc** command.

Syntax

/usr/sbin/mkkeyserv [**-I** | **-B** | **-N**]

Description

The **mkkeyserv** command uncomments the entry in the **/etc/rc.nfs** file for the **keyserv** daemon. The **mkkeyserv** command starts the daemon by using the **startsrc** command.

You can use the System Management Interface Tool (SMIT) **smit mkkeyserv** fast path to run this command.

Flags

Item	Description
-I	Uncomments the entry in the **/etc/rc.nfs** file to start the **keyserv** daemon on the next system restart.
-B	Uncomments the entry in the **/etc/rc.nfs** file to start the **keyserv** daemon and uses the **startsrc** command to start the **keyserv** daemon. This flag is the default.
-N	Uses the **startsrc** command to start the **keyserv** daemon. This flag does not change the **/etc/rc.nfs** file.

Examples

To modify the **/etc/rc.nfs** file to invoke the **keyserv** daemon on the next system restart, enter:

```
mkkeyserv -I
```

Files

Item	Description
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.

mkkrb5clnt Command

Purpose

Configures a Kerberos client.

Syntax

To configure Kerberos against IBM Network Authentication Service only:

mkkrb5clnt -h | [**-c** *KDC* **-r** *Realm* **-s** *Server* **-U** [**-a** *Admin*] **-d** *Domain* [**-A**] [**-i** *Database*] [**-K**] [**-T**] [**-t** *ticket_lifetime*] [**-n** *renew_lifetime*]] [**-l** {*ldapserver* | *ldapserver:port*}]

To configure Kerberos against non-kadmind services:

mkkrb5clnt -h | **-c** *KDC* **-r** *Realm* **-s** *Server* **-d** *Domain* [**-i** *Database*] [**-K**] [**-t** *ticket_lifetime*] [**-n** *renew_lifetime*] **–D** [**-l** {*ldapserver* | *ldapserver:port*}] | **-U**

Description

This command configures the Kerberos client. The first part of the command reads realm name, KDC, VDB path, and domain name from the input and generates a **krb5.conf** file.

Item	Description
/etc/krb5/krb5.conf:	Values for realm name, Kerberos admin server, and domain name are set as specified on the command line. Also updates the paths for **default_keytab_name**, **kdc**, and **kadmin** log files.

If DCE is not configured, this command creates a link to **/etc/krb5/krb5.conf** from **/etc/krb5.conf**.

The command also allows you to configure root as admin user, configure integrated Kerberos authentication, and configure Kerberos as default authentication scheme.

For integrated login, the **-i** flag requires the name of the database being used. For LDAP, use the load module name that specifies LDAP. For local files, use the keyword files.

Item	Description
Standard Output	Consists of information messages when the **-h** flag is used.
Standard Error	Consists of error messages when the command cannot complete successfully.

Flags

Item	Description
-a *Admin*	Specifies the principal name of the Kerberos server admin.
-A	Specifies root to be added as a Kerberos administrative user.
-c *KDC*	Specifies the KDC server.
-d *Domain*	Specifies the complete domain name for the Kerberos client.
-D	Specifies Kerberos against non-kadmind services.
-h	Specifies that the command is only to display the valid command syntax.
-i *Database*	Configures integrated Kerberos authentication.
-K	Specifies Kerberos to be configured as the default authentication scheme.
-l *ldapserver* | *ldapserver:port*	For servers, specifies the LDAP directory used to store the Network Authentication Service principal and policy information. For clients, specifies the LDAP directory server to use for Administration server and KDC discovery using LDAP. If the **-l** flag is used, then the KDC and server flags are optional. If the **-l** option is not used, the KDC and server flags must be specified. The port number can optionally be specified. For clients and servers, the port number can optionally be specified. If the port number is not specified, the client connects to the default LDAP server port 389 or 636 for SSL connections. **Note:** Only the client configuration is updated.

Item	Description
-n *renew_lifetime*	Specifies the client-specific time to generate a renewable ticket if the server supports it. By default, the ticket is nonrenewable. The *renew_lifetime* parameter value is composed of four numeric values that are delimited by colons.
-r *Realm*	Specifies the full realm name for which the Kerberos client is to be configured.
-s *Server*	Specifies the fully qualified host name for Kerberos admin server.
-t *ticket_lifetime*	Specifies the client-specific ticket lifetime for received tickets if the server supports it. If you do not specify the flag, the server sets the ticket lifetime. The *ticket_lifetime* parameter value is composed of four numeric values that are delimited by colons.
-T	Specifies the flag to acquire server admin TGT based admin ticket.
-U	Undo the setup from the previous configuration command.

Exit Status

Failure of this command to execute successfully may result in incomplete client configuration.

Item	Description
0	Indicates the successful completion of the command.
1	Indicates that an error occurred.

Security

A user with the **aix.security.kerberos** authorization is authorized to use this command.

Examples

1. To display the command syntax, enter the following command:

```
mkkrb5clnt -h
```

2. To configure **testbox.austin.ibm.com** as a client to **sundial.austin.ibm.com** where KDC is also running on **sundial.austin.ibm.com**, enter the following command:

```
mkkrb5clnt -c sundial.austin.ibm.com -r UD3A.AUSTIN.IBM.COM \
-s sundial.austin.ibm.com -d austin.ibm.com
```

3. To configure **testbox.austin.ibm.com** as the client, make root as the server admin, configure integrated login, configure Kerberos as default authentication scheme, enter the following command:

```
mkkrb5clnt -c sundial.austin.ibm.com -r UD3A.AUSTIN.IBM.COM \
-s sundial.austin.ibm.com -d austin.ibm.com \
-A -i files -K -T
```

4. To configure **testbox.austin.ibm.com** as the client against a non-AIX machine, enter the following command:

```
mkkrb5clnt -c non-aix.austin.ibm.com -r NON-AIX.AUSTIN.IBM.COM \
-s non-aix.austin.ibm.com -d austin.ibm.com –D
```

5. To configure **testbox.austin.ibm.com** as the client against a non-AIX machine with the ticket lifetime of 1 day, 2 hours, 3 minutes, and 4 seconds, and the renew lifetime of 5 days, 6 hours, 7 minutes, and 8 seconds, enter the following command:

```
mkkrb5clnt -c non-aix.austin.ibm.com -r NON-AIX.AUSTIN.IBM.COM \
-s non-aix.austin.ibm.com -d austin.ibm.com –D \
-t 1:2:3:4 -n 5:6:7:8
```

Files

Item	Description
/usr/krb5/sbin	Contains the **mkkrb5clnt** command.

mkkrb5srv Command

Purpose

Configures a Kerberos server.

Syntax

mkkrb5srv **-h** | [**-r** *Realm* **-d** *Domain* **-a** *AdminName*] [**-l** *ldapserver* | *ldapserver:port*] [**-u** *ldap_DN*] [**-p** *ldap_DN_pw*] [**-f** {*keyring* | *keyring:entry_dn*}] [**-k** *keyring_pw*] [**-b** *bind_type*] [**-m** *masterkey_location*] [**-U**]

Description

The **mkkrb5srv** command configures the Kerberos server. This command creates the **kadm5.acl** file, the **kdc.conf** file, and the Kerberos database. It also adds the administrator to the database and updates the **/etc/inittab** file with Kerberos daemons. This command does the initial configuration once the variables are set. They can be modified by editing the following files:

Item	Description
/etc/krb5/krb5.conf:	Values for realm name, Kerberos admin server, and domain name are set as specified on the command line. Also updates the paths for **default_keytab_name**, **kdc**, and **kadmin** log files.
/var/krb5/krb5kdc/kdc.conf	This command sets the value for **kdc_ports**. Paths for database name, **admin_keytab**, **acl_file**, **dict_file**, **key_stash_file**. Values for **kadmin_port**, **max_life**, **max_renewable_life**, **master_key_type**, and **supported_enctypes**.
/var/krb5/krb5kdc/kadm5.acl	Sets up the acls for admin, root, and host principals.

If DCE is not configured, this command creates a link to **/etc/krb5/krb5.conf** from **/etc/krb5.conf**.

Item	Description
Standard Output	Consists of information messages when the **-h** flag is used.
Standard Error	Consists of error messages when the command cannot complete successfully.

Flags

Item	Description
-a *AdminName*	Specifies the Kerberos Principal name for the administrator.
-b *bind_type*	Specifies the LDAP bind type. Supported values are the following: • simple • cram-md5 • external These bind types can be specified in either upper case or lower case.

Item	Description
-d *Domain*	Specifies the domain name for the Kerberos realm.
-f {*keyring* | *keyring:entry_dn*}	Specifies the LDAP keyring database file name if you are using SSL communication.
-h	Specifies that the command is only to display the valid command syntax.
-k*keyring_pw*	Specifies the password for the LDAP keyring database file. If not specified, SSL uses the password that is encrypted in the appropriate password stash file.
-l *ldapserver* | *ldapserver:port*	For servers, specifies the LDAP directory used to store the Network Authentication Service principal and policy information. For clients, specifies the LDAP directory server to use for Administration server and KDC discovery using LDAP. If the **-l** flag is used, then the KDC and server flags are optional. If the **-l** option is not used, the KDC and server flags must be specified. The port number can optionally be specified. For clients and servers, the port number can optionally be specified. If the port number is not specified, the client connects to the default LDAP server port 389 or 636 for SSL connections. **Note:** Only the client configuration is updated.
-m *masterkey_location*	Specifies the fully qualified file name for storing the master key in the local file system when using LDAP to store data. **Note:** This flag is only for use with the LDAP directory.
-p *ldap_DN_pw*	Specifies the password for the entry being used for the *ldap_DN_pw*.
-r *Realm*	Specifies the realm for which the Kerberos server is to be configured.
-u *ldap_DN*	Specifies the LDAP entry to be used as the *ldap_DN*. **Note:** With external bind, the **-u** and **-p** flags are not required, and the values come form the certificate.
-U	Undo the setup from the previous configuration command.

Exit Status

Failure of this command to execute successfully results in incomplete server configuration.

Item	Description
0	Indicates the successful completion of the command.
1	Indicates that an error occurred.

Security

A user with the **aix.security.kerberos** authorization is authorized to use this command.

Examples

1. To display the command syntax, type:

```
mkkrb5srv -h
```

2. To configure sundial as a Kerberos server, type:

```
mkkrb5srv -r UD3A.AUSTIN.IBM.COM -d austin.ibm.com
```

Files

Item	Description
/usr/sbin/mkkrb5srv	Contains the **mkkrb5srv** command.

mklost+found Command

Purpose

Creates a lost and found directory for the **fsck** command.

Syntax

mklost+found

Description

The **mklost+found** command creates a lost and found directory in the current directory. A number of empty files are created within the lost and found directory and then removed so that there are empty slots for the **fsck** command. The **fsck** command reconnects any orphaned files and directories by placing them in the lost and found directory with an assigned i-node number. The **mklost+found** command is not normally needed, since the **fsck** command automatically creates the lost and found directory when a new file system is created.

Examples

To make a lost+found directory for the **fsck** command, enter:

```
mklost+found
```

Files

Item	Description
/usr/sbin/mklost+found	Contains the **mklost+found** command.

mklpcmd Command

Purpose

Defines a new least-privilege (LP) resource to the resource monitoring and control (RMC) subsystem and specifies user permissions.

Syntax

`mklpcmd` [`-n` *host*] [`-l`] [`-c 0 | 1 | 2 | 3`] [–R *RunCmdName*] [`-s` *FilterScript*] [`-A` *FilterArg*] [`-h`] [`-TV`] *resource_name command_path* [*ID perm*] ...

Description

The mklpcmd command defines a new LP resource to the resource monitoring and control (RMC) subsystem subsystem. An LP resource is a root command or script to which users are granted access based on permissions in the LP access control lists (ACLs). Specify the LP resource using the *resource_name* parameter. The *command_path* parameter specifies the command or script that could be run with LP access. Specify the complete path name of the command or the script. If *command_path* exists when a resource is created, the LP resource manager calculates the CheckSum and assigns the CheckSum attribute value. If *command_path* does not exist, the LP resource manager assigns 0 as the CheckSum attribute value.

Use the -l flag to lock the LP resource. The resource must be unlocked before it can be deleted. Use the -c flag to specify the control settings of the resource.

You can also use the mklpcmd command to specify permissions for users when you are creating a resource. To do this, you need to have administrator permission on the resources. Administrator permission gives you the ability to set and edit permissions. You can specify multiple user IDs and permissions with this command. See the Examples section for more information.

This command runs on any node. In a management domain or a peer domain, use the -n flag to define the LP resource on the node that is specified by *host*. Otherwise, this command runs on the local node.

Flags

-n *host*

Specifies the node in the domain on which the LP resource is to be defined. By default, the LP resource is defined on the local node. The –n flag is valid only in a management or peer domain. If the CT_MANAGEMENT_SCOPE variable is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The mklpcmd command runs once for the first valid scope that the LP resource manager finds.

-l

Defines the new LP resource as locked so that it cannot be changed accidentally. The resource cannot be removed from the RMC subsystem until the Lock attribute is unset.

If you do not specify this flag, the new resource is not locked. This is the default.

-c 0 | **1** | **2** | **3**

Sets the ControlFlags attribute, which is used to specify the control features for an LP command. If ControlFlags is not specified, it is set to 1 by default. Use this flag to specify one of these values:

0

Does not validate the CheckSum value.

1

Does not validate the CheckSum value. This is the default.

2

Validates the CheckSum value.

3

Validates the CheckSum value.

When an attempt is made to run the LP resource using the runlpcmd command, the value of the ControlFlags attribute determines which checks are performed before running the command represented by the resource.

In this release of RSCT, the ControlFlags attribute value specifies whether the CheckSum value is to be validated.

In previous releases of RSCT, the `ControlFlags` attribute value also specified whether the presence of certain characters in the input arguments to `runlpcmd` were to be disallowed. Checking for these characters is no longer necessary.

To maintain compatibility with LP resources that were defined in previous releases of RSCT, the `ControlFlags` attribute values, with respect to validating the `CheckSum` value, have remained the same. Consequently, values `0` and `1` indicate that the `CheckSum` value is not to be validated, and values 2 and 3 indicate that the `CheckSum` value is to be validated.

-R *RunCmdName*
Specifies the `RunCmdName` value for this resource, which will be used as a parameter of the `runlpcmd` command.

-s *script_path*
Specifies the fully-qualified path of the filter script.

-A *argument*
Specifies a string of arguments to be passed to the filter script.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error.

-V
Writes the command's verbose messages to standard output.

Parameters

resource_name
Is the name or identifier of the LP resource that is to be defined to the RMC subsystem.

command_path
Is the complete, fully-qualified path name of the command or script.

ID perm ...
Specifies permissions for users when you are creating a resource. This parameter is optional.

ID
Specifies the user identity for the ACL entry. See the `User identities` section of the `lpacl` information for the valid forms of this parameter.

perm
Specifies the user permissions for the ACL entry. This parameter can consist of a combination of any of the following values:

r
Read permission (consists of the q, `l`, e, and v permissions)

w
Write permission (consists of the d, c, `s`, and o permissions)

a
Administrator permission

x
Execute permission

q
Query permission

l
Enumerate permission

e
Event permission

v
Validate permission

d
Define and undefine permission

c
Refresh permission

s
Set permission

o
Online, offline, and reset permission

0
No permission

See the `User permissions` section of the `lpacl` information for descriptions of these permissions.

Security

- To run the `mklpcmd` command with one or more *ID*`:`*perm* parameters, you need:
  - read and write permission in the Class ACL of the `IBM.LPCommands` resource class.
  - read and administrator permission in the Resource Initial ACL.

    As an alternative, the Resource Initial ACL can direct the use of the Resource Shared ACL if these permissions exist in the Resource Shared ACL.
- To run the `mklpcmd` command with no *ID*`:`*perm* parameters, you need write permission in the Class ACL of the `IBM.LPCommands` resource class.

Permissions are specified in the LP ACLs on the contacted system. See the `lpacl` file for general information about LP ACLs and the *RSCT Administration Guide* for information about modifying them.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
The resource was not found.

Environment Variables

CT_CONTACT
Determines the system that is used for the session with the RMC daemon. When `CT_CONTACT` is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If `CT_CONTACT` is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the LP resources that are processed.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to process the LP resource. The management scope determines the set of possible target nodes where the resource can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. When the `-V` flag is specified, this command's verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To create an LP resource called `LP1` that points to a command called `/tmp/user1/lpcmd1` on the local node, enter:

```
mklpcmd LP1 /tmp/user1/lpcmd1
```

2. To create an LP resource called `LP2` that points to a command called `/tmp/my_command1` on `nodeB` in the management domain, enter:

```
mklpcmd -n nodeB LP2 /tmp/my_command1
```

3. To create an LP resource called `lp3` with `ControlFlags` set to `3` (which means verify the `CheckSum` value), enter:

```
mklpcmd -c 3 LP3 /tmp/cmd_lp3
```

4. To create an LP resource called `lp4` that points to `/tmp/testscript`, has a `RunCmdName` value of `test`, a `FilterScript` value of `/tmp/filterscr`, and filter arguments `node1` and `node2`, enter:

```
mklpcmd -R test -f /tmp/filterscr -A "node1,node2" lp4 /tmp/testscript
```

5. To create an LP resource called `lp5` that points to `/usr/bin/mkrsrc` and gives users `user1@LOCALHOST` and `user2@LOCALHOST` read, write, and execute permission, enter:

```
mklpcmd lp5 /usr/bin/mkrsrc user1@LOCALHOST rwx  user2@LOCALHOST rwx
```

Location

/opt/rsct/bin/mklpcmd
Contains the `mklpcmd` command

mklv Command

Purpose

Creates a logical volume.

Syntax

mklv [**-a** *position*] [**-b** *badblocks*] [**-c** *copies*] [**-C** *stripewidth*] [**-d** *schedule*] [**-R** *PreferredRead*] [**-e** *range*] [**-i**] [**-L** *label*] [**-m** *mapfile*] [**-o** *y / n*] [**-r** *relocate*] [**-s** *strict*] [**-t** *type*] [**-T O**] [**-u** *upperbound*] [**-v** *verify*] [**-w** *mirrorwriteconsistency*] [**-x** *maximum*] [**-y** *newlogicalvolume* | **-Y** *prefix*] [**-S** *stripsize*] [**-U** *userid*] [**-G** *groupid*] [**-P** *modes*] [**-p** *copyn*=*mirrorpool*] [**-O** y | n] [**-k** y | n] *volumegroup number* [*physicalvolume ...*]

Description

The **mklv** command creates a new logical volume within the *volumegroup*. For example, all file systems must be on separate logical volumes. The **mklv** command allocates the number of logical partitions to the new logical volume. If you specify one or more physical volumes with the *physicalvolume* parameter, only those physical volumes are available for allocating physical partitions; otherwise, all the physical volumes within the volume group are available.

The default settings provide the most commonly used characteristics, but use flags to tailor the logical volume to the requirements of your system. After a logical volume is created, its characteristics can be changed with the **chlv** command.

The default allocation policy is to use a minimum number of physical volumes per logical volume copy, to place the physical partitions belonging to a copy as contiguously as possible, and then to place the physical partitions in the desired region specified by the **-a** flag. Also, by default, each copy of a logical partition is placed on a separate physical volume.

The **-m** flag specifies exact physical partitions to be used when creating the logical volume.

The `-U`, `-G`, and `-P` flags can be used to set the ownership, group, and permissions, respectively, of the logical volume device special files. Only root users can set these values. For scalable and big vg format volume groups that are exported, specify the `-R` flag with the `importvg` command to restore these values upon import.

You can specify logical volumes sizes in 512 Blocks/KB/MB/GB when using the **mklv** command. The logical volumes sizes must be integer values.

Physical partitions are numbered starting at the outermost edge with number one.

Note:

1. Changes made to the logical volume are not reflected in the file systems. To change file system characteristics use the **chfs** command.
2. Each logical volume has a control block. This logical volume control block is the first few hundred bytes within the logical volume. Care has to be taken when reading and writing directly to the logical volume to allow for the control block. Logical volume data begins on the second 512-byte block.
3. To use this command, you must either have root user authority or be a member of the **system** group.
4. When creating a striped logical volume using the **-S** flag, you must specify two or more physical volumes or use the **-C** or **-u** flag.
5. When creating a striped logical volume, the number of partitions must be an even multiple of the striping width. If not, the number of partitions will be rounded up to the next valid value.
6. The **mklv** command is not allowed on a snapshot volume group.
7. Mirror Write Consistency (MWC) and Bad Block Relocation (BBR) are not supported in a concurrent setup with multiple active nodes accessing a disk at the same time. These two options must be disabled in this type of concurrent setup.

8. Bad block relocation policy of a logical volume is not supported on a volume group that is created with 4 KB block physical volumes.

You can use the System Management Interface Tool (SMIT) **smit mklv** fast path to run this command.

File Systems on Striped Logical Volumes

If you want to create a file system on a striped logical volume, you should create the striped logical volume before you run the **crfs** command or **mkfs** command to create the file system. In order to maximize the use of disk space within the striping width, you should choose hard disks of the same size when creating the striped logical volume. The striping width is the number of hard disks that form the striped logical volume.

Flags

Item	Description
-a *position*	Sets the intra-physical volume allocation policy (the position of the logical partitions on the physical volume). The *position* variable can be one of the following: **m** Allocates logical partitions in the outer middle section of each physical volume. This is the default position. **c** Allocates logical partitions in the center section of each physical volume. **e** Allocates logical partitions in the outer edge section of each physical volume. **ie** Allocates logical partitions in the inner edge section of each physical volume. **im** Allocates logical partitions in the inner middle section of each physical volume.
-b *badblocks*	Sets the bad-block relocation policy. The *Relocation* variable can be one of the following: **y** Causes bad-block relocation to occur. This is the default. **n** Prevents bad-block relocation from occurring.
-c *copies*	Sets the number of physical partitions allocated for each logical partition. The *copies* variable can be set to a value from 1 to 3; the default is 1.
-C *stripewidth*	Sets the Stripe width of the logical volume. If the *Stripewidth* is not entered it is assumed to be the *upperbound* or the total number of disks specified on the command line.

Item	Description
-d *schedule*	Sets the scheduling policy when more than one logical partition is written. The *schedule* variable can be one of the following: **p** Establishes a parallel scheduling policy. This is the default for scheduling policy. **ps** Parallel write with sequential read policy. All mirrors are written in parallel but always read from the first mirror if the first mirror is available. **pr** Parallel write round robin read. This policy is similar to the parallel policy except an attempt is made to spread the reads to the logical volume more evenly across all mirrors. **s** Establishes a sequential scheduling policy. **Note:** The **-R** flag overwrites the read policy specified by the **-d** flag. If the preferred copy is not available, the read operations follow the scheduling policy.
-R *PreferredRead*	Sets read preference to the copy of the logical volume. If the **-R** flag is specified and if the preferred copy is available, the read operation occurs from the preferred copy. If the preferred copy is not available, the read operations follow the scheduling policy of the logical volume. The *PreferredRead* variable can be set to a value in the range 0 -3. The default value is 0.
-e *range*	Sets the inter-physical volume allocation policy (the number of physical volumes to extend across, using the volumes that provide the best allocation). The *Range* value is limited by the *upperbound* variable, (set with the **-u** flag) and can be one of the following: **x** Allocates across the maximum number of physical volumes. **m** Allocates logical partitions across the minimum number of physical volumes. This is the default range.
-G *groupid*	Specifies group ID for the logical volume special file.
-i	Reads the *physicalvolume* parameter from standard input. Use the **-i** flag only when *physicalvolume* is entered through standard input.

Item	Description
-k y | n	Enables the data encryption option in the logical volume. The **-k** flag is available in IBM AIX 7.2 with Technology Level 5, or later. You can specify the following values for this flag: **y** The data encryption option of the logical volume is enabled. The primary key of the logical volume must be initialized to access the logical volume. Use the **hdcryptmgr authinit** command to initialize the primary key of the logical volume. **n** The data encryption option of the logical volume is not enabled. This is the default value. **Note:** • The data encryption option must be enabled at the volume group level before you can enable the data encryption option for a logical volume. • The **-k** flag cannot be used if the volume group is varied on in the concurrent mode. • The **-k** flag is not supported on `boot`, `dump`, `paging`, and `aio_cache` logical volume type.
-L	Sets the logical volume label. The default label is **None**. The maximum size of the label file is 127 characters. **Note:** If the logical volume is going to be used as a journaled file system (JFS), then the JFS will use this field to store the mount point of the file system on that logical volume for future reference.
-m *mapfile*	Specifies the exact physical partitions to allocate. Partitions are used in the order given by the file designated by the *mapfile* parameter. All physical partitions belonging to a copy are allocated before allocating for the next copy. The *mapfile* format is: **PVname:PPnum1[-PPnum2]** where *pvname* is a physical volume name (for example, `hdisk0`). It is one record per physical partition or a range of consecutive physical partitions. **PVname** Name of the physical volume as specified by the system. **PPnum** Physical partition number. **Important:** When you use map files, you must understand and adhere to all LV-allocation parameters such as strictness, upperbound, and stripe width. Using map files bypasses the checks done in the LVM-allocation routines. This is important for striped LVs, which are assumed to have a typical striped allocation pattern conforming to the stripe width.

Item	Description
-o *y / n*	Turns on/off serialization of overlapping I/Os. If serialization is turned on then overlapping I/Os are not allowed on a block range and only a single I/O in a block range is processed at any one time. Most applications like file systems and databases do serialization so serialization should be turned `off`. The default for new logical volumes is `off`.
-O *y / n*	Enables the infinite retry option of the logical volume. **n** The infinite retry option of the logical volume is not enabled. The failing I/O of the logical volume is not retried. This is the default value. **y** The infinite retry option of the logical volume is enabled. The failed I/O request is retried until it is successful. **Note:** The infinite retry option is ignored for a logical volume (LV) when *active* mirror write consistency is set. The infinite retry option must be enabled at the volume group level to work for a logical volume when *active* mirror write consistency is set. **Note:** The infinite retry option is not supported in the Geographic Logical Volume Manager (GLVM) environment.
-p *copyn***=***mirrorpool*	Enables mirror pools for the logical volume. A mirror pool is assigned to a copy using the *copyn***=***mirrorpool* parameter. Specify a mirror pool for each copy. To specify more than one *copyn***=***mirrorpool* pair, provide multiple **-p** *copyn***=***mirrorpool* flags. Mirror pool names can be up to 15 characters and follow the same rules that apply to volume group names and logical volume names.
-P *modes*	Specifies permissions (file modes) for the logical volume special file.
-r *relocate*	Sets the reorganization relocation flag. For striped logical volumes, the *relocate* parameter must be set to **n** (the default for striped logical volumes). The *relocate* parameter can be one of the following: **y** Allows the logical volume to be relocated during reorganization. This is the default for relocation. **n** Prevents the logical volume from being relocated during reorganization.

Item	Description
-s *strict*	Determines the strict allocation policy. Copies of a logical partition can be allocated to share or not to share the same physical volume. The *strict* parameter is represented by one of the following: **y** Sets a strict allocation policy, so copies for a logical partition cannot share the same physical volume. This is the default for allocation policy. **n** Does not set a strict allocation policy, so copies for a logical partition can share the same physical volume. **s** Sets a super strict allocation policy, so that the partitions allocated for one mirror cannot share a physical volume with the partitions from another mirror.
-S *stripSize*	Specifies the number of bytes per strip (the strip size multiplied by the number of disks in an array equals the stripe size). Valid values include 4K, 8K, 16K, 32K, 64K, 128K, 256K, 512K, 1M, 2M, 4M, 8M, 16M, 32M, 64M, and 128M. **Note:** The **-d**, **-e**, and **-s** flags are not valid when creating a striped logical volume using the **-S** flag.
-t *type*	Sets the logical volume type. The standard types are **jfs** (journaled file systems), **jfslog** (journaled file system logs), **jfs2** (enhanced journaled file system), **jfs2log** (enhanced journaled file system logs), and **paging** (paging spaces), but a user can define other logical volume types with this flag. You cannot create a striped logical volume of type **boot**. The default is **jfs**. If a log is manually created for a file system, the user must run the **logform** command to clean out the new **jfslog** before the log can be used. For example, to format the logical volume `logdev`, type: `logform /dev/logdev` where `/dev/logdev` is the absolute path to the logical volume.
-T O	The **-T O** option indicates that the logical volume control block does not occupy the first block of the logical volume. Therefore, the space is available for application data. Applications can identify this type of logical volume with the **IOCINFO** `ioctl` operation. The logical volume has a device subtype of `DS_LVZ`. A logical volume created without this option has a device subtype of `DS_LV`.
-U *userid*	Specifies user ID for logical volume special file.

Item	Description
-u *upperbound*	Sets the maximum number of physical volumes for new allocation. The value of the *upperbound* variable should be between one and the total number of physical volumes. When using super strictness, the upper bound indicates the maximum number of physical volumes allowed for each mirror copy. When using striped logical volumes, the upper bound must be multiple of *stripewidth*. If *upperbound* is not specified it is assumed to be *stripewidth* for striped logical volumes.
-v *verify*	Sets the write-verify state for the logical volume. Causes (**y**) all writes to the logical volume to either be verified with a follow-up read, or prevents (**n**) the verification of all writes to the logical volume. The *verify* parameter is represented by one of the following: **n** Prevents the verification of all write operations to the logical volume. This is the default for the **-v** flag. **y** Causes the verification of all write operations to the logical volume.
-w *mirrorwriteconsistency*	**y or a** Turns on *active* mirror write consistency that ensures data consistency among mirrored copies of a logical volume during typical I/O processing. **p** Turns on *passive* mirror write consistency that ensures data consistency among mirrored copies during volume group synchronization after a system interruption. **Note:** This function is available only on **big** type and **scalable** type of volume groups. **n** No mirror write consistency. See the **-f** flag of the **syncvg** command.
-x *maximum*	Sets the maximum number of logical partitions that can be allocated to the logical volume. The default value is 512. The number represented by the *number* parameter must be equal to or less than the number represented by the *maximum* variable.

Item	Description
-y *newlogicalvolume*	Specifies the logical volume name rather than having the name generated automatically. Logical volume names must be unique system wide and can range from 1 to 15 characters. If the *volumegroup* is varied on in concurrent mode, the new logical volume name should be unique across all the concurrent nodes where the *volumegroup* is varied on. The name cannot begin with a prefix already defined in the **PdDv** class in the Device Configuration Database for other devices. The logical volume name created is sent to standard output. The logical volume name can only contain the following characters: • "A" through "Z" • "a" through "z" • "0" through "9" • "_" (the underscore) • "-" (the minus sign) • "." (the period) All other characters are considered not valid.
-Y *prefix*	Specifies the *prefix* to use instead of the prefix in a system-generated name for the new logical volume. The prefix must be less than or equal to 13 characters. The name cannot begin with a prefix already defined in the **PdDv** class in the Device Configuration Database for other devices, nor be a name already used by another device.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To make a logical volume in volume group `vg02` with one logical partition and a total of two copies of the data, type:

   ```
   mklv -c 2 vg02 1
   ```

2. To make a logical volume in volume group `vg03` with nine logical partitions and a total of three copies spread across a maximum of two physical volumes, and whose allocation policy is not strict, type:

   ```
   mklv -c 3  -u 2  -s n vg03 9
   ```

3. To make a logical volume in `vg04` with five logical partitions allocated across the center sections of the physical volumes when possible, with no bad-block relocation, and whose type is paging, type:

   ```
   mklv -a c -t paging -b n vg04 5
   ```

4. To make a logical volume in `vg03` with 15 logical partitions chosen from physical volumes `hdisk5`, `hdisk6`, and `hdisk9`, type:

```
mklv vg03 15 hdisk5 hdisk6 hdisk9
```

5. To make a striped logical volume in vg05 with a strip size of 64K across 3 physical volumes and 12 logical partitions, type:

```
mklv -u 3 -S 64K vg05 12
```

6. To make a striped logical volume in vg05 with a strip size of 8K across hdisk1, hdisk2, and hdisk3 and 12 logical partitions, type:

```
mklv -S 8K vg05 12 hdisk1 hdisk2 hdisk3
```

7. To request a logical volume with a minimum size of 10MB, type:

```
mklv VGNAME 10M #
```

   The **mklv** command will determine the number of partitions needed to create a logical volume of at least that size.

   You can use uppercase and lowercase letters as follows:

```
    B/b       512 byte blocks
    K/k       KB
    M/m       MB
    G/g       GB
```

8. To create three copies of logical volume in volume group testvg and to set the read preference to the second logical volume copy, enter the following command:

```
mklv -c 3 -R 2 -y testlv testvg 10
```

Files

Item	Description
/usr/sbin	Directory where the **mklv** command resides.
/tmp	Directory where the temporary files are stored while the command is running.
/dev	Directory where the character and block device entries for the logical volume are created.

mklvcopy Command

Purpose

Provides copies of data within the logical volume.

Syntax

mklvcopy [**-a** *position*] [**-e** *range*] [**-k**] [**-m** *mapfile*] [**-s** *strict*] [**-u** *upperbound*] [**-p** *copyn*=*mirrorpool*] *logicalvolume copies* [*physicalvolume*...]

Description

The **mklvcopy** command increases the number of copies in each logical partition in *logicalvolume.* This is accomplished by increasing the total number of physical partitions for each logical partition to the number represented by *Copies*. The *logicalvolume* parameter can be a logical volume name or logical volume ID. You can request that the physical partitions for the new copies be allocated on specific physical volumes

(within the volume group) with the *physicalvolume* parameter; otherwise, all the physical volumes within the volume group are available for allocation.

The logical volume modified with this command uses the *copies* parameter as its new **copy** characteristic. The data in the new copies are not synchronized until one of the following occurs: the **-k** option is used, the volume group is activated by the **varyonvg** command, or the volume group or logical volume is synchronized explicitly by the **syncvg** command. Individual logical partitions are always updated as they are written to.

The default allocation policy is to use minimum numbering of physical volumes per logical volume copy, to place the physical partitions belong to a copy as contiguously as possible, and then to place the physical partitions in the desired region specified by the **-a** flag. Also, by default, each copy of a logical partition is placed on a separate physical volume.

Notes:

- To use this command, you must either have `root` user authority or be a member of the **system** group.
- The **mklvcopy** command is not allowed on a snapshot volume group.
- When you create a copy of a logical volume with a superstrict allocation policy, the **mklvcopy** command first attempts to mimic the physical-partition mapping of the first mirror copy onto another set of disks in the volume group. This algorithm ignores the interphysical and intraphysical volume allocation policies, even when the policies are specified as arguments to the **mklvcopy** command. If it is not possible to mimic the first copy's physical partition mapping, the usual allocation algorithm, which utilizes the interphysical and intraphysical volume allocation policies, is used.

You can use the System Management Interface Tool (SMIT) **smit mklvcopy** fast path to run this command.

Flags

Note: The **-e** and **-s** flags are not valid with a striped logical volume.

Item	Description
-a *position*	Sets the intra-physical volume allocation policy (the position of the logical partitions on the physical volume). The *position* variable can be one of the following: **m** Allocates logical partitions in the outer middle section of each physical volume. This is the default position. **c** Allocates logical partitions in the center section of each physical volume. **e** Allocates logical partitions in the outer edge section of each physical volume. **ie** Allocated logical partitions in the inner edge section of each physical volume. **im** Allocates logical partitions in the inner middle section of each physical volume.

Item	Description
-e *range*	Sets the inter-physical volume allocation policy (the number of physical volumes to extend across, using the volumes that provide the best allocation). The *range* value is limited by the *upperbound* variable (set with the **-u** flag), and can be one of the following: **x** Allocates across the maximum number of physical volumes. **m** Allocates logical partitions across the minimum number of physical volumes. This is the default for the **-e** flag.
-k	Synchronizes data in the new partitions.
-m *mapfile*	Specifies the exact physical partitions to allocate. Partitions are used in the order given by the file designated by the *mapfile* parameter. All physical partitions belonging to a copy are allocated before allocating for the next copy. The *mapfile* format is: **PVname:PPnum1[-PPnum2]** where *pvname* is a physical volume name (for example, `hdisk0`). It is one record per physical partition or a range of consecutive physical partitions. **PVname** Name of the physical volume as specified by the system. **PPnum** Physical partition number. **Important:** When you use map files, you must understand and adhere to all LV-allocation parameters such as strictness, upperbound, and stripe width. Using map files bypasses the checks done in the LVM-allocation routines. This is important for striped LVs, which are assumed to have a typical striped allocation pattern conforming to the stripe width.
-p *copyn*=*mirrorpool*	Assigns mirror pools to the copies being created. A mirror pool is assigned to a copy using the *copyn*=*mirrorpool* parameter. Specify a mirror pool for each copy being created. To specify more than one *copyn*=*mirrorpool* pair, provide multiple **-p** *copyn*=*mirrorpool* flags.
-s *strict*	Determines the strict allocation policy. Copies of a logical partition can be allocated to share or not to share the same physical volume. The *strict* variable is represented by one of the following: **y** Sets a strict allocation policy, so copies for a logical partition cannot share the same physical volume. flag. **n** Does not set a strict allocation policy, so copies for a logical partition can share the same physical volume. **s** Sets a super strict allocation policy, so that the partitions allocated for one mirror cannot share a physical volume with the partitions from another mirror. See Note 4 for other effects of the superstrict allocation policy on **mklvcopy** behavior. **Note:** When changing a nonsuper strict logical volume to a super strict logical volume, you must specify physical volumes or use the **-u** flag.

Item	Description
-u *upperbound*	Sets the maximum number of physical volumes for new allocation. The value of the *upperbound* variable should be between one and the total maximum number of physical volumes per VG. When using super strictness, the upper bound indicates the maximum number of physical volumes allowed for each mirror copy. When using striped logical volumes, the upper bound must be multiple of *stripewidth*.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

To add physical partitions to the logical partitions in the logical volume `lv01`, so that a total of three copies exists for each logical partition, enter:

```
mklvcopy lv01 3
```

The logical partitions in the logical volume represented by directory `lv01` have three copies.

Files

Item	Description
/usr/sbin/mklvcopy	Contains the **mklvcopy** command.

mkmaster Command

Purpose

Executes the **ypinit** command and starts the NIS daemons to configure a controller server.

Syntax

/usr/sbin/mkmaster [**-s** *HostName* [**,***HostName* ...]] [**-O** | **-o**] [**-E** | **-e**] [**-P** | **-p**] [**-U** | **-u**] [**-C** | **-c**] [**-I** | **-B** | **-N**]

Description

The **mkmaster** command invokes the **ypinit** command to build the NIS maps for the current domain, if the domain name of the system is currently set. After the **ypinit** command completes successfully, the **mkmaster** command uncomments the entries in the **/etc/rc.nfs** file for the **ypserv** command, **yppasswdd** command, **ypupdated** command, and **ypbind** command.

You can use the System Management Interface Tool (SMIT) **smit mkmaster** fast path to run this command.

Flags

Item	Description
-s *HostName* [**,***HostName ...*]	Specifies the worker host names for this controller server. These worker hosts must be configured after the controller server has been configured. The **mkmaster** command automatically adds the current host to this list.
-O	Overwrites existing maps for this domain.
-o	Prevents the overwriting of existing maps for this domain. This flag is the default.
-E	Prevents further action if errors are encountered while building new maps. This is true for both the **ypinit** command and the **mkmaster** command. This flag is the default.
-e	Does not exit from the **ypinit** command and the **mkmaster** command if errors are encountered.
-P	Starts the **yppasswdd** daemon along with the **ypserv** daemon.
-p	Suppresses the start of the **yppasswdd** daemon. This flag is the default.
-U	Starts the **ypupdated** daemon along with the **ypserv** daemon.
-u	Suppresses the start of the **ypupdated** daemon. This flag is the default.
-C	Starts the **ypbind** daemon along with the **ypserv** daemon. This flag is the default.
-c	Suppresses the start of the **ypbind** daemon.
-I	Directs the **mkmaster** command to change the **/etc/rc.nfs** file to start the appropriate daemons on the next system restart. The execution of the **ypinit** command occurs when this command is invoked.
-B	Executes the **ypinit** command, uncomments the entries in the **/etc/rc.nfs** file, and starts the daemons. This flag is the system default.
-N	Executes the **ypinit** command and starts the appropriate daemons without changing the **/etc/rc.nfs** file.

Example

To execute the **ypinit** command, overwrite any existing maps for the current domain, and make `host1` and `host3` worker servers, enter:

```
mkmaster -s host1,host3 -O -p -u -B
```

This command will not start the **yppasswdd** daemon or the **ypupdated** daemon.

Files

Item	Description
/var/yp/*domainname* directory	Contains the NIS maps for the NIS domain.
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.

mknamsv Command

Purpose

Configures TCP/IP-based name service on a host for a client.

Syntax

mknamsv { **-a "***Attribute*=*Value* ..." | **-A** *FileName* }

Description

The **mknamsv** high-level command configures a TCP/IP instance to use a name server. It calls the **namerslv** low-level command to configure the **resolv.conf** file appropriately.

You can use the System Management Interface Tool (SMIT) **smit mknamerslv** fast path to run this command.

Flags

Item	Description
-A *FileName*	Specifies the name of the file containing **named** daemon initialization information.
-a "*Attribute*=*Value...*"	Specifies a list of attributes with corresponding values to be used for updating the **named** server initialization files in the database. Attributes available are: ***domain*** Domain name ***nameserver*** Internet address of name server in dotted decimal format

Examples

1. To configure the name server initialization files, enter the command in the following format:

   ```
   mknamsv  -a"domain=austin.century.com nameserver=192.9.200.1"
   ```

   In this example the domain name and name server address are updated. The previous domain and name server are overwritten.
2. To configure name server initialization files according to information in another file, enter the command in the following format:

   ```
   mknamsv  -A namsv.file
   ```

   In this example, the file that contains the configuration information is `namsv.file`. The **"attribute=value"** pairs must not be placed in one line. As an example, enter the **"attribute=value"** pairs to `namsv.file` in the following format:

   ```
   domain=austin.century.com
   nameserver=192.9.200.1
   ```

Files

Item	Description
/etc/resolv.conf	Contains domain name server information for local resolver routines.

mknetid Command

Purpose

Generates data for the **netid.byname** map for use by the Network Information Services (NIS).

Syntax

To Create an NIS Map:

/usr/sbin/mknetid [**-q**] [**-p** *PasswordFile*] [**-g** *GroupFile*] [**-h** *HostsFile*] [**-m** *NetidFile*]

Description

The **mknetid** command is used to produce the data for the **netid.byname** NIS map. It will parse the files specified on the command line and build the corresponding netid keys and values. Users will get the following entries:

```
unix.<uid>@<domainname> <uid>:<gid1>,<gid2>,...
```

Hosts will get the following entries:

```
unix.<hostname>@<domainname> 0:<hostname>
```

The domainname that is used is the same that is configured on the system at the time **mknetid** is run. The generated data is sent to **stdout**. Each line contains one entry, with the key and the data separated by a space.

Flags

Item	Description
-q	Quiet mode - do not report any warnings about the data.
-p *PasswordFile*	Specifies which passwd file to be used for reading the list of users.
-g *GroupFile*	Specifies which groups file to be used for reading the list of group memberships.
-h *HostsFile*	Specifies which hosts file to be used for reading the list of hostnames.
-m *NetidFile*	Specifies a file from which to read any additional netid entries to be included.

Files

Item	Description
/var/yp/Makefile	**mknetid** is most commonly used when rebuilding the NIS databases using **/var/yp/Makefile**.
/etc/passwd	Where *PasswordFile* resides.
/etc/groups	Where *GroupFile* resides.
/etc/hosts	Where *HostsFile* resides.
/etc/netid	Where *NetidFile* resides.

mknfs Command

Purpose

Configures the system to run NFS.

Syntax

/usr/sbin/mknfs [**-I** | **-N** | **-B**]

Description

The **mknfs** command configures the system to run the Network File System (NFS) daemons. The **mknfs** command adds an entry to the **inittab** file so that the **/etc/rc.nfs** file is executed on system restart.

Flags

Item	Description
-B	Adds an entry to the **inittab** file to execute the **/etc/rc.nfs** file on system restart. The **mknfs** command also executes the **/etc/rc.nfs** file immediately to start the NFS daemons. This flag is the default.
-I	Adds an entry to the **inittab** file to execute the **/etc/rc.nfs** file on system restart.
-N	Starts the **/etc/rc.nfs** file to start the NFS daemons immediately. When started this way, the daemons run until the next system restart.

Files

Item	Description
inittab	Controls the initialization process of the system.
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

mknfsexp Command

Purpose

Exports a directory to NFS clients.

Syntax

/usr/sbin/mknfsexp **-d** *Directory* [**-e** *pathname*] [**-f** *Exports_File*] [**-t** [{ **rw** | **ro** | **remove** } { **rm** **-h** *HostName* [**,** *HostName ...*] }]] [**-a** *UID*] [**-r** *HostName* [**,** *HostName ...*]] [**-c** *HostName* [**,** *HostName ...*]] [**-s** | **-n**] [**-I** | **-B** | **-N**] [**-P** | **-p**] [**-D** {yes | no}] [**-v**] [**-x** | **-X**] [**-S** *flavor* [*,flavor*]] [-G *rootpath@host*[+*host*][**:** *rootpath@host*[+*host*]]] [-g *rootpath@host*[+*host*][**:** *rootpath@host*[+*host*]]] [-o *Ordering*]

Description

The **mknfsexp** command takes the flags and parameters specified and constructs a line that is syntactically correct for the **/etc/exports** file. If this command is invoked with the **-B** flag, an entry will be added to the **/etc/exports** file and the **exportfs** command is invoked to export the directory specified.

Alternatively, the **-I** flag adds an entry to the **exports** file and does not export the directory, or the **-N** flag does not add an entry to the **exports** file but does export the directory.

Flags

Item	Description
-a *UID*	Uses the UID variable as the effective user ID only if a request comes from an unknown user. The default value of this option is -2. **Note:** Root users (UID 0) are always considered unknown by the NFS server, unless they are included in the root option. Setting the value of UID to -1 disables anonymous access.
-B	Adds an entry to the **/etc/exports** file and the **exportfs** command is executed to export the directory. This flag is the default.
-c *HostName* [**,** *HostName*] ...	Gives mount access to each of the clients listed. A client can either be a host or a netgroup. The default is to allow all hosts access.
-d *Directory*	Specifies the directory that is to be exported or changed.
-D {yes | no}	Enables or disables file delegation for the specified export. This option overrides the system-wide delegation enablement for this export. The system-wide enablement is done through **nfso**.
-e *pathname*	Specifies an export name for the directory.
-f *Exports_File*	Specifies the full path name of the exports file to use if other than the **/etc/exports** file.
-g *rootpath@host* [+*host*] [:*rootpath@host* [+*host*]]	The specified directory will be marked with replica information. If the server becomes unreachable by an NFS client, the client can switch to one of the specified servers. This option is only accessible using NFS version 4 protocol, and version 4 access must be specified in the options. Because the directory is being exported for client access, specifying NFS version 2 or version 3 access will not cause an error, but the request will simply be ignored by the version 2 or version 3 server. This option cannot be specified with the `-G` flag. Only the host part of each specification is verified. The administrator must ensure that the specified *rootpaths* are valid and that the target servers contain appropriate data. If the directory being exported is not in the replica list, that directory will be added as the first replica location. The administrator should ensure that appropriate data exists at the replica locations. For a more complete description of replication, see the **exportfs** command. The `-g` option is available only on AIX 5.3 with 5300-03 or later. **Note:** A referral or replica export can only be made if replication is enabled on the server. Use `chnfs -R on` to enable replication.

Item	Description
-G *rootpath@host* [+*host*] [:*rootpath@host* [+*host*]]	A namespace referral will be created at the specified path. The referral directs clients to the specified alternate locations where they can continue operations. A referral is a special object. If a nonreferral object exists at the specified path, the export is disallowed and an error message is printed. If nothing exists at the specified path, a referral object is created there that includes the path name directories leading to the object. A referral cannot be specified for the `nfsroot`. The name `localhost` cannot be used as a *hostname*. The `-G` option is allowed only for version 4 exports. If the export specification allows version 2 or version 3 access, an error message will be printed and the export will be disallowed. The administrator should ensure that appropriate data exists at the referral locations. For a more complete description of referrals, see the **exportfs** command. The `-G` option is available only on AIX 5L Version 5.3 with the 5300-03 Recommended Maintenance package or later. **Note:** A referral or replica export can only be made if replication is enabled on the server. Use `chnfs -R on` to enable replication.
-h *HostName* [**,** *HostName*] ...	Specifies which hosts have read-write access to the directory. This option is valid only when the exported file is to be read-mostly.
-I	Adds an entry to the **/etc/exports** file so that the next time the **exportfs** command is run during system restart, the directory will be exported.
-n	Does not require the client to use the more secure protocol. This flag is the default.
-N	Does not add an entry to the **/etc/exports** file but the **exportfs** command is run with the correct parameters so that the directory is exported.
`-o` *Ordering*	Defines how the alternate locations list is generated from the servers that you specified on the **refer** or **replicas** option. The option applies only to directories exported for access by NFS version 4 protocol. The *Ordering* parameter has three allowable values: **full** All of the servers are scattered to form the combinations of alternate locations. **partial** The first location of all the combinations is fixed to the first server specified on the **refer** or **replicas** option. The rest of the locations and the first location are scattered as if they are scattered using the `scatter=full` method. **none** No scatter is to be used. The value can also be used to disable scattering if it was enabled previously.
-p	Specifies that the exported directory is not a public directory.
-P	Specifies that the exported directory is to be a public directory.
-r *HostName* [**,** *HostName*] ...	Gives root users on the specified hosts access to the directory. The default is for no hosts to be granted root access.
-s	Requires clients to use a more secure protocol when accessing the directory.

Item	Description
-S *flavor*[,*flavor*]	May be used in conjunction with the **-c**, **-t**, or **-r** options to associate the option with one or more specific security methods. Most **exportfs** options can be clustered using the **sec** option. Any number of **sec** stanzas may be specified, but each security method can be specified only once. Allowable flavor values are: **sys** UNIX authentication. **dh** DES authentication. **none** Use the anonymous ID if it has a value other than -1. Otherwise, a weak auth error is returned. **krb5** Kerberos. Authentication only. **krb5i** Kerberos. Authentication and integrity. **krb5p** Authentication, integrity, and privacy.
-t *Type*	Specifies whether the directory is read-write, read-only, or read-mostly. The possible values for the *Type* variable are: **rw** Exports the read-write directory. This is the system default. **ro** Exports the read-only directory. **remove** Removes the exported directory. **rm** Exports the read-mostly directory. If chosen, the **-h** flag must be used to specify the hosts that have read-write permission.
-v *number* [**,** *number*] ...	The directory specified by the **-d** option is made available to clients using the specified NFS versions. Valid values are 2, 3, or 4. You can export two entries for the same directory with different versions 2 (or 3) and 4.
-x	Accepts the replica location information specified with the **-g** option as-is. Does not insert the server's primary hostname into the list if it is not present. This flag is intended for use with servers with multiple network interfaces. If none of the server's host names are in the replica list, NFSv4 clients might treat the location information as faulty and discard it.
-X	Enables auto-insert of the primary hostname into the replica list. If the server's primary hostname is not specified in the replica list, the hostname will be added as the first replica location.

Examples

1. To export a directory with read-only permission, enter:

```
mknfsexp -d /usr -t ro
```

   In this example, the `mknfsexp` command exports the `/usr` directory with read-only permission.
2. To export a directory with read-mostly permission and a secure protocol to specific hosts, enter:

```
mknfsexp -d /home/guest -t rm -h bighost,littlehost -s
```

In this example the `mknfsexp` command exports the `/home/guest` directory with read-mostly permission, using more secure protocol.

3. To export a directory with read-write permission to a specific netgroup and specific hosts, and to make the export effective on the next system restart, enter:

```
mknfsexp -d /usr -t rw -c host1,host3,grp3 -I
```

In the above example, the `mknfsexp` command exports the `/usr` directory and gives read and write permission to `host1`, `host2`, and `grp3`. The `-I` flag makes this change effective on the next system restart.

4. To export a directory with read-only permission to an exports file other than **/etc/exports**, enter:

```
mknfsexp -d /usr -t ro -f /etc/exports.other
```

In the above example, the `mknfsexp` command exports the `/usr` directory with read-only permission to the `/etc/exports.other` file.

5. To export the **/common/documents** directory to allow access only to clients using NFS version 4 protocol, enter:

```
mknfsexp -d /common/documents -v 4
```

6. To export the **/common/documents** directory, allowing access to `client1` and `client2` for clients using krb5 access, enter:

```
mknfsexp -d /common/documents -S krb5 -r client1,client2
```

7. To export the **/common/documents** directory with full scattering for the hosts named `s1` and `s2` specified as referrals, enter the following command:

```
mknfsexp -d /common/documents -v 4 -G  /common/documents@s1:/common/
documents@s2 -o full
```

8. To export the **/common/documents** directory with partial scattering at hosts named `s1`, `s2` and `s3`, specified as replicas, enter the following command:

```
mknfsexp  -d /common/documents -v 4 -g /common/documents@s1:/common/
documents@s2:/common/documents@s3 -o partial
```

9. To export the **/common/documents** directory with the export name `/exports1/cool/mike`, enter the following command:

```
mknfsexp -d /common/documents -e /exports1/cool/mike -S sys -v 4
```

Files

Item	Description
/etc/exports	Lists the directories that the server can export.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

mknfsmnt Command

Purpose

Mounts a directory from an NFS server.

Syntax

/usr/sbin/mknfsmnt -f *PathName* **-d** *RemoteDirectory* **-h** *RemoteHost* [**-t** { **rw** | **ro** }] [**-m** *MountTypeName*] [**-w** { **fg** | **bg** }] [**-X** | **-x**] [**-S** | **-H**] [**-Y** | **-y**] [**-Z** | **-z**] [**-e** | **-E**] [**-a** | **-A**] [**-j** | [**-J**] [**-q** | [**-Q**] [**-g** | [**-G**] [**-s** | **-n**] [**-I** | **-B** | **-N**] [**-r** *TimesToRetry*] [**-R** *NumRetrans*] [**-b** *ReadBufferSize*] [**-c** *WriteBufferSize*] [**-o** *TimeOut*] [**-P** *PortNumber*] [**-u** *AcRegMin*] [**-U** *AcRegMax*] [**-v** *AcDirMin*] [**-V** *AcDirMax*] [**-T** *AcTimeOut*] [**-p** *NumBiods*] [**-K** { *any* | *2* | *3* | *4* }] [**-k** { *any* | *tcp* | *udp* }] [**-l** | **-L**] [**-M** *security_methods*] [**-i** { *dio* | *cio* [,*cior*] }]

Description

The **mknfsmnt** command constructs an entry that is appended to the **/etc/filesystems** file, thus making a file system available for mounting. If the mount is to be permanent, this entry remains. If the mount is temporary, the flags are used directly for the **mount** command. If the mount is soft and the server does not respond, the system returns an error. If the mount is hard, the client continues trying until the server responds. The hard mount is the default mount.

Flags

Item	Description
-A	The **/etc/filesystems** entry for this file system specifies that it should be automatically mounted at system restart.
-a	The **/etc/filesystems** entry for this file system specifies that it should not be automatically mounted at system restart. This is the default flag.
-B	Adds an entry to the **/etc/filesystems** file and attempts to mount the file system. This is the default flag.
-b *ReadBufferSize*	Indicates the size of the read buffer in bytes specified by the *ReadBufferSize* variable.
-c *WriteBufferSize*	Indicates the size of the write buffer in bytes specified by the *WriteBufferSize* variable.
-d *RemoteDirectory*	Specifies the directory that is mounted on the path name specified.
-E	Allows keyboard interrupts on hard mounts.
-e	Prevents keyboard interrupts on hard mounts. This is the default flag.
-f *PathName*	Specifies the mount point for the remote directory.
-G	Directs any file or directory created on the file system to inherit the group ID of the parent directory.
-g	Does not direct new files or directories created on the file system to inherit the group ID of the parent directory. This is the default flag.
-H	Creates a hard mount, which causes the client to continue retrying until the server responds. This is the default flag.
-h *RemoteHost*	Specifies the NFS server that is exporting the directory.
-I	Causes an entry to be added to the **/etc/filesystems** file. The directory is not mounted.

Item	Description
-i	Specifies I/O mode for the mount. The options are: ***cio*** Specifies concurrent I/O mode. Specifies the file system to be mounted for concurrent readers and writers. I/O on files in this file system will behave as if they had been opened with **O_CIO** specified in the **open()** system call. ***dio*** Specifies direct I/O mode. Specifies that I/O on the file system will behave as if all the files had been opened with **O_DIRECT** specified in the **open()** system call. ***cior*** Specifies concurrent I/O with read-only mode. **Note:** For more information on the **cio** and **dio** options, see the **mount** command.
-J	Indicates that **acl**s are used on this mount.
-j	Indicates that **acl**s are not used on this mount. This is the default flag.
-K	Specifies the NFS version used for this NFS mount. The options are: ***any*** Uses the `mount` command to determine the correct match. Refer to the `mount` command for a description of the current default behavior. **2** Specifies NFS Version 2. **3** Specifies NFS Version 3. **4** Specifies NFS Version 4.
-k	Specifies the transport protocol used for the mount. The options are: ***any*** Uses the mount command to select the protocol to use. TCP protocol is the preferred protocol. ***tcp*** Specifies the TCP protocol. ***udp*** Specifies the UDP protocol.
L	Indicates that the lock requests are handled locally without connecting to the server.
l	Indicates that the lock requests are not handled locally. The server handles the lock requests.
-M *security_methods*	A list of security methods to use when attempting the mount. A comma separated list of the values `sys`, dh, `krb5`, `krb5i`, `krb5p`, which correspond to UNIX, DES, Kerberos 5, Kerberos 5 with integrity, and Kerberos 5 with privacy. Multiple values are allowed, but are meaningful only with NFS Version 4 mounts. If multiple methods are given for a Version 2 or 3 protocol mount, the first method is used. For a NFS Version 4 mount, the methods are tried in the listed order.

Item	Description
-m *MountTypeName*	Specifies the type of file system to mount. File system types are specified in the **/etc/filesystems** file with the **type** variables. When the **mount -t** *MountTypeName* command is issued, all the currently unmounted file systems with a type equal to the *MountTypeName* are mounted.
-N	Mounts the directory with the options specified but does not modify the **/etc/filesystems** file.
-n	Instructs the mount not to use a more secure protocol. This is the default flag.
-o *TimeOut*	Indicates the length of the NFS timeout in tenths of a second as specified by the *TimeOut* variable.
-P *PortNumber*	Indicates the Internet Protocol port number for the server.
-p *NumBiods*	Specifies the number of **biod** daemons that are allowed to work on a particular file system. The **biod** daemons handle client requests. The default number of daemons is 7 for NFS Version 2 and 32 for NFS Version 3 and NFS Version 4.
-Q	Requests that no posix pathconf information be exchanged and made available on an NFS Version 2 mount. Requires a mount Version 2 **rpc.mountd** at the NFS server.
-q	Specifies that no posix pathconf information is exchanged if mounted as an NFS Version 2 mount. This is the default flag.
-r *TimesToRetry*	Indicates the number of times to retry a mount. The default value is 1000.
-R *NumRetrans*	For a soft mount, this flag specifies the number of times that a request has to be transmitted if it is not acknowledged by the server. If the request is unacknowledged after *NumRetrans* transmissions, the client gives up the request. If this flag is not specified, the default value 3 is used.
-S	Creates a soft mount, which means the system returns an error if the server does not respond.
-s	Instructs the mount to use a more secure protocol.
-T *AcTimeOut*	Sets the minimum and maximum times allowed for regular files and directories to the number of seconds specified by the *Actimeo* variable. If this flag is specified, the other cached attribute times are overridden.
-t *Type*	Specifies that the directory is either read-write or read-only. **rw** Mounts the directory read-write. This type is the default for the system. **ro** Mounts the directory read-only.
-U *AcRegMax*	Holds cached attributes for no more than the number of seconds specified by the *AcRegMax* variable after file modification.
-u *AcRegMin*	Holds cached attributes for at least the number of seconds specified by the *AcRegMin* variable after file modification.
-V *AcDirMax*	Holds cached attributes for no more than the number of seconds specified by the *AcDirMax* variable after directory update.

Item	Description
-v *AcDirMin*	Holds cached attributes for at least the number of seconds specified by the *AcDirMin* variable after directory update.
-w *Location*	Indicates where the mount should be attempted. The *Location* variable can have one of the following values: **fg** Attempts the mount in the foreground. This is the default value. **bg** Attempts the mount in the background. If background is specified and the attempt to mount the directory fails, the mount will be retried in the background.
-x	Specifies that the server does not support long device numbers. Use this flag when mounting from an NFS server that does not correctly handle device numbers that are 32 bits long.
-X	Specifies that the server does support long device numbers. This is the default flag.
-y	Indicates that the execution of suid and sgid programs is not allowed in this file system.
-Y	Indicates that the execution of suid and sgid programs are allowed in this file system. This is the default flag.
-z	Indicates that device access through this mount is not allowed; that is, the device cannot be opened on this mount point.
-Z	Indicates that device access through this mount is allowed. This is the default flag.

Example

To add the mount of a remote directory, enter:

```
mknfsmnt -f /usr/share/man -d /usr/share/man -h host1
```

In this example, the `mknfsmnt` command mounts the remote directory `/usr/share/man` on the `/usr/share/man` directory that resides on `host1`.

Files

Item	Description
/etc/filesystems	Lists the remote file systems to be mounted during the system restart.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

mknfsproxy Command

Purpose

Creates a new NFS proxy-enabled Cachefs instance that is backed with an NFS client mount to a back-end NFS server.

Syntax

/usr/sbin/mknfsproxy [**-L**] **-c** *local_cache_directory* **-d** *Cachefs_mount_point* [**-o param=n**[**, param=n**]] **-m** [*nfs_mount_options*] *remote_server*:*remote_directory* [**-e** [*export_option*, [*export_option*]]]

Description

The local file system used by the created Cachefs instance must be a JFS2 file system. The required inputs include the remote server and directory (*remote_server*:*remote_directory*) that the Cachefs instance will access, the local directory (*local_cache_directory*) where information will be cached, and the directory where the Cachefs will be mounted.

After the cache is initialized, the Cachefs instance is mounted and ready to be NFS exported. Provide NFS export information so that the cached view will also be NFS exported using the specified options.

Flags

Item	Description
`-c`	Specifies the local JFS2 file system directory where Cachefs will store cached data and state. This is a required option.
`-d`	Specifies the directory where Cachefs will be mounted. This is a required option.
`-e`	Specifies the NFS server export options for the created Cachefs instance. If this is supplied, the created Cachefs instance will also be NFS exported using the supplied options. If this option is not supplied, the created Cachefs instance will be exported with the same NFS version specified by the `-m` option.
`-L`	Causes the Cachefs instance to acquire a single lock from its associated NFS back-end that covers the entire file when any byte range locks are requested. When the count of byte range locks drops to 0 (zero), the lock on the back-end NFS server is released.
`-m`	Specifies the NFS client mount, which might optionally include NFS client mount options as described in the `mount` man page. This is a required option, and the remote server and remote directory must be supplied.
`-o`	Specifies Cachefs configuration options in the form `param=n`. For descriptions of the Cachefs resource parameters, refer to the `cfsadmin` command.

Parameters

Item	Description
Cachefs_mount_point	Specifies where the proxy-enabled Cachefs instance is to be mounted.
export_option	Specifies which options of the `export` command are used for the Cachefs instance.
local_cache_directory	Specifies the local directory where information is cached.
nfs_mount_options	Specifies the NFS client options of the `mount` command.
remote_directory	Specifies the remote directory that the Cachefs instance accesses.
remote_server	Specifies the remote server that the Cachefs instance accesses.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Examples

1. To create a proxy-enabled Cachefs instance at `/edge` that accesses `/project1` at NFS server `foo`, enter:

   ```
   mknfsproxy -c /cache/project1 -d /edge -m hard,vers=4,intr foo:/project1
   ```

   In the preceding example, information accessed through `/edge` is cached at `/cache/project1`.

   The following variation of the preceding example also exports the created Cachefs instance (`proj1_cached`) for NFS V4 access with authentication flavor of Kerberos 5 and an external name set to `/nfs4/projects/project1`:

   ```
   mknfsproxy -c /cache/project1 -d /edge -m hard,vers=4,intr foo:/project1
       -e sec=krb5,vers=4,exname=/nfs4/projects/project1
   ```

Location

`/usr/sbin/mknfsproxy`

mknod Command

Purpose

Creates a special file.

Syntax

Only executed by root or system group member

mknod *Name* { **b** | **c** } *Major Minor*

Creates FIFO (first-in, first-out) files, which are also called pipes or pipelines

mknod *Name* { **p** }

Description

The **mknod** command makes a directory entry and corresponding i-node for a special file. The first parameter is the name of the entry device. Select a name that is descriptive of the device. The **mknod** command has two forms that have different flags.

The first form of the **mknod** command can only be executed by root or a member of the system group. In the first form, the **b** or **c** flag is used. The **b** flag indicates the special file is a block-oriented device (disk, diskette, or tape). The **c** flag indicates the special file is a character-oriented device (other devices).

The last two parameters of the first form are numbers specifying the *Major* device, which helps the operating system find the device driver code, and the *Minor* device, that is the unit drive or line number, which may be either decimal or octal. The major and minor numbers for a device are assigned by the device's configure method and are kept in the CuDvDr class in ODM.

It is important that the major and minor numbers be defined in this object class to ensure consistency of device definitions through the system.

In the second form of the **mknod** command, the **p** flag is used to create FIFO pipelines.

Flags

Item	Description
b	Indicates the special file is a block-oriented device (disk, diskette, or tape).
c	Indicates the special file is a character-oriented device (other devices).
p	Creates FIFOs (named pipelines).

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To create the special file for a new diskette drive, enter the following command:

   ```
   mknod /dev/fd2  b 1 2
   ```

   This command creates the **/dev/fd2** special file that is a special block file with the major device number 1 and the minor device number 2.
2. To create the special file for a new character drive, enter the following command:

   ```
   mknod /dev/fc1  b 1 2
   ```

   This command creates the /dev/fc1 special file that is a special character file with the major device number 1 and the minor device number 2.
3. To create a FIFO pipe file, enter the following command:

   ```
   mknod fifo1 p
   ```

   This command creates a FIFO pipe file that has the name fifo1.

Files

Item	Description
/usr/sbin/mknod	Contains the **mknod** command.

mknotify Command

Purpose

Adds a notify method definition to the Notify object class.

Syntax

mknotify -n *NotifyName* **-m** *NotifyMethod*

Description

The **mknotify** command adds a notify method definition to the **Notify** object class. When a notify method is defined for both a subsystem name and a group name, the subsystem name takes precedence. For example, if the subsystem notify method is executed by the System Resources Controller (SRC), the group notify method is not performed.

The SRC places the name of the unsuccessful subsystem as the first argument to the method and the name of the unsuccessful subsystem group as the second argument.

Flags

Item	Description
-m *NotifyMethod*	Specifies an absolute path to an executable program that starts when the subsystem stops abnormally.
-n *NotifyName*	Specifies the subsystem or group name to which the notify method belongs. The *NotifyName* variable must exist as either a valid subsystem name or a valid group name in the **Subsystem** object class. The **mknotify** command is unsuccessful if the *NotifyName* variable already exists in the **Notify** object class.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To add a notify method for the `srctest` subsystem, enter:

   ```
   mknotify  -n srctest  -m /usr/lpp/srctest/failure
   ```

   This adds a subsystem notify method for the `srctest` subsystem, with a notify method designated in the `/usr/lpp/srctest/failure` file.
2. To add a notify method for the `tcpip` group, enter:

```
mknotify -n tcpip -m /usr/lpp/tcpip/tcpfailure
```

This adds a group notify method for the `tcpip` group, with a notify method designated in the `/usr/lpp/tcpip/tcpfailure` file.

Files

Item	Description
/etc/objrepos/SRCsubsys	Specifies the SRC **Subsystem Configuration** object class.
/etc/objrepos/SRCnotify	Specifies the SRC **Notify Method** object class.

mkpasswd Command

Purpose

Organizes the basic user database for efficient searches.

Syntax

mkpasswd [**-v**] { **-f** | **-d** | **-c**} |*indexname*

Description

The **mkpasswd** generates indexes over certain security files. These indexes are used by the **getpwnam**, **getpwuid**, **getuserattr**, and **putuserattr** library subroutines.

This approach significantly enhances performance for large user base systems. The following indexes, defined in **/usr/include/usersec.h**, are created:

Item	Description
/etc/passwd.nm.idx:	Index over **/etc/passwd** file using username as key.
/etc/passwd.id.idx:	Index over **/etc/passwd** file using userid number as key.
/etc/security/passwd.idx:	Index over **/etc/security/passwd** file.
/etc/security/lastlog.idx:	Index over **/etc/security/lastlog** file.

Notes:

1. Modifying the security files over which indexes are built by an editor disables the use of indexing mechanism.
2. Indexed read of a data file is automatically done if a corresponding index exists over the file and is not older than it (except for lastlog index) .
3. In order for indexed mechanism to be used at login, the **mkpasswd** command must have generated indexes.
4. The indexing mechanism replaces the previous hashing mechanism which used dbm files.

Flags

Item	Description
-v	Reports progress if index built.
-f	Forces building of all indexes.
-d	Deletes all indexes.

Item	Description
-c	Checks all indexes and rebuilds the ones that look suspicious.
indexname	Forces building of a particular index.

Security

Access Control: Only the root user and members of the security group should have execute (x) access to this command. The command should be setuid to the root user so the command has access to the user database. Members of the security group should have access to all the files listed in the Files section. This command should have the **trusted computing base** attribute.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files Accessed:

Item	Description
Mode	**File**
r	**/etc/passwd**
rw	**/etc/passwd.nm.idx***nnnn***.tmp** and **/etc/passwd.id.idx***nnnn***.tmp** where *nnnn* is the process id.
r	**/etc/security/passwd**
rw	**/etc/security/passwd.idx**
rw	**/etc/security/passwd.idx***nnnn***.tmp** where *nnnn* is the process id
r	**/etc/security/lastlog**
rw	**/etc/security/lastlog.idx**
rw	**/etc/security/lastlog.idx***nnnn***.tmp** where *nnnn* is the process id

Examples

1. To create and enable indexed read of security files, enter:

```
mkpasswd -f
```

2. To create and enable indexed read of only the /etc/security/passwd file, enter:

```
mkpasswd /etc/security/passwd.idx
```

3. To check and rebuild outdated or bad indexes, enter:

```
mkpasswd -c
```

Files

Item	Description
/usr/sbin/mkpasswd	Contains the mkpasswd command.

Item	Description
/etc/passwd	Contains basic user attributes.
/etc/security/passwd	Contains user password attributes
/etc/security/lastlog	Contains lastlog related attributes

mkpath Command

Purpose

Adds to the system another path to an MPIO capable device.

Syntax

mkpath [**-l** *Name*] [**-p** *Parent*] [**-w** *Connection*] [**-i** *PathID*]

mkpath [**-l** *Name*] [**-p** *Parent*] [**-w** *Connection*] [**-d**]

mkpath -h

Description

The **mkpath** command defines, and possibly configures, one or more paths to the target device (**-l** *Name*). The paths are identified by a combination of the **-l** *Name*, **-p** *Parent*, and **-w** *Connection* flags. Both the target device and parent must be previously defined in the system to define a path. They both must be "AVAILABLE" to configure a path.

If the **-d** flag is specified, the **mkpath** command only defines the new path definition to the system. If the **-d** flag is not specified, the **mkpath** command attempts to define the path, if it does not already exist, before it attempts to configure the path. Configuring a path requires the path to already be defined and both the device and the parent device to already be configured.

The **mkpath** command displays a status message upon completion. It is possible for some paths to configure and others to fail.

Note that any device that cannot be manually defined using the **mkdev** command will not be able to have paths manually defined to using the **mkpath** command. These limitations are due to the way that path information is stored for these devices. Fiber channel devices fall into this category.

The **mkpath** command provides status messages about the results of operation. Messages in one of the following formats will be generated:

path [available | defined]
: This message is displayed when **mkpath** is run on a single path. If the path is successfully configured the message "path available" is displayed. If the path is not successfully configured and there is no explicit error code returned by the method, the message "path defined" is displayed.

paths available
: This message is displayed if multiple paths were identified and all paths were successfully configured.

some paths available
: This message is displayed if multiple paths were identified, but only some of them were successfully configured.

no paths processed
: This message is generated if no paths were found matching the selection criteria.

Flags

Item	Description
-d	Defines a new path to a device by adding a path definition to the system. The new path will not automatically be configured when the **-d** flag is specified. Note that only one path may be defined at a time.
-h	Displays the command usage message.
-i *PathID*	Indicates the path ID associated with the path to be added and is used to uniquely identify a path. This flag cannot be used with the **-d** flag.
-l *Name*	Specifies the logical device name of the target device to which the path(s) are being added. The path(s) to be added are qualified by the **-p** and **-w** flags.
-p *Parent*	Indicates the logical device name of the parent device associated with the path(s) to be added. This flag is required if the **-d** flag is specified.
-w *Connection*	Indicates the connection information associated with the path to be added. This flag is required if the **-d** flag is specified.

Security

Privilege Control: Only the **root** user and members of the **system** group have execute access to this command.

Auditing Events:

Event	Information
DEV_Change	mkpath,Define,<define method arguments>
DEV_Change	mkpath,Configure,<configure method arguments>

Examples

1. To define and configure an already defined path between scsi0 and the hdisk1 device at SCSI ID 5 and LUN 0 (i.e., connection 5,0), enter:

   ```
   mkpath -l hdisk1 -p scsi0 -w 5,0
   ```

   The system displays a message similar to the following:

   ```
   path available
   ```

2. To configure an already defined path from '**fscsi0**' to fiber channel disk '**hdisk1**', the command would be:

   ```
   mkpath -l hdisk1 -p fscsi0
   ```

   The message would look similar to:

   ```
   path available
   ```

3. To only add to the Customized Paths object class a path definition between **scsi0** and the **hdisk1** disk device at SCSI ID 5 and LUN 0, enter:

   ```
   mkpath -d -l hdisk1 -p scsi0 -w 5,0
   ```

   The system displays a message similar to the following:

   ```
   path defined
   ```

Files

Item	Description
/usr/sbin/mkpath	Contains the **mkpath** command.

mkprojldap Command

Purpose

Configures the LDAP client and server machines for handling advanced accounting subsystem data.

Syntax

`mkprojldap -s -h` *hostname* `-D` *bindDN* `-w` *bindPWD* `-i -p` *projectInstallPoint* `-a` *adminInstallPoint*

`mkprojldap -u -h` *hostname* `-D` *bindDN* `-w` *bindPWD*

`mkprojldap -c -D` *bindDN* `-w` *bindPWD* [`-p` *accountingProjectDN*] [`-a` *accountingAdminDN*] [`-r` *cron*]

`mkprojldap` { `-l` | `-L` [`-D` *bindDN* `-w` *bindPWD*] | `-V` } [`-p`] [`-a`]

Description

The `mkprojldap` command configures the LDAP server and client machines for handling the advanced accounting subsystem data. The LDAP server and client relationship must already be defined, and `mkprojldap` makes only incremental changes. The `mkprojldap` command can be used to configure the basic LDAP connection.

To add advanced accounting support to the LDAP server, the LDAP schema for advanced accounting must be uploaded to the server. The schema describes the format of advanced accounting data to the server, enabling the server to process accounting data without being enabled specifically for accounting. This is accomplished with the `-u` option. The LDAP server is not dependent on advanced accounting. This command needs to be run only once for each LDAP server. After this command is run, use the `-s` option to define the location on the LDAP server where advanced accounting data is to be stored. This command can be run one or more times to establish one or more accounting domains. An LDAP client can only access only one accounting domain at a time.

To configure an LDAP client so that it receives advanced accounting data, use the `-c` option to specify the location of the advanced accounting data sets on the LDAP server that are to be used by the LDAP client. The `mkprojldap` command is used to configure absolute paths, which are known as *distinguished names* (DNs), to projects and admin policies. The advanced accounting subsystem stores project definitions and admin policies on LDAP servers, so there are two advanced accounting DNs that can be configured. The `mkprojldap -c` command must be run on each client.

Flags

Item	Description
`-a` *accountingAdminDN*	Specifies the accounting admin DN location on the LDAP server, when used with `-s` or `-c` options. When used with `-l` or `-L` options, this flag displays the accounting admin DN.
`-c`	Configures the LDAP client.
`-D` *bindDN*	Specifies the Bind DN to be used during the server configuration.
`-h` *hostname*	Specifies the host name of the LDAP server during the server configuration.

Item	Description
`-i`	Provides the admin (`-a`) and project (`-p`) install points during the server configuration.
`-L`	Displays the potential accounting DNs that are visible from the server.
`-l`	Displays the accounting DNs in the `ldap.cfg` file.
`-p` *accountingProjectDN*	Specifies the accounting project DN location on the LDAP server when used with the `-s` or `-c` options. When used with `-l` or `-L` options, this flag displays the accounting project DN.
`-r` *con*	Specifies the frequency for refreshing the LDAP repositories (hourly, daily, or off).
`-s`	Configures the LDAP server.
`-u`	Uploads the advanced accounting schema to the LDAP server.
`-V`	Displays the current LDAP client configuration details in a colon separated format.
`-w` *bindPWD*	Used to provide the Bind password for the Bind DN specified with the `-D` option.

Note: When using the preceding flags with this command, use the following guidelines:

- During server and client configuration, both the `-p` and `-a` arguments can be specified at the same time, but neither is required. If neither is specified, the `mkprojldap` command tries to compute the missing accounting DNs by searching for the objects on the LDAP server. These objects are `ou=projects` and `ou=adminpolicy`. If an object is found, the corresponding accounting DN is computed and added to the `ldap.cfg` file.
- While listing the accounting DNs using the `-l` or `-L` options, both `-p` and `-a` can be used. If neither of them are provided, all accounting DNs in the `ldap.cfg` file are listed.
- The colon-separated data displayed by the `-V` option takes the following format:

```
ldap-server-hostname:bind DN:bind password:default-projectdn:default-admindn:cron
```

Exit Status

Item	Description
`0`	Successful completion.
`>0`	An error occurred.

Examples

1. To upload the advanced accounting schema, type:

```
mkprojldap -u -h mozilla -D cn=root -w mozillapasswd
```

2. To configure the LDAP server, type:

```
mkprojldap -s -h ldap.svr.com -D cn=root -w passwd -i
             -p cn=aixdata,o=ibm -a cn=aixdata,o=ibm
```

This command creates two DNs in the following format:

```
ou=projects,ou=aacct,cn=aixdata,o=ibm and ou=adminpolicy,ou=aacct,cn=aixdata,o=ibm
```

3. To configure the LDAP client, type:

```
mkprojldap -c -D cn=testroot -w testpwd -p ou=projects,ou=aacct,ou=cluster1,cn=aixdata -a
          ou=adminpolicy,ou=aacct,ou=cluster1,cn=aixdata -r hourly
```

4. To display the currently configured accounting DNs, type:

```
mkprojldap -l
```

Files

Item	Description
`/usr/sbin/mkprojldap`	Contains the `mkprojldap` command.
`/etc/security/ldap/ldap.cfg`	Contains the LDAP configuration data.
`/etc/security/ldap/sec.ldif`	Contains the LDAP schema for advanced accounting.

mkproto Command

Purpose

Constructs a prototype file system.

Syntax

mkproto *Special Prototype*

Description

The **mkproto** command is used to construct a prototype for a new file system. It exists solely for Berkeley Software Distribution (BSD) compatibility.

The *Special* parameter can be a block device name, raw device name, or file system name. The *Prototype* parameter is the name of the prototype file that specifies the structure and contents of the file system to be created. The **mkproto** command calls the **mkfs** command with the *Prototype* and *Special* parameters.

Prototype Files

The **mkproto** and **mkfs** commands require an extended prototype file to create a Journaled File System (JFS). A *prototype file* is a formatted listing of the contents and structure of a file system. A prototype file describes the file system by a series of tokens separated by spaces and new lines. The main body of a prototype file defines the objects of the file system.

A JFS prototype file consists of the main body, which can be created by the **proto** command, preceded by five special tokens. These five tokens are defined as follows:

Prototype Files

Item	Description
1st token	Name of a file to be copied onto block 0 as the bootstrap program or the special token `<noboot>`.
2nd token	Size of the file system. For a JFS, the size is expressed in units of 512-byte blocks. If the 2nd token is 0, the **mkfs** command creates the file system to fill the entire logical volume.

Prototype Files *(continued)*

Item	Description
3rd token	Number of i-nodes on the file system. This token is not used by a JFS but must be provided to preserve the position.
4th token	Size of the file system fragment in bytes. If the 4th token is 0 (zero), the **mkfs** command uses the default fragment size. For JFS, the token must be either 0 (default value used), 512, 1024, 2048, or 4096. The default fragment size is 4096 for a JFS. An invalid fragment size causes the **mkfs** command to fail.
5th token	Number of bytes per i-node (nbpi). If this token is 0, the **mkfs** command uses the default nbpi. For a JFS, this token must be either 0 (default value used), 512, 1024, 2048, 4096, 8192, or 16384. The default number of bytes per i-node is 4096 for a JFS. An invalid nbpi causes the **mkfs** command to fail.

The remaining tokens define the contents and structure of the file system. These tokens are grouped into sets, with each set defining one object of the file system. The syntax of each set is as follows:

{ [*Name*] { **-** | { **-** | **d** | **b** | **c** | **l** | **L** | **p** } { **-** | **u** } { **-** | **g** } { **-** | **t** } *Mode Owner Group* { *Major Minor* | *SourceFile* | *DirectoryListing* } } | { **$** }

where:

Tokens

Item	Description
Name	Specifies the name of the object as it is to appear in the new file system. The *Name* token is required for every object except for the root directory definition.

Tokens *(continued)*

Item	Description
{ **-** | **d** | **b** | **c** | **l** | **L** | **p** } { **-** | **u** } { **-** | **g** } { **-** | **t** }	Represents a string of 4 positional characters, where: **{ - | d | b | c | l | L | p }** Defines the object type. Valid types are: **-** Regular file **d** Directory **b** Block special file **c** Character special file **l** Symbolic link **L** Hard link **p** Named pipe **{ - | u }** Toggles the set UID bit of the object, as follows: **u** Set UID on execution **-** Do not set UID on execution **{ - | g }** Toggles the set group ID (GID) bit of the object, as follows: **g** Set GID on execution **-** Do not set GID on execution **{ - | t }** Toggles the sticky bit of the object, as follows: **t** Sticky bit on **-** Sticky bit off This 4-character token is required for every object.
Mode	Represents a string of 3 octal characters defining the read, write, and execute permissions of the object. The *Mode* token is required of every object. See the **chmod** command for more information about permissions.
Owner	Specifies the UID of the owner of the object. The owner token is required for every object.

Tokens *(continued)*

Item	Description
Group	Specifies the GID of the owner of the object. The group token is required for every object.
Major Minor	Specifies the major and minor device numbers of the object if its type is a block or character special file. If the object is not a block or character special file, these tokens are omitted.
SourceFile	Applies only to regular file, hard link, and symbolic link objects. For regular files, this token is the path name to the file from which the object file is to be initialized. For both symbolic and hard links, this token is the source of the link. The source of the link is relative to the new file system for hard links.
DirectoryListing	Defines the contents of the object if it is a directory. The contents of the directory are defined using the token syntax described here. For example, a directory listing can include one or more regular files, one or more block files, and one or more directory listings. The **mkfs** command creates the directory entries . (dot) and .. (dot dot). Each directory listing is terminated with the special **$** token.
$	Ends the current directory listing or indicates the end of the prototype file.

Example Prototype Specification

The following prototype specification describes a JFS that does not have a boot program in block 0 and occupies the entire device. The 3rd token is ignored. The 4th and 5th tokens define the fragment size as 1024 bytes and the number of bytes per i-node as 2048. The main body of this prototype defines the file system contents.

```
<noboot> 0 0 1024 2048
d---  755 0 0
 dir1    d--- 755 0 2
  block_dev   b--- 644 0 0   880 881
  char_dev    c--- 644 0 0   990 991
  named_pipe  p--- 644 0 0
  regfile3    ---- 644 0 0   /tmp/proto.examp/dir1/regfile3
  regfile4    ---- 644 0 0   /tmp/proto.examp/dir1/regfile4
  $
 dir2    d--- 755 205 300
  regfile6   ---- 644 0 0    /tmp/proto.examp/dir2/regfile6
  symlnOutofFS l--- 644 0 0  /tmp/proto.examp/dir2/regfile6
  symlnNoExist l--- 644 0 0  /home/foobar
  symlnInFs  l--- 644 0 0    /dir2/regfile6
  regfile5   ---- 644 0 0    /tmp/proto.examp/dir2/regfile5
  hardlink   L--- 644 0 0    /dir2/regfile5
  $
 dir3   d--- 755 0 0
  setgid     --g- 755 0 0    /tmp/proto.examp/dir3/setgid
  setuid     -u-- 755 0 0    /tmp/proto.examp/dir3/setuid
  sticky     ---t 755 0 0    /tmp/proto.examp/dir3/sticky
  $
 dir4   d--- 755 0 0
  dir5  d--- 755 0 0
   dir6 d--- 755 0 0
    $
   dir7 d--- 755 0 0
    $
   $
  regfile7   ---- 644 0 0    /tmp/proto.examp/dir4/regfile7
  $
```

```
 regfile1    ---- 555 205 1  /tmp/proto.examp/regfile1
 regfile2    ---- 744 0 0    /tmp/proto.examp/regfile2
 $
$
```

Three entries for the dir2 object deserve further examination:

dir2 object

Item	Description
symlnOutofFS l—- 644 0 0 /tmp/proto.examp/dir2/regfile6	This entry defines a symbolic link to a file outside the file system to be created. The command ls -l lists something similar to symlnOutofFS -> /tmp/proto.examp/dir2/regfile6.
symlnNoExist l—- 644 0 0 /home/foobar	This entry defines a symbolic link to a file outside the file system to be created to a file that does not exist. The command ls -l lists something similar to symlnNoExist -> /home/foobar.
symlnInFs l—- 644 0 0 /dir2/regfile6	This entry defines a symbolic link to a file within the file system to be created. The command ls -l lists something similar to symlnInFS -> /dir/regfile6.

Examples

To make a prototype JFS using the prototype file described in the "Example Prototype File Specification" :

1. Generate the main body of the prototype file using the **proto** command or a text editor. For the purposes of this example, call the file /tmp/ProtoFile.
2. Add the first 5 tokens as required for a JFS. In the example prototype file, the tokens are:

   ```
   <noboot> 0 0 1024 2048
   ```

3. Create a logical volume to hold the file system, as follows:

   ```
   mklv -y protolv -t jfs SomeVGname 5
   ```

   This command creates a logical volume named protolv in the SomeVGname volume group. The size of the logical volume is 5 logical partitions.
4. Add an appropriate stanza to the /etc/filesystem file. A minimal example stanza is:

   ```
   /protofs:
     dev             = /dev/protolv
     vfs             = jfs
     log             = /dev/loglv00
     mount           = false
   ```

5. Run the following **mkproto** command:

   ```
   mkproto /dev/protolv /tmp/ProtoFile
   ```

   This command creates a JFS on the protolv logical volume. The size of the JFS is 5 logical partitions, its fragment size is 1024 bytes, and its nbpi ratio is 2048. The structure and contents of the file system are as specified in the prototype file /tmp/ProtoFile.

Files

Files

Item	Description
/usr/sbin/mkproto	Contains the **mkproto** command.

mkprtldap Command

Purpose

Configures IBM Directory (LDAP) for Directory enabled System V print. It also configures client machines to use the Directory for System V print information.

Syntax

To configure the IBM Directory to store System V Print information:

mkprtldap **-s** **-a** *AdminDN* **-p** *Adminpasswd* **-w** *ACLBindPasswd* [**-f**] [**-d** *nodeDN*]

To configure clients to use the IBM Directory for System V Print information:

mkprtldap **-c** **-h** *DirectoryServerHostname* **-w** *ACLBindPasswd* [**-d** *PrintBindDN*] [**-U**]

To get usage information for the **mkprtldap** command:

mkprtldap **?**

Description

The **mkprtldap** command configures the IBM Directory (LDAP) server, and one or more clients to use the Directory (LDAP) for System V Print information. This command must be run on the system being setup as the server and on all the client systems. Once the Directory (LDAP) server is configured for System V print, the directory enabled System V Print commands (**dslpadmin**, **dslpaccess**, **dslpsearch**, **dslpenable**, **dslpdisable**, **dslpaccept**, **lp**, **lpstat**, **cancel** and **dslpreject**) must be run to add , remove and manage System V print information (printers and print queues) on the Directory (LDAP) server. The **mkprtldap** command configures client machines to use the Directory (LDAP) server for System V print information.

The **mkprtldap** command requires the IBM Directory server software to be installed on the machine being configured as the server. The command also requires the IBM Directory client software to be installed on all client machines that will use the Directory (LDAP) server for System V print information.

Note: The client (**-c** flag) and the server (**-s** server) options cannot be run at the same time. When setting up a system as the server, the mkprtldap command should be run twice on that system. Once to set up the server, and again to set up the client.

During the server side configuration, using the **-s** flag, the **mkprtldap** command:

- Requires the IBM Directory Administrator's DN and password if the Directory has been configured. If the Directory Administrator's DN and password have not been set, **mkprtldap** will set them with the values passed to the command.
- Creates the AIX Information tree DN (cn=aixdata container object) on the Directory if one is not present. The print subtree will be created under the AIX Information subtree. If an existing AIX Information subtree exists on the Directory, the print subtree will be created under it. All System V print information will be stored under the print subtree. The directory enabled System V print commands have to be run to add printers and print queues under the print subtree created.
- The default suffix and AIX Information tree for the **mkprtldap** command is a top level container object cn=aixdata. The Print subtree (ou=print) will be created under the AIX Information tree.
- The print subtree is ACL protected with the value of the **ACLBindPasswd** parameter passed to the command. The same value must be used when configuring clients to use the Directory for System V

print information. Select a password value that is difficult for people or password cracking programs to guess.

- If the **-d** option is used and a valid existing node on the Directory is passed to the command, the AIX Information subtree is created under the given node. The print subtree is then created under the AIX Information subtree.
- Starts the IBM Directory server after all the above is done
- Adds the IBM Directory server process (slapd) to the **/etc/inittab** file to have the server start after a reboot.

During the client configuration, the **mkprtldap** command:

- Saves the IBM Directory (LDAP) server host name in the **/etc/ldapsvc/server.print** file.
- Saves the AIX Print Bind DN in the **/etc/ldapsvc/server.print** file.
- Saves the ACL Bind Password for the AIX Print Bind DN in the **/etc/ldapsvc/system.print** file. The value of the ACL Bind password must be the same as the one specified during the configuration of the Directory server.
- Undo a previous client configuration if the **-U** flag is specified. This option will replace the **/etc/ldapsvc/system.print** and **/etc/ldapsvc/server.print** files with the previous saved copies of the files (**/etc/ldapsvc/server.print.save** and **/etc/ldapsvc/system.print.save**).

Flags

Server

Item	Description
-a *AdminDN*	Specifies the Directory (LDAP) Administrator's DN.
-d *nodeDN*	This advanced option requires a valid existing node DN on the Directory under which the AIX Information tree and Print Subtree will be created.
-f	The force flag is required by the **mkprtldap** command to force the creation of the Print subtree (and AIX Information subtree if needed) when one or more AIX Information trees exist on the Directory.
-p *adminpasswd*	Specifies the Directory (LDAP) Administrator's password.
-s	Indicates the command is being run to configure the Directory for System V print.
-w *ACLBindPasswd*	Specifies the password to ACL protect the Print Subtree on the Directory. Select a password value that is difficult for people or password cracking programs to guess.

Client

Item	Description
-c	Indicates the command is being run to configure clients to use the Directory for System V Print information.
-d *PrintBindDN*	Specifies the Print Bind DN. The default Print Bind DN is **ou=print,cn=aixdata**. The Print Bind DN to use during Client configuration is displayed at the end of the server setup of the **mkprtldap** command.
-h *DirectoryServerHostname*	Hostname of the IBM Directory server setup to store System V Print information.
-U	Undo a previous configuration of a client.

Item	Description
-w *ACLBindPasswd*	The ACL Bind Password for the print subtree. The ACL Bind password is specified during the server setup of the **mkprtldap** command. The value of the ACL Bind Password must match the one used during the setup of the Directory server.

Usage

Item	Description
?	Displays usage information for the **mkprtldap** command.

Security

This command can be run by the `root` user only.

Examples

1. To configure System V print on a machine with a configured IBM Directory server -

   The Administrator DN and password are required to configure System V print on the Directory. Assume the existing Administrator's DN and password are **cn=admin** and **passwd**.

   ```
   mkprtldap -s -a cn=admin -p passwd -w pass123wd
   ```

2. The **mkprtldap** command provides the option to configure the IBM Directory to store the print information under a pre-existing node (e.g. **o=ibm**,**c=us**) on the Directory [Advanced Option]. This is only recommended when it is necessary to store the print information under the existing node on the Directory for specific reasons. The recommend option is to store the print subtree in the default location on the Directory by not specifying the **-d** option. The Administrator DN and password are required to configure System V print on the Directory Assume the existing Administrator's DN and password are **cn=admin** and **passwd**.

   ```
   mkprtldap -a cn=admin -p passwd -w acl123passwd -d o=ibm,c=us
   ```

   Running the command will create an AIX Information tree (**cn=aixdata**) under the **o=ibm,c=us** object. The print subtree will be created under this new object (**cn=aixdata**, **o=ibm**, **c=us**).

3. To configure System V print on a machine with a configured IBM Directory server and an existing AIX Information tree. There might be situations where the Directory contains an existing AIX information tree with other subsystem specific information (e.g Security or NIS information). It might be required to store the print information in a separate location on the Directory under a different AIX Information tree. The command, by default, will not create a new AIX Information tree if one exists on the Directory. To force the command to create a new AIX Information tree to store the print information, use the **-f** flag with the command. Consider the case where the Security and NIS subsystem information is stored under the AIX Information tree at **cn=aixdata,o=ibm,c=us**. To create a new AIX Information tree for print information different from the existing one, run the command with the **-f** flag and specify the default location or another node. The Administrator DN and password are required to configure System V print on the Directory. Assume the existing Administrator's DN and password are **cn=admin** and **passwd**:

   ```
   mkprtldap -a cn=admin -p passwd -w passwd123 -f
   ```

   Running the command will create a new AIX Information tree (**cn=aixdata**) with the suffix (**cn=aixdata**) and the print information will be stored under this new AIX Information tree (**ou=print**, **cn=aixdata**). There will be two AIX Information trees on the Directory in this example **cn=aixdata,o=ibm,c=us** and **cn=aixdata**. The print information will be under the **cn=aixdata** object (suffix - **cn=aixdata**). For **mkprtldap**, it is recommend to use the default location to add the print information to the Directory.

4. To configure a client to use an IBM Directory setup for System V Print on host **server.ibm.com**, type:

   ```
   mkprtldap -c -h server.ibm.com -w passwd
   ```

   Please ensure that the ACL Bind Password (**passwd**) is the same as the one specified during the setup of the Directory Server. Running the command without specifying a Print Bind DN value with the -d option will cause the command to use the default Print Bind DN **ou=print,cn=aixdata**. The Print Bind DN must match the one displayed at the end of running the **mkprtldap** command to configure the server.
5. To change the information in the client side configuration files, run the **mkprtldap** command with the new information

   ```
   mkprtldap -c -h server.ibm.co.uk -w aclpasswd -d ou=print,cn=aixdata,c=uk
   ```

   Executing this command on a client that has already been configured will change the information in the **/etc/ldapsvc/server.print** and **/etc/ldapsvc/system.print** files to contain the new configuration information. The original contents of the **/etc/ldapsvc/server.print** and **/etc/ldapsvc/system.print** will stored in the **/etc/ldapsvc/server.print.save** and **/etc/ldapsvc/system.print.save** files.

Files

Mode	File	Description
rw	**/etc/slapd32.conf**	(Server configuration) - Contains the IBM Directory (LDAP Version 5.2) configuration information.
rw	**/home/ldapdb2/idsslapd-ldapdb2/etc/ibmslapd.conf**	(Server configuration) - Contains the IBM Directory (LDAP Version 6.0 or later) configuration information.
rw	**/etc/ldapsvc/server.print**	(Client configuration) - Contains information about the Directory Server configured to store System V Print information. (Machine name, Location of Print subtree on the Directory and LDAP port)
rw	**/etc/ldapsvc/system.print**	(Client configuration) - Contains the ACL Bind Password for the Print subtree on the Directory.

mkprtsv Command

Purpose

Configures TCP/IP-based print service on a host.

Syntax

To Configure and Start Print Service for a Client Machine

mkprtsv **-c** [**-S**][**-q** *QueueName* **-v** *DeviceName* **-b** "*Attribute* =*Value* ..." **-a** "*Attribute* =*Value* ..." | **-A** *FileName*]

To Configure and Start Print Service for a Server Machine

mkprtsv **-s** [**-S**] [**-q** *QueueName* **-v** *DeviceName* **-b** "*Attribute* =*Value* ..." **-a** "*Attribute* =*Value* ..." | **-A** *FileName*] [**-h** "*HostName* ..." | **-H** *FileName*]

Description

The **mkprtsv** high-level command configures a TCP/IP-based print service on a host. The print service configuration can be done for a host functioning as a client or for a host functioning as a server.

Use the command to configure and start the print service.

To configure print service for a client, the **mkprtsv** command calls the spooler **mkque** and **mkquedev** commands to change the **/etc/lpd/qconfig** file (or its object class equivalent) appropriately and set up a spooler queue on the client machine.

To configure print service for a server, the **mkprtsv** command does the following:

1. Calls the **ruser** command to set up remote users to print on the server.
2. Calls the **mkque** and **mkquedev** commands to change the server's **/etc/lpd/qconfig** file appropriately and set up the necessary device queues on the server machine.
3. Calls the **startsrc** command to activate the **lpd** and **qdaemon** server daemons. The **qdaemon** server daemon starts the **piobe** printer backend.

Flags

Item	Description
-A *FileName*	Specifies name of file containing entries related to the **qconfig** file.
-a "*Attribute* **=***Value...***"**	Specifies a list of attributes and their corresponding values to be used for updating the spooler's **qconfig** file or object class. The **-a** flag is optional. Valid attribute types are listed below: **acctfile (true/false)** Identifies the file used to save **print** command accounting information. The default value of **false** suppresses accounting. If the named file does not exist, no accounting is done. **argname** Specifies the logical printer name. **device** Identifies the symbolic name that refers to the device stanza. **discipline** Defines the queue-serving algorithm. The default value of **fcfs** means first come, first served. A **sjn** value means shortest job next. **pserver** Specifies the remote print server. **up (true/false)** Defines the state of the queue. The default value of **true** indicates that it is running. A **false** value indicates that it is not.

Item	Description
-b "*Attribute* **=***Value...***"**	Specifies a list of attributes and their corresponding values to be used for updating the spooler's **qconfig** file or object class. At least one attribute must be defined for the **-b** option. The **backend** attribute is required. Valid attribute types are listed below: **access (true/false)** Specifies the type of access the backend has to the file specified by the **file** attribute. The **access** attribute has a value of **write** if the backend has write access to the file, or a value of **both** if the backend has both read and write access. This field is ignored if the file field has a value of **false**. **align (true/false)** Specifies whether the backend sends a form-feed control before starting the job if the printer has been idle. The default value is **false**. **backend** Specifies the full path name of the backend, optionally followed by flags and parameters to be passed to it. The **backend** attribute is required. **feed** Specifies the number of separator pages to print when the device becomes idle, or takes a **never** value, which indicates that the backend is not to print separator pages. **file** Identifies the special file where the output of the backend is to be redirected. The default value of **false** indicates no redirection. In this case, the backend opens the output file. **header (never/always/group)** Specifies whether a header page prints before each job or group of jobs. The default value of **never** indicates no header page. To produce a header page before each job, specify an **always** value. To produce a header before each group of jobs for the same user, specify a **group** value. **trailer (never/always/group)** Specifies whether a trailer page prints after each job or group of jobs. The default value of **never** indicates no trailer page. To produce a trailer page after each job, specify an **always** value. To produce a trailer after each group of jobs for the same user, specify a **group** value. **host** Specifies the host name from which to print. **s_statfilter** Translates short queue-status information to a format recognized by this operating system. **l_statfilter** Translates long queue-status information to a format recognized by this operating system.
-c	Performs print service configuration for a client machine. Use the **-q** flag with the **-c** option.
-H *FileName*	Specifies the name of a file containing a list of host names.
-h "*HostName...***"**	Specifies a list of host names to be included in the list of remote users who can use the print server. The queuing system does not support multibyte host names.
-q *QueueName*	Specifies the name of a queue in the **qconfig** file.

Item	Description
-S	Starts print service after it is configured. If the **-S** flag is omitted, print service is configured but not started.
-s	Performs print service configuration for a server machine. Use the **-h**, **-H**, and **-q** flags with the **-s** flag.
-v *DeviceName*	Specifies the name of the device stanza in the **qconfig** file.

Examples

1. To configure and enable print service for a client, enter the command in the following format:

   ```
   mkprtsv -c -S -a"argname=rp1 backend=piobe \
   pserver=print802"
   ```

   In this example, `rp1` is the logical printer name, `piobe` is the printer backend, and `print802` is the remote print server.
2. To configure a print server using initialization information and allow remote printing, enter the command in the following format:

   ```
   mkprtsv -s -H hnames -A qinfo
   ```

   In this example, attribute information stored in the `qinfo` file initializes the spooler, and the list of host names stored in the `hnames` file is the list of remote hosts that have access rights to the print server.

Files

Item	Description
/etc/lpd/qconfig	Contains configuration information for the printer queuing system.

mkps Command

Purpose

Adds an additional paging space.

Syntax

To Add a Logical Volume for Additional Paging Space

mkps [**-t** *lv* | *[ps_helper psname]*] [**-a**] [**-n**] [**-c** *ChksumSize*]**-s** *LogicalPartitions VolumeGroup* [*PhysicalVolume*]

To Add Additional Paging Space On an NFS Server

mkps [**-a**] [**-n**] **-t** *nfs ServerHostName ServerFileName*

Description

The **mkps** command adds an additional paging space. Before the paging space can be used it must be activated, using the **swapon** command. The *VolumeGroup* parameter specifies the volume group within which the logical volume for the paging space is to be made. The *PhysicalVolume* parameter specifies the physical volume of the *VolumeGroup* on which the logical volume is to be made.

Note: A paging space larger than 2 GB is possible when using NFS (Network File System) v4 rather than the default UDP (User Datagram Protocol) or the NFSv2 protocol. NFSv2 swapping can only handle swap file size up to 2 GB due to NFSv2 protocol limitation.

In the second form of the **mkps** command, the *ServerHostName* parameter specifies the NFS server where the *ServerFileName* resides. The *ServerFileName* specifies the file which will be used for the NFS paging of the system. The *ServerFileName* file must exist and be exported correctly to the client that will use the file for paging.

When adding a NFS paging space, the client attempts to contact the server using UDP and then TCP. The method that succeeds first is used to contact the server when accessing that paging space.

If the **-t** flag is specified, the argument will be assumed to be a third-party helper executable. If the helper executable is present in the /sbin/helpers/pagespace path then it will be spawned passing all the arguments and with the **-m** flag to specify **mkps** command. An entry will be added into /etc/swapspaces path if the helper executable returns zero. In this case, if *psname* starts with /, it is considered to be absolute path of device entry , or else /dev is prepended to the *psname*. The helper executable must take care of creating the device, making it pageable and adding an entry into ODM. If the helper program doesn't exist in the /sbin/helpers/pagespace directory the **mkps** command will display the usage error. The helper executable must exit with a 0 if successful and a non-zero if it fails.

You can use the System Management Interface Tool (SMIT) **smit mkps** fast path to run this command.

Flags

Item	Description
-a	Specifies that the paging space is configured at subsequent restarts.
-c	Specifies the size of the checksum to use for the paging space, in bits. Valid options are 0 (checksum disabled), 8, 16 and 32. If -c is not specified it will default to 0.
-n	Activates the paging space immediately.
-s *LogicalPartitions*	Specifies the size of the paging space and the logical volume to be made in logical partitions.
-t	Specifies the type of paging space to be created. One of the following variables is required: ***lv*** Specifies that a paging space of type logical volume should be created on the system. ***nfs*** Specifies that a paging space of type NFS should be created on the system. ***ps_helper*** Name of the helper program for a third party device. ***psname*** Name of the device entry for paging space.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To create a paging space in volume group `myvg` that has four logical partitions and is activated immediately and at all subsequent system restarts, enter:

   ```
   mkps  -a  -n  -s4 myvg
   ```

2. To create an NFS paging space on the NFS server `swapserve` where the host `swapserve` has the **/export/swap/swapclient** file exported, enter:

```
mkps -t nfs swapserve /export/swap/swapclient
```

3. To create a paging space myps using helper executable foo:

```
mkps -t foo /dev/myps -s1 myvg mydisk
```

Files

Item	Description
/etc/swapspaces	Specifies the paging space devices and their attributes.

mkqos Command

Purpose

Configures the system to support QoS.

Syntax

/usr/sbin/mkqos [**-I** | **-N** | **-B**]

Description

The **mkqos** command configures the system to support Quality of Service (QoS).

Flags

Item	Description
-B	Adds an entry to the **inittab** file to execute the **/etc/rc.qos** file now and on the next system restart. This flag is the default.
-I	Adds an entry to the **inttab** file to execute the **/etc/rc.qos** file on the next system restart.
-N	Executes the **/etc/rc.qos** file to start the QoS daemons. When invoked in this way, the QoS daemons run until the next system restart.

Files

Item	Description
inittab	Controls the initialization process of the system.
/etc/rc.qos	Contains the startup script for the QoS daemons.

mkque Command

Purpose

Adds a printer queue to the system.

Syntax

mkque [**-D**] **-q** *Name* [**-a** '*Attribute* **=** *Value*' ...]

Description

The **mkque** command adds a printer queue to the system by adding the stanza described on the command line to the end of the **/etc/qconfig** file.

You can use the System Management Interface Tool (SMIT) **smit mkque** fast path to run this command.

To use the SMIT fast path to go directly to the **Add a Local Queue** dialog, enter:

```
smit mklque
```

To use the SMIT fast path to go directly to the **Add a Remote Queue** dialog, enter:

```
smit mkrque
```

Recommendation: To edit the **/etc/qconfig** file, use the **chque**, **mkque**, **rmque**, **chquedev**, **mkquedev**, and **rmquedev** commands or SMIT. Further, it is recommended to run these commands during slow or off-peak time.

If manual editing of the **/etc/qconfig** file is necessary, you can first issue the **enq -G** command to bring the queuing system and the **qdaemon** to a halt after all jobs are processed. Then you can edit the **/etc/qconfig** file and restart the **qdaemon** with the new configuration.

Flags

Item	Description
-a *'Attribute* **=** *Value'*	Specifies a line to be added to the queue stanza in the **/etc/qconfig** file. This flag must be the last flag when entering the **mkque** command on the command line. For a list of all valid attributes, see the **/etc/qconfig** file. **Note:** It is recommended that you do not use the `'device = '` attribute. This attribute is handled automatically by the **mkquedev** command. Also note that the queuing system does not support multibyte host names.
-D	Specifies that the queue defined by the *Name* variable queue is added to the top of the **/etc/qconfig** file and is therefore the default queue. If you do not specify this flag, the *Name* variable is added to the bottom of the **/etc/qconfig** file and is not the default queue.
-q *Name*	Specifies the name of the queue to be added. **Note:** The queue name must not exceed 20 characters.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To add the print queue `lp0` specifying a host name of `leo` and a remote print queue named `lp013`, enter:

```
mkque  -qlp0  -a 'host = leo' -a 'rq = lp013'
```

Files

Item	Description
/usr/bin/mkque	Contains the **mkque** command.
/etc/qconfig	Configuration file.

mkquedev Command

Purpose

Adds a printer queue device to the system.

Syntax

mkquedev -d *Name* **-q** *Name* **-a '***Attribute* **=** *Value***'** ...

Description

The **mkquedev** command adds a printer queue device to the system by adding the stanza described on the command line to the **/etc/qconfig** file.

You can use the System Management Interface Tool (SMIT) **smit mkquedev** fast path to run this command.

Recommendation: To edit the **/etc/qconfig** file, use the **chque**, **mkque**, **rmque**, **chquedev**, **mkquedev**, and **rmquedev** commands or SMIT. Further, it is recommended to run these commands during slow or off-peak time.

If manual editing of the **/etc/qconfig** file is necessary, you can first issue the **enq -G** command to bring the queuing system and the **qdaemon** to a halt after all jobs are processed. Then you can edit the **/etc/qconfig** file and restart the **qdaemon** with the new configuration.

Flags

Item	Description
-a '*Attribute* **=** *Value***'**	Specifies the **'***Attribute* **=** *Value***'** attribute to be added to the device stanza in the **/etc/qconfig** file. This flag must be the last flag when entering the **mkquedev** command on the command line. For a list of valid attributes, see the **/etc/qconfig** file. **Note:** The **'backend = '** attribute must be included when entering this command on the command line.
-d *Name*	Specifies with the *Name* variable the name of the queue device to add. **Note:** The queue device name must not exceed 20 characters.

Item	Description
-q *Name*	Specifies with the *Name* variable the name of the queue (this name must already exist) to which the queue device is added. The **mkquedev** command automatically adds the **'device = '** attribute to the specified queue stanza.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To add the `postscript` print queue device to the `lp0` queue, specify the backend program to be the **piobe** command (`backend = /usr/lib/lpd/piobe`) and direct the backend program not to align the paper (`align = FALSE`), enter:

```
mkquedev -qlp0 -dpostscript -a 'backend = /usr/lib/lpd/piobe' \
 -a 'align = FALSE'
```

Files

Item	Description
/usr/bin/mkquedev	Contains the **mkquedev** command.
/etc/qconfig	Contains the configuration file.

mkramdisk Command

Purpose

Creates a RAM disk using a portion of RAM that is accessed through normal reads and writes.

Syntax

mkramdisk [**-u**] *size*[**M** | **G**]

Description

The **mkramdisk** command is shipped as part of **bos.rte.filesystems**, which allows the user to create a RAM disk. Upon successful execution of the **mkramdisk** command, a new RAM disk is created, a new entry added to **/dev**, the name of the new RAM disk is written to standard output, and the command exits with a value of 0. If the creation of the RAM disk fails, the command prints an internalized error message, and the command will exit with a nonzero value.

The size can be specified in terms of MB or GB. By default, it is in 512 byte blocks. A suffix of *M* will be used to specify size in megabytes and *G* to specify size in gigabytes.

The names of the RAM disks are in the form of **/dev/rramdisk***x* where x is the logical RAM disk number (0 through 63).

The **mkramdisk** command also creates block special device entries (for example, **/dev/ramdisk5**) although use of the block device interface is discouraged because it adds overhead. The device special files in **/dev** are owned by root with a mode of 600. However, the mode, owner, and group ID can be changed using normal system commands.

Up to 64 RAM disks can be created.

Note: The size of a RAM disk cannot be changed after it is created.

The **mkramdisk** command is responsible for generating a major number, loading the ram disk kernel extension, configuring the kernel extension, creating a ram disk, and creating the device special files in **/dev**. Once the device special files are created, they can be used just like any other device special files through normal **open**, **read**, **write**, and **close** system calls.

RAM disks can be removed by using the **rmramdisk** command. RAM disks are also removed when the machine is rebooted.

By default, RAM disk pages are pinned. Use the -u flag to create RAM disk pages that are not pinned.

Flags

Item	Description
-u	Specifies that the ram disk that is created will not be pinned. By default, the ram disk will be pinned.

Parameters

Item	Description
size	Indicates the amount of RAM (in 512 byte increments) to use for the new RAM disk. For example, typing: `mkramdisk 1` creates a RAM disk that uses 512 bytes of RAM. To create a RAM disk that uses approximately 20 MB of RAM, type: `mkramdisk 40000`

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To create a new ram disk using a default 512-byte block size, and the size is 500 MBs (1048576 * 512), enter:

   ```
   mkramdisk 1048576
   /dev/rramdisk0
   ```

   The **/dev/rramdisk0** ramdisk is created.

2. To create a new ramdisk with a size of 500 Megabytes, enter:

```
mkramdisk 500M
/dev/rramdisk0
```

   The **/dev/rramdisk0** ramdisk is created. Note that the ramdisk has the same size as example 1 above.

3. To create a new ram disk with a 2-Gigabyte size, enter:

```
mkramdisk 2G
/dev/rramdisk0
```

4. To set up a RAM disk that is approximately 20 MB in size and create a JFS file system on that RAM disk, enter the following commands:

```
mkramdisk 40000
ls -l /dev | grep ram
mkfs -V jfs /dev/ramdiskx
mkdir /ramdisk0
mount -V jfs -o nointegrity /dev/ramdiskx /ramdiskx
```

   x is the logical RAM disk number.

   To set up a RAM disk that is approximately 20 MB in size and create a JFS2 file system on that RAM disk, enter the following commands:

```
mkramdisk 40000
ls -l /dev | grep ram
/sbin/helpers/jfs2/mkfs -V jfs2 /dev/ramdiskx
mkdir /ramdiskx
mount -V jfs2 -o log=NULL /dev/ramdiskx /ramdiskx
```

   x is the logical RAM disk number.

   Note: For both JFS and JFS2, for using a file system on a RAM disk, the RAM disk must be pinned.

Files

Item	Description
/usr/sbin/mkramdisk	Contains the **mkramdisk** command.

mkresponse Command

Purpose

Creates a new response definition.

Syntax

To create a response with no actions:

mkresponse [**-b**] [-p *node_name*] [-h] [-TV] *response*

To create a response with one action:

mkresponse -n *action* [**-d** *days_of_week*[**,***days_of_week*...]] [**-t** *time_of_day*[**,***time_of_day*...]] **-s** *action_script* [**-r** *return_code*] [**-b** | [**-e a** | **A** | **b** | **e** | **r**]] [**-o**] [**-E** *env_var*=*value*[**,***env_var*=*value*...]] [**-u**] [**-p** *node_name*] [**-h**] [**-TV**] *response*

To copy a response:

mkresponse -c *existing_response*[**:***node_name*] [-p *node_name*] [-h] [-TV] *response*

Description

The `mkresponse` command creates a new response definition with the name specified by the *response* parameter. One action can also be specified when the response is defined. Actions define commands to be run when the response is used with a condition and the condition occurs. The action defines days of the week when the action can be used, the time of day for those days of the week, the script or command to be run, what type of event causes the command to be run, the expected return code of the script or command, and whether to keep standard output. The days and times are paired so that different times can be specified for different days. A response with no actions only logs the events.

Use the **-b** flag to specify that the response, and all actions to be defined in this response, support event batching. For event batching, multiple events can be batched or grouped together and passed to a response. The actions of the response are directed to a file that contains the details for the batched events. A response that supports event batching can only be used for conditions that specify the events are to be batched. The **-b** flag cannot be specified with the **-e** flag.

In a cluster environment, use the `-p` flag to specify the node in the domain that is to contain the response definition. If you are using `mkresponse` on the management server and you want the response to be defined on the management server, do *not* specify the `-p` flag. If the `-p` flag is not specified, the response is defined on the local node.

Use the **chresponse** command to add actions to a response or to remove actions from a response. Use the **startcondresp** command to start monitoring. The **startcondresp** command links a response to a condition, if they are not already linked.

To lock a response so it cannot be modified or removed, use the `chresponse` command with the `-L` flag.

Flags

-b

Specifies that the response, and all actions to be defined in this response, support event batching. For event batching, multiple events can be batched or grouped together and passed to a response. The actions of the response are directed to a file that contains the details for the batched events. A response that supports event batching can only be used for conditions that specify the events are to be batched.

An event response can be created for batched event conditions without an action script.

The **-b** flag cannot be specified with the **-e** flag.

-c *existing_response*[:*node_name*]

Copies an existing response. Links with conditions are not copied. The existing response is defined on the node known as *node_name* in a cluster. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable. If any other flags are specified, update the new response as indicated by the flags.

-d *days_of_week*

Specifies the days of the week when the action being defined can be run. *days_of_week* and *time_of_day* together define the interval when the action can be run.

Enter the numbers of the days separated by a plus sign (+) or as a range of days separated by a hyphen (-). More than one *days_of_week* parameter can be specified, but the parameters must be separated by a comma (,). The number of *days_of_week* parameters specified must match the number of *time_of_day* parameters specified. The default is all days. If no value is specified but a comma is entered, the default value is used. The values for each day follow:

1
Sunday

2
Monday

3
Tuesday

4
Wednesday

5
Thursday

6
Friday

7
Saturday

-e a | A | b | e | r
Specifies the type of event that causes the action being defined to run:

a
Specifies an event. This is the default.

A
Specifies any type of event (event, error event, or rearm event).

b
Specifies an event and a rearm event.

e
Specifies an error event.

r
Specifies a rearm event.

More than one event type can be specified, for example: **-e ae**. The **-e** flag cannot be specified with the **-b** flag.

-E *env_var*=*value*[,*env_var*=*value*...]
Specifies any environment variables to be set before running the action. If multiple *env_var=value* variables are specified, they must be separated by commas.

-n *action*
Specifies the name of the action being defined. Only one action can be defined when the response is created. Use the `chresponse` command to add more actions to the response.

-o
Directs all standard output from *action_script* to the audit log. The default is not to keep standard output. Standard error is always directed to the audit log.

-p *node_name*
Specifies the name of the node where the response is defined. This is used in a cluster environment and the node name is the name by which the node is known in the domain. The default *node_name* is the local node on which the command runs. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

If you are using `mkresponse` on the management server and you want the response to be defined on the management server, do *not* specify the `-p` flag.

-r *return_code*
Specifies the expected return code for *action_script*. If the expected return code is specified, the actual return code of *action_script* is compared to the expected return code. A message is written to the audit log indicating whether they match. If the `-r` flag is not specified, the actual return code is written to the audit log, and no comparison is performed.

-s *action_script*
Specifies the fully-qualified path for the script or command to run for the action being defined. See the `logevent`, `notifyevent`, and `wallevent` commands for descriptions of the predefined response scripts provided with the application.

-t *time_of_day*
Specifies the time range when *action* can be run, consisting of the start time followed by the end time, separated by a hyphen. *days_of_week* and *time_of_day* together define the interval when the action can be run.

The time is in 24-hour format (HHMM) where the first two digits represent the hour and the last two digits represent the minutes. The start time must be less than the end time because the time is specified by day of the week. More than one *time_of_day* parameter can be specified, but the parameters must be separated by a comma (,). The number of *days_of_week* parameters specified must match the number of *time_of_day* parameters specified. The default value is 0000-2400. If no value is specified but a comma is entered, the default value is used.

-u
Specifies that the action is to be run when a monitored resource becomes undefined.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

response
The *response* name is a character string that identifies the response. If the name contains spaces, it must be enclosed in quotation marks. A name cannot consist of all spaces, be null, or contain embedded double quotation marks.

Security

The user needs write permission for the `IBM.EventResponse` resource class to run `mkresponse`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *Administering RSCT* guide for details on the ACL file and how to modify it.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To define a response with the name "Log event in audit log", run this command:

```
mkresponse "Log event in audit log"
```

2. To define a response with the name "E-mail root anytime" that has an action named "E-mail root", to be used any time Saturday and Sunday and uses the command **/opt/rsct/bin/notifyevent root** for both events and rearm events, run this command:

```
 mkresponse -n "E-mail root" -d 1+7  \
-s "/opt/rsct/bin/notifyevent root" -e b   \
"E-mail root anytime"
```

3. To define a response with the name "E-mail root anytime" that has an action named "E-mail root", to be used anytime Saturday and Sunday but only 8 am to 5 pm Monday through Friday and that uses the command **/opt/rsct/bin/notifyevent root** for events, run this command:

```
mkresponse -n "E-mail root"     \
-d 1+7,2-6  -t 0000-2400,0800-1700  \
-s "/opt/rsct/bin/notifyevent root" -e a  \
"E-mail root anytime"
```

4. To define a response with the name "E-mail root anytime" that has an action named "E-mail root" to be used any time Saturday and Sunday, that uses the command **/opt/rsct/bin/notifyevent root** for both events and rearm events, and that sets the environment variable LANG to en_US, run this command:

```
mkresponse -n "E-mail root" -d 1+7  \
-s "/opt/rsct/bin/notifyevent root" -e b \
-E LANG="en_US"  "E-mail root anytime"
```

5. To define a response with the name "E-mail root first shift" that has an action named "E-mail root" to be used Monday through Friday from 8 am to 6 pm, that uses the command **/opt/rsct/bin/notifyevent root** for rearm events, and that saves standard output in the audit log, expecting return code 5, run this command:

```
mkresponse -n "E-mail root" -d 2-6  -t 0800-1800  \
-s "/opt/rsct/bin/notifyevent root" -e r  -o   \
-r 5  "E-mail root first shift"
```

6. To define a response with the name "Critical notifications" as a copy of "Warning notifications", enter:

```
mkresponse -c "Warning notifications" "Critical notifications"
```

7. To define a batching-capable response called "Batched Event Response" without an action script, enter:

```
mkresponse -b "Batched Event Response"
```

These examples apply to management domains:

1. To define a response on the management server with the name "E-mail root anytime" that has an action named "E-mail root", to be used any time Saturday and Sunday and uses the command **/opt/rsct/bin/notifyevent root** for both events and rearm events, run this command on the management server:

```
 mkresponse -n "E-mail root" -d 1+7  \
-s "/opt/rsct/bin/notifyevent root" -e b   \
"E-mail root anytime"
```

2. To define a response on the managed node nodeB with the name "E-mail root anytime" that has an action named "E-mail root", to be used any time Saturday and Sunday and uses the command **/opt/rsct/bin/notifyevent root** for both events and rearm events, run this command on the management server:

```
 mkresponse -n "E-mail root" -d 1+7  \
-s "/opt/rsct/bin/notifyevent root" -e b   \
-p nodeB "E-mail root anytime"
```

3. To define a response on the managed node nodeB with the name "nodeB Warning notifications" as a copy of "nodeA Warning notifications" on the managed node nodeA, run this command on the management server:

```
mkresponse -c "nodeA Warning notifications":nodeA \
-p nodeB "nodeB Warning notifications"
```

These examples apply to peer domains:

1. To define a response on the current node with the name "E-mail root anytime" that has an action named "E-mail root", to be used any time Saturday and Sunday and uses the command **/opt/rsct/bin/notifyevent root** for both events and rearm events, run this command from any node in the domain:

```
mkresponse -n "E-mail root" -d 1+7  \
-s "/opt/rsct/bin/notifyevent root" -e b   \
"E-mail root anytime"
```

2. To define a response on the node nodeB in the domain with the name "E-mail root anytime" that has an action named "E-mail root", to be used any time Saturday and Sunday, that uses the

command **/opt/rsct/bin/notifyevent root** for both events and rearm events, and that sets two environment variables (PAGE ALL and TIMER SET), run this command from any node in the domain:

```
 mkresponse -n "E-mail root" -d 1+7  \
-s "/opt/rsct/bin/notifyevent root" -e b   \
-p nodeB -E 'ENV1="PAGE ALL", ENV2="TIMER SET"' \
"E-mail root anytime"
```

3. To define a response on the node nodeB in the domain with the name "nodeB Warning notifications" as a copy of "nodeA Warning notifications" on the node nodeA in the domain, run this command from any node in the domain:

```
mkresponse -c "nodeA Warning notifications":nodeA \
-p nodeB "nodeB Warning notifications"
```

Location

/opt/rsct/bin/mkresponse

mkrole Command

Purpose

Creates new roles.

Syntax

mkrole [**-R** *load_module*] *Attribute*=*Value* [*Attribute*=*Value* ...] *Name*

Description

The **mkrole** command creates a new role. The *Name* parameter must be a unique role name. You cannot use the **ALL** or **default** keywords as the role name.

You can use the System Management Interface Tool (SMIT) **smit mkrole** fast path to run this command.

If the system is configured to use multiple domains for the role database, the new role is created in the first domain specified by the **secorder** attribute of the roles stanza in the **/etc/nscontrol.conf** file. Use the **-R** flag to create a role in a specific domain.

Every role must have a unique role ID that is used for security decisions. If the **id** attribute is not specified when a role is created, the **mkrole** command automatically assigns a unique ID to the role.

When the system is operating in enhanced (RBAC) mode, roles created in the role database can be immediately assigned to users but are not used for security considerations until the database is sent to the kernel security tables using the **setkst** command.

Flags

Item	Description
-R *load_module*	Specifies the loadable module to use for role creation.

Parameters

Item	Description
Attribute=*Value*	Initializes a role attribute. Refer to the **chrole** command for the valid attributes and values.

Item	Description
Names	Specifies a unique role name string. **Restrictions on Creating Role Names** The *Name* parameter that you specify must be unique, and can be a maximum of 63 single-byte printable characters. To prevent inconsistencies, restrict role names to characters with the POSIX portable filename character set. You cannot use the keywords **ALL** or **default** as a role name. Additionally, do not use any of the following characters within a role-name string: • : (colon) • " (quotation mark) • # (pound sign) • , (comma) • = (equal sign) • \ (backslash) • / (forward slash) • ? (question mark) • ' (single quotation mark) • ` (back quotation mark) **Restriction:** The *Name* parameter cannot contain any space, tab, or newline characters.

Security

The **mkrole** command is a privileged command. You must assume a role that has the following authorization to run the command successfully.

Item	Description
aix.security.role.create	Required to run the command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files Accessed:

Mode	File
rw	/etc/security/roles
r	/etc/security/user.roles

Auditing Events:

Event	Information
ROLE_Create	role

Examples

1. To create the ManageRoles role and have the command automatically generate a role ID, use the following command:

```
mkrole authorizations=aix.security.role ManageRoles
```

2. To create the ManageRoles role in LDAP, use the following command:

```
mkrole -R LDAP authorizations=aix.security.role manageRoles
```

Files

Item	Description
/etc/security/roles	Contains the attributes of roles.
/etc/security/user.roles	Contains the role attribute of users.

mkrpdomain Command

Purpose

Creates a peer domain definition.

Syntax

To create a peer domain definition, by:

- Specifying node names on the command line:

  mkrpdomain [-t *TS_port*] [-g *GS_port*] [-Q *quorum_type* | *quorum_type_name*] [-c] [-m *fanout*] [-S *mode*] [-k *cssk_type* [-r *refresh_interval*]] [**–6**] [**-C** *cluster_type* **-R** *repository_disk* [**-D** *shared_disk1*[**,***shared_disk2*...]]] [-h] [-TV] *peer_domain node_name1* [*node_name2* ...]

- Using a list of node names in an input file:

  mkrpdomain -f | -F { *file_name* | "–" } [-t *TS_port*] [-g *GS_port*] [-Q {*quorum_type* | *quorum_type_name*}] [-c] [-m *fanout*] [-S *mode*] [-k *cssk_type* [-r *refresh_interval*]] [**–6**] [**-C** *cluster_type* **-R** *repository_disk* [**-D** *shared_disk1*[**,***shared_disk2*...]]] [-h] [-TV] *peer_domain*

To create a peer domain definition with the policy information:

mkrpdomain [-p *Policy*] *ApplDomain nameA* [*@host_nameA*] [*nameB* [*@host_nameB*] ...]

Description

The mkrpdomain command creates a peer domain definition with the name specified by the *peer_domain* parameter. The nodes that are specified by *node_name* are defined to the new peer domain. A peer domain can be used to provide high-availability services when you configure application and system resources.

The preprpnode command must have been run on each of the nodes to be defined to the peer domain. The preprpnode command prepares the security environment for the peer domain operations. See the preprpnode command for more information about peer domain definition requirements. Only those nodes that have the appropriate security setup are successfully defined to the peer domain.

The **mkrpdomain** command fails if one or more of these situations occurs:

- The name of the peer domain is already in use.
- One or more nodes cannot be successfully defined to the peer domain.

- The UDP port numbers for group services and topology services are not available on all of the nodes to be defined to the peer domain.

Use the **-c** flag to enable **mkrpdomain** to continue when there is an error on one of the nodes. The peer domain quorum rules can be modified by using the **-Q** flag. The quorum rules determine under what conditions operational changes, such as starting or stopping resources, and configuration changes, such as adding or removing a node, can be made. Start up quorum defines how many nodes are contacted to get configuration information to start the peer domain. In a typical environment, two quorum rule types are used: normal and quick. For the quick quorum type, only one node is contacted before the peer domain group is started. Operational and configuration quorum rules are the same. To see what quorum rule types are available on a node, run:

```
lsrsrc -c IBM.PeerDomain AvailableQuorumTypes
```

You can use the **-k** flag to set the cluster shared secret key (CSSK). The CSSK is used for message authentication in the peer domain. By default, the CSSK is disabled (that is, set to CSSKTYPE_None). To enable message authentication, use a CSSK value such as CSSKTYPE_DES_MD5 with the **-k** flag. Enabling message authentication affects performance. The complexity of the encryption algorithm determines the effect.

Message authentication also requires that the time-of-day clocks (TODs) of the nodes in the peer domain are synchronized — according to the system time — to within 2 minutes of each other. When the nodes' TODs are synchronized across the peer domain, this function helps to defend against message replay attacks. If the nodes' TODs are not synchronized to within 2 minutes of each other, messages that are passed between a sending node and a receiving node with a time difference that is longer than 2 minutes are discarded.

When message authentication is enabled by using the **-k** flag, a key refresh interval can be specified by using the **-r** flag. By default, the key is refreshed daily.

To change the CSSK type for a peer domain, use the `chrsrc` command. For example:

```
chrsrc -c IBM.RSCTParameters CSSKType=cssk_type
```

To list the CSSK type that is used for an online peer domain, use the `lsrsrc` command. For example:

```
lsrsrc -c IBM.RSCTParameters CSSKType
```

To cause the CSSK to be refreshed, use the `runact` command. For example:

```
runact -c IBM.PeerDomain UpdateKey
```

For information about setting up and managing CSSK settings, see the *Administering RSCT* guide.

Use the **-6** flag to establish a peer domain in which the IPv6 addresses that are configured on the nodes' network interfaces are visible as resources in **IBM.NetworkInterface** class. These IPv6 addresses are not used for heartbeating or internal peer domain operations. If the -6 flag is not specified, no IPv6 addresses are visible as resources in **IBM.NetworkInterface**.

The `mkrpdomain` command does not bring the peer domain online automatically. To bring the peer domain online, run the `startrpdomain` command. You can add nodes to the peer domain by using the `addrpnode` command. To remove nodes from the peer domain, use the `rmrpnode` command.

A node can be defined in more than one peer domain but it can be online in only one peer domain at a time.

Flags

Item	Description
`-6`	Specifies that the IPv6Support persistent class attribute of the **IBM.NetworkInterface** class has a value of 1 rather than the default (0) in the peer domain that is to be created. For any IP interface on any node in a cluster that has one or more IPv6 addresses configured, only one of these IPv6 addresses are made visible as a resource in **IBM.NetworkInterface**. Therefore, if a network interface has IPv4 addresses and IPv6 addresses configured on it, two resources in **IBM.NetworkInterface** refers to the interface (through the Name attribute), one with the IP address value set to the primary IPv4 address, and one with the selected IPv6 address. If multiple IPv6 addresses are configured on an interface, preference is given to global addresses over link-local addresses for representation as a resource. In addition, IPv6 addresses are used for heartbeating or internal peer domain operations. **Note:** Even if IPv6Support is changed, the current registered applications do not receive the notification for any resource addition or deletion until the domain or the **IBM.ConfigRM** class is restarted.
`-c`	Continues to run the `mkrpdomain` command on the remaining nodes. By default, if the `mkrpdomain` command fails on any node, it fails on all nodes. The `-c` flag overrides this behavior, so that the `mkrpdomain` command runs on the other nodes, even if it fails on one node.
`-C` *cluster_type*	Specifies the cluster type. Valid values are as follows: **0** Creates a peer domain. This value is the default. **1** Creates a peer domain and the underlying Cluster-Aware AIX (CAA) cluster. If you specify the **-C 1** flag, you must also specify a repository disk by using the **-R** flag. Also, you can optionally specify one or more shared disks by using the **-D** flag.
`-D` *shared_disk1* [,*shared_disk2*...]	Specifies one or more shared disks for a CAA cluster. If you specify the **-D** flag, you must also specify the **-C** and **-R** flags.

Item	Description
`-f` | `-F` { *file_name* | `"-"`	Specifies that node names are read from a file or from standard input. Use **-f** *node_file* or **-F** *node_file* to read the node names from a file. **Note:** The command requires that the following conditions be met to display a valid output: • Specify 1 node name per line. The command ignores any blank characters to the left of the node name. • Use a number sign (#) to indicate that the remainder of the line (or the entire line if the # is in column 1) is a comment. • Specify the actual host name of the node by using @ sign without any space between node name and its host name. An example of the syntax follows: `[nodeA@hostA]` By default, all of the nodes that are listed in *node_file*: • are group services group leader candidates • are used for quorum decisions • have access to the peer domain's tiebreaker mechanism You can customize node characteristics by using an at sign (@) control character followed by one or more of these special characters: **P | p** Specifies that the node is a group services group leader candidate. **Q | q** Specifies that the node is a quorum node. **B | b** Specifies that the node has access to the peer domain's tiebreaker mechanism. B or b can be specified for quorum nodes only. **!** Specifies that the node does not have a certain characteristic. For example, **!Q** indicates that the node is not a quorum node. When customizing node characteristics, consider the following (where x is P, Q, or B): • Use only one @ control character per line, followed immediately by one or more special characters, after the node name and before any comments. • Do not specify !QB for a node, as it results in an error. • If you use a node number, add it after the node name and before any comments. The node number can precede or follow the node characteristic specifications. • If x is specified for one or more nodes and !x is not specified for any nodes, the nodes that do not have an x specified are assumed to have a value of !x. • If !x is specified for one or more nodes and x is not specified for any nodes, the nodes that do not have an !x specified are assumed to have a value of x. • If x and !x are specified for different nodes in the same node file, all of the nodes in the file must have a specification of x or !x. See the *Administering RSCT* for more information. Use `-f "-"` or `-F "-"` to read the node names from standard input.
`-g` *GS_port*	Specifies the group services port number. This UDP port is for daemon-to-daemon communication. Any unused port in the range 1024 - 65535 can be assigned. The command fails if the specified port is unavailable. The default is 12348.
`-h`	Writes the command's usage statement to standard output.

Item	Description
-k *cssk_type*	Specifies the cluster shared secret key (CSSK) to be used for message authentication in the peer domain. Use the CSSK that best suits your applications in terms of the degree of data protection, overhead, and performance. The longer the key and message digest, the stronger the encryption algorithm. The stronger the algorithm, the slower the performance. The valid key types are as follows: **CSSKTYPE_None** Indicates that message authentication is disabled. This is the default value. **Note:** If the **-S** flag is specified with mode value `nist_sp800_131a`, the default CSSK type is `CSSKTYPE_AES256_SHA256`. **CSSKTYPE_DES_MD5** Indicates that a Data Encryption Standard (DES) key with the message digest function MD5 is used to generate a 16-byte signature. This CSSK is recommended if a high degree of data protection is not required and if you want good performance with less data overhead. **CSSKTYPE_3DES_MD5** Indicates that a triple DES key with an MD5 digest is used to generate a 16-byte signature. Compared to `CSSKTYPE_DES_MD5`, this CSSK provides added data protection with slower performance, but with the same data overhead. **CSSKTYPE_AES256_MD5** Indicates that an Advanced Encryption Standard (AES) 256-bit key with an MD5 digest is used to generate a 24-bit signature. This CSSK provides more data protection than `CSSKTYPE_3DES_MD5`, but with slower performance and more data overhead. The following CSSK types are compliant with the National Institute of Standards and Technology (NIST) Special Publications SP800-131a. You must be running RSCT 3.2.0.0, or later, to configure these key types. **CSSKTYPE_AES128_SHA256** Indicates that an Advanced Encryption Standard (AES) 128-bit key that has an SHA-1 (Secure Hash Algorithm) 256-bit digest is used to generate a 16-byte signature. **CSSKTYPE_AES128_SHA512** Indicates that an AES 128-bit key that has an SHA-1 512-bit digest is used to generate a 16-byte signature. **CSSKTYPE_AES256_SHA256** Indicates that an AES 256-bit key that has an SHA-2 256-bit digest is used to generate a 32-byte signature. **CSSKTYPE_AES256_SHA512** Indicates that an AES 256-bit key that has an SHA-2 512-bit digest is used to generate a 32-byte signature. **Notes:** • You must be running RSCT 2.4.7.1 or later to use this flag. • If the **-S** flag is specified with the mode value `nist_sp800_131a`, the CSSK type must be either `CSSKType_None` or a key type that is compliant with the mode. If the created domain is compliant with the mode value `nist_sp800_131a`, and the **-k** flag is not specified, the domain is configured to use CSSK type `CSSK_AES256_SHA256`.
-m *fanout*	Specifies the maximum number of threads to use in parallel operations for the specified peer domain. This value is stored as a persistent attribute in the peer domain's `IBM.PeerNode` class. *fanout* can be an integer from 16 to 2048. If this flag is not specified, the default value (128) is used.

Item	Description
-p *Policy*	Reads the policy from the user input when the `mkrpdomain` command creates the domain. You can use this command to specify the policy information when you create the domain. The valid values for the *Policy* attribute are 0 and 1. If you do not specify the `-p` flag for the `mkrpdomain` command, the default value 0 is set in non-CAA clusters and 1 is set in CAA clusters. If the value of policy is set as 1, the `Name` field of the `IBM.PeerNode` class is maintained in sync with the host name of the `IBM.PeerNode` class. If the value of policy is set as 0, the `Name` field is not maintained in sync with the host name, irrespective of the domain. However, the `-p` *0* flag cannot be specified for CAA domain as a limitation. The policy information can be changed by using a `chrsrc` class action after the cluster is created.
-Q *quorum_type* | *quorum_type_name*	Specifies the quorum rules that are used for startup, operational, and configuration quorum. Startup quorum defines how many nodes are contacted to obtain configuration information before the peer domain is started. Operational quorum defines how many nodes must be online to start and stop resources and how tie breaking is used. Configuration quorum defines how many nodes must be online to change the peer domain (adding or removing a node, for example). To see what quorum rule types are available on a node, run: `lsrsrc -c IBM.PeerDomain AvailableQuorumTypes` The valid values are as follows: **0 | normal** Specifies normal quorum rules. This value is the default. For startup quorum, at least half of the nodes are contacted for configuration information. For configuration quorum, more than half of the nodes must be online to make configuration changes. For operational quorum, the cluster or subcluster must have a majority of the nodes in the peer domain. If a tie exists between subclusters, the subcluster that holds the tiebreaker has operational quorum. **1 | quick** Specifies quick quorum rules. For startup quorum, even if no other nodes can be contacted, the node still comes online. For configuration quorum, more than half of the nodes must be online to make configuration changes. For operational quorum, the cluster or subcluster must have a majority of the nodes in the peer domain. If a tie exists between subclusters, the subcluster that holds the tiebreaker has operational quorum.
-r *refresh_interval*	Specifies the CSSK refresh interval when message authentication is enabled in the peer domain. This is the interval at which the CSSK is refreshed. The format of *refresh_interval* is: *dd*:*hh*:*mm*:*ss*, where *dd* is the number of days between key refreshes, *hh* is the number of hours, *mm* is the number of minutes, and *ss* is the number of seconds. The *refresh_interval* value can be truncated on the right, so `-r 5` means refresh every 5 days and `-r 0:12` means refresh every 12 hours. The default refresh interval is 1 day. The minimum refresh interval is 30 seconds. The maximum refresh interval is 30 days. The `-r` flag can be specified when the `-k` flag is used. You must be running RSCT 2.4.7.1 or later to use this flag.
-R *repository_disk*	Specifies the repository disk for a CAA cluster. If you specify the **-R** flag, you must also specify the **-C** flag.
-S *mode*	Enforces a security compliance mode for RSCT in the peer domain. The *mode* parameter can have the following values: **none** The domain does not enforce a security compliance mode. **nist_sp800_131a** RSCT is configured to be compliant with the National Institute of Standards and Technology (NIST) Special Publications SP800-131a. This mode value requires that all nodes that are specified in the **mkrpdomain** command must already be migrated to `nist_sp800_131a` mode or the nodes must be configured to use public or private keys that are compliant with this specification. **Note:** You must be running RSCT 3.2.0.0, or later, to use the **-S** flag.

Item	Description
`-t` *TS_port*	Specifies the topology services port number. This UDP port is used for daemon-to-daemon communication. Any unused port in the range 1024 - 65535 can be assigned. The command fails if the specified port is unavailable. The default is 12347.
`-T`	Writes the command's trace messages to standard error. For your software service organization's use only.
`-V`	Writes the command's verbose messages to standard output.

Parameters

peer_domain
Specifies the name of the new peer domain to be created. You can use these ASCII characters only in the peer domain name: A to Z, a to z, 0 to 9, . (period), and _ (underscore). In addition, the peer domain name *cannot* be IW.

node_name1* [*node_name2 ...]
Specifies the node (or nodes) to include in this peer domain definition. The node name is the IP address or the long or short version of the DNS host name. The node name must resolve to an IP address.

Security

The user of the `mkrpdomain` command requires **write** permission to the `IBM.PeerDomain` resource class on each node that is to be defined to the peer domain. This permission is set up by running the `preprpnode` command on each node that is to be defined to the domain, specifying the name of the node on which the user runs `mkrpdomain`.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the Resource Monitoring and Control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

Any node to be defined to the peer domain must be reachable from the node on which this command runs.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) file set for AIX.

Standard Input

When the `-f "-"` or `-F "-"` flag is specified, this command reads one or more node names from standard input.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To define a peer domain that is called `ApplDomain` that consists of a node that is called `nodeA`, run this command on `nodeA`:

   ```
   mkrpdomain ApplDomain nodeA
   ```

2. To define a peer domain that is called `ApplDomain` that consists of three nodes that are called `nodeA`, `nodeB`, and `nodeC`, run this command on `nodeA`, `nodeB`, or `nodeC`:

   ```
   mkrpdomain ApplDomain nodeA nodeB nodeC
   ```

3. To define a peer domain that is called `ApplDomain` that consists of 2 nodes that are called `nodeA` and `nodeB`, with a topology services port number of 1200 and a group services port number of 2400, run this command on `nodeA` or `nodeB`:

   ```
   mkrpdomain -t 1200 -g 2400 ApplDomain nodeA nodeB
   ```

4. To define a peer domain that is called `ApplDomain` that consists of 2 nodes that are called `nodeA` and `nodeB` by using message authentication key algorithm `CSSKTYPE_DES_MD5`, run this command on `nodeA` or `nodeB`:

   ```
   mkrpdomain -k CSSKTYPE_DES_MD5 ApplDomain nodeA nodeB
   ```

5. To define a peer domain that is called **ApplDomain** that consists of the nodes **nodeA**, **nodeB**, **nodeC**, **nodeD**, and **nodeE**, by using the **/pd/pdnodes.config** file, run the following command on any of the nodes:

   ```
   mkrpdomain -f /pd/pdnodes.config ApplDomain
   ```

   where the contents of **/pd/pdnodes.config** are as follows:

   ```
   # peer domain nodes for mkrpdomain
   nodeA       # dev node
   nodeB       # dev node
   nodeC       # prod node
   nodeD       # test node
   nodeE       # test node
   ```

6. To define a peer domain that is called **ApplDomain** that consists of **nodeA**, **nodeB**, **nodeC**, **nodeD**, and **nodeE**, by using the **/pd/pdnodes.config** file, which specifies that **nodeA** has access to the peer domain's tiebreaker mechanism, **nodeB** and **nodeC** cannot be used in quorum decisions, and **nodeC** and **nodeD** cannot be the group services group leader, run the following command on any of the nodes:

   ```
   mkrpdomain -f /pd/pdnodes.config ApplDomain
   ```

   where the contents of **/pd/pdnodes.config** are as follows:

   ```
   # peer domain nodes for mkrpdomain
   nodeA    @QB     # dev node
   nodeB    @!Q     # dev node
   nodeC    @!Q!P   # prod node
   nodeD    @!P     # test node
   nodeE    @Q      # test node
   ```

7. To define a peer domain that is called **ApplDomain**, which consists of 2 nodes that are called **nodeA** and **nodeB**, with the policy **NamePolicy 1**, run the following command:

   ```
   mkrpdomain -p 1 ApplDomain nodeA nodeB
   ```

   NamePolicy 1 means that any change in host name also updates the node name. In this case, the host name is not specified in the beginning. Hence, the node names (**nodeA** and **nodeB**) are set as host names for the respective nodes.

8. To define a peer domain that is called **ApplDomain**, which consists of 2 nodes that are called **nodeA** and **nodeB**, whose host names are **hostA** and **hostB**, run the following command:

   ```
   mkrpdomain ApplDomain nodeA@hostA nodeB@hostB
   ```

   These host names are the actual host names that are used for communication.

Location

/opt/rsct/bin/mkrpdomain

Files

The `/etc/services` file is modified.

mkrset Command

Purpose

Makes an rset containing the specified CPUs and memory regions and places it in the system registry.

Syntax

```
mkrset -c CPUlist [ -m MEMlist ] rsetname
```

Description

The **mkrset** command creates and places into the system registry an rset or exclusive rset (xrset) with the specified set of CPUs and/or memory regions. The rset name must not exist in the registry. The owner and group IDs of the rset will be set to the owner and group IDs of the command issuer. The rset will have read/write owner permissions and read permission for group and other. When used to create an xrset, the `mkrset` command changes the state of the corresponding CPUs on the system to exclusive mode. Creating an xrset requires root privilege.

Flags

Item	Description
-c	List of CPUs to be in the rset. This can be one or more CPUs or CPU ranges.
-m	List of memory regions to be in the rset. This can be one or more memory regions or ranges.

Parameters

Item	Description
rsetname	The name of the rset to be placed in the system registry. The name consists of a *namespace* and an *rsname* separated by a "/" (slash). Both the namespace and *rsname* may contain up to 255 characters. See the **rs_registername()** service for additional information about character set limits of rset names.

Security

The user must have root authority or CAP_NUMA_ATTACH and CAP_PROPAGATE capability.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To make an **rset** containing CPUs 0-7 named **test/cpus0to7**, type:

```
mkrset -c 0-7 test/cpus0to7
```

2. To make an **rset** containing CPUs 1, 3, 5, 6, 7, 10 named **test/lotsofcpus**, type:

```
mkrset -c 1 3 5-7 10 test/lotsofcpus
```

Files

Item	Description
/usr/bin/mkrset	Contains the **mkrset** command.

mkrsrc Command

Purpose

Defines a new resource.

Syntax

To define a new resource, using data that is...

- entered on the command line:

  `mkrsrc` [**-a** | **-N** { *node_file* | **"-"** }] [`-v`] [`-h`] [`-TV`] *resource_class attr=value...*

- predefined in an input file:

  `mkrsrc -f` *resource_data_input_file* [`-v`] [**-a** | **-N** { *node_file* | **"-"** }] [`-h`] [`-TV`] *resource_class*

To display the names and datatypes of the command arguments:

mkrsrc -l [**-h**] *resource_class*

To see examples of the `mkrsrc` command for a resource class:

`mkrsrc -e [-h] [-TV]` *resource_class*

Description

The `mkrsrc` command requests that the RMC subsystem define a new resource instance for the class specified by the *resource_class* parameter. At least one persistent attribute name and its value must be specified either as a parameter or by a resource definition file using the `-f` flag.

Before you run `mkrsrc`, you should run the `lsrsrcdef` command to determine which attributes are designated as `reqd_for_define` (required) or `option_for_define` (optional). Only attributes that are designated as `reqd_for_define` or `option_for_define` can be defined using the `mkrsrc` command. The `lsrsrcdef` command also identifies the datatype for each attribute. The value specified for each attribute must match this datatype.

To verify that all of the attribute names that are specified on the command line or in *resource_data_input_file* are defined as persistent attributes and are designated as `reqd_for_define` or `option_for_define`, use the `-v` flag. When the `mkrsrc` command is run with the `-v` flag, the resource is not defined. Instead, the resource attributes are merely verified to be persistent and designated as `reqd_for_define` or `option_for_define`. Once you have run `mkrsrc -v` to verify that all of the attributes that are specified on the command line or in *resource_data_input_file* are valid, you can issue the `mkrsrc` command without the `-v` flag to define the new resource.

If you are running in an RSCT peer domain or on the management server in an RSCT management domain and the resource class management type is subdivided, you can create the same resource on multiple nodes in one of two ways. The first way is to use the **-N** *node_file* flag to indicate that the node names to create the resources on are in a file. Use **-N "-"** to read the node names from standard input. The second way is to specify multiple node names in the **NodeNameList** resource attribute. The **NodeNameList** attribute defines where the resource is created when a cluster is present. If the **NodeNameList** attribute is not used, the resource is created on the local node. To find out if a resource class management type is subdivided, enter **lsrsrcdef -c** *resource_class* | **grep properties**.

Flags

-e
Displays examples of `mkrsrc` command-line input for:

1. required attributes only
2. required and optional attributes

-f *resource_data_input_file*
Specifies the name of the file that contains resource attribute information.

-l
Lists the command arguments and datatypes. Some resource managers accept additional arguments that are passed to the define request. Use this flag to list any defined command arguments and the datatypes of the command argument values.

-N { *node_file* | "-" }
Specifies that node names are read from a file or from standard input. Use **-N** *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (#) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use **-N "-"** to read the node names from standard input.

The **CT_MANAGEMENT_SCOPE** environment variable determines the scope of the cluster. If the resource class management type of the resource that is to be defined is subdivided and **CT_MANAGEMENT_SCOPE** is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and **CT_MANAGEMENT_SCOPE** is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set **CT_MANAGEMENT_SCOPE** to 2.

-v
Verifies that all of the attribute names specified on the command line or in the input file are defined as persistent attributes and are designated as `reqd_for_define` or `option_for_define`. The `mkrsrc` command does *not* define any resources when you use this flag.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

resource_class
Specifies the resource class name of the resource to be defined.

attr=value...
Specifies the attributes of the resource being defined. When defining a new resource instance, there are specific required attributes for each resource that must be defined. These attributes can be specified as parameters on the command line or defined in an input file by using the `-f` flag.

attr
The name of a persistent attribute for this resource. This attribute must be designated as `reqd_for_define` or `option_for_define`. Use the `lsrsrcdef` command to check the designation.

value
The value for this persistent attribute. The data type for this value must match the defined data type for the value of this attribute. Use the `lsrsrcdef` command to verify the data type for each attribute.

Security

The user needs write permission for the *resource_class* specified in `mkrsrc` to run `mkrsrc`. Permissions are specified in the access control list (ACL) file on the contacted system. See *Administering RSCT* guide for information about the ACL file and how to modify it.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the **rsct.rmc** fileset for the AIX® operating system.

Standard Output

- All command output is written to standard output.
- When the **-h** flag is specified, this command's usage statement is written to standard output.
- When the **-V** flag is specified, this command's verbose messages (if there are any available) are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To create a new resource in the `IBM.Host` class, assuming you already know which persistent attributes are required when defining a resource of this class, enter:

```
mkrsrc IBM.Host Name=c175n05
```

2. To create a new resource in the IBM.Processor class by first generating a template to aid in the defining of these resources, enter:

```
lsrsrcdef -i IBM.Processor > /tmp/IBM.Processor.rdef
```

   Then, edit the file /tmp/IBM.Processor.rdef and enter values for all of the attributes, substituting the type for an appropriate value, or leaving it blank for the default value.

   Finally, enter:

```
mkrsrc -f /tmp/IBM.Processor.rdef IBM.Processor
```

3. To create two new IBM.Host resources using the information defined in file /tmp/IBM.Host.rdef, enter:

```
mkrsrc -f /tmp/IBM.Host.rdef IBM.Host
```

   where the file /tmp/IBM.Host.rdef looks like this:

```
PersistentResourceAttributes::
resource 1:
    Name        = c175n04

resource 2:
    Name        = c175n05
```

4. This example creates a new resource in the IBM.Foo class. In this class, Name and NodeList are required attributes. The Binary, SD, StringArray, and SDArray attributes are optional. This example shows how to enter the more difficult data types from the command line. The data types for the optional attributes (Binary, SD, StringArray, and SDArray) are self-explanatory. Enter:

```
mkrsrc IBM.Foo Name=c175n05 \
NodeList={1} \
Binary="0xaabbccddeeff00" \
SD='[testing123,1,{2,4,6}]' \
StringArray='{"testing 1 2 3",testing123,"testing 1 2 3"}' \
SDArray='{["testing 1 2 3",1,{1,3,5}],[testing,2,{2,4,6}]}'
```

5. To create resources for the IBM.Example class on multiple nodes in a peer domain, run this command:

```
mkrsrc -N /u/joe/common_node_file  IBM.Example  Name=Example_bar1 \
Binary="0xaabbccddeeff00"
```

   where the contents of **/u/joe/common_node_file** look like this:

```
# common node file
#
node1.ibm.com      main node
node2.ibm.com      main node
node4.ibm.com      backup node
node6.ibm.com      backup node
#
```

6. To create resources of the IBM.Example class on multiple managed nodes in a management domain, run this command on the management server:

```
mkrsrc IBM.Example  Name=Example_bar1 Binary="0xaabbccddeeff00" \
NodeNameList='{"mgnode1.ibm.com","mgnode2.ibm.com"}'
```

   where the contents of **/u/joe/common_node_file** look like this:

```
# common node file
#
node1.ibm.com      main node
node2.ibm.com      main node
node4.ibm.com      backup node
node6.ibm.com      backup node
#
```

Note: As discussed in the `rmccli` general information file, attribute values for certain data types (structured data, array of structured data, and arrays containing strings enclosed in double quotation marks) should be enclosed in single quotation marks.

Location

/opt/rsct/bin/mkrsrc

mkrtc Command

Purpose

Configures or unconfigures Power SC real-time compliance for the operating system instance.

Syntax

To configure Power SC real-time compliance:

mkrtc **-e** *email1, email2...* [**-a** *alertStyle*] [**-d** *debug*] [**-i** *infoLevel*] [**-s** *emailSubject*] [**-c** *minCheckTime*]

To unconfigure Power SC real-time compliance:

mkrtc **-u**

Description

The **mkrtc** command is used to configure or unconfigure Power SC real-time compliance. To configure the **-e** flag, the email addresses must be provided as arguments. All other flags are optional. The **mkrtc** command saves the options to the `/etc/security/rtc/rtcd.conf` file, adds the Power SC real-time compliance entry to the `/etc/inittab`, and starts the **rtcd** daemon.

On unconfiguration, the **mkrtc** command removes the entry from the `/etc/inittab` file and stops the **rtcd** daemon.

Flags

Flag	Description
-a *alertStyle*	Specifies the alert style. The following are valid values: • `Once`: Alerts once for the same set of compliance violations. This is the default alert style. • `Event`: Alerts once for the same set of compliance violations, but keeps alerting for each file modification event. • `Always`: Alerts compliance violations and the file modification. It keeps alerting for the file modification.
-c *minCheckTime*	Specifies the minimum amount of time between the compliance verifications. This flag checks the Power SC for compliance regularly even without file modification triggers, so that the **mkrtc** command can detect compliance implications in the files that are created by the user. For example, this flag can detect the `.rhost` file creation in the home directory that can have compliance implication. The default minimum time is 30 minutes. If this value is set to 0, it indicates that the compliance check is never run unless the files are modified.
-d *debug*	Specifies the debug option to be turned on or off. The valid values are `On` or `Off`. The default value is `Off`.

Flag	Description
-e *email1, email2...*	Provides a comma-separated list of emails to which email alerts are to be sent.
-i *infoLevel*	Specifies the information level of file modification events.
-s *emailSubject*	Provides the subject line to be used for the email alert.
-u	Unconfigures the Power SC real-time compliance.

On configuring Power SC real-time compliance, the **mkrtc** command performs the following tasks:

1. Updates the `/etc/security/rtc/rtcd.conf` file with the options from the command line.
2. Updates the `/etc/inittab` file with *pscrtc:2:wait: /usr/bin/startsrc -s rtcd*.
3. Starts the **rtcd** daemon.

On unconfiguration, the **mkrtc** command performs the following tasks:

1. Removes the Power SC real-time compliance entry from `/etc/inittab` file.
2. Stops the **rtcd** daemon.

Security

Only the root user and users with **aix.security.aixpert** authorization are authorized to run this command.

Exit Status

Value	Description
0	The command runs successfully.
>0	An error occurred. The printed error message lists further details about the type of failure.

Examples

1. To configure Power SC real-time compliance, type the following command:

   ```
   # mkrtc -e test@abc.com,dummy@abc.com -a event
   ```

   This command configures Power SC real-time compliance to send compliance violation alert and file modification events to *test@abc.com* and *dummy@abc.com*. The alert style is set to *event*.
2. To unconfigure Power SC real-time compliance, type the following command:

   ```
   # mkrtc -u
   ```

Files

Mode	File
rw	`/etc/security/rtc/rtcd.conf`

mkseckrb5 Command

Purpose

Migrates existing operating system users to Kerberos.

Syntax

mkseckrb5 [**-h** | [**-r**] [**user_name...**]]

Description

This command gets the list of user names and creates Kerberos users. If the **-r** flag is not specified, the command prompts for a new password for each user.

Item	Description
Standard Output	Consists of information messages when the **-h** flag is used.
Standard Error	Consists of error messages when the command cannot complete successfully.

Flags

Item	Description
-h	Specifies that the command is only to display the valid command syntax.
-r	Specifies that random passwords are to be used.

Exit Status

Failure of this command to execute successfully results in incomplete migration. The admin must check the Kerberos database for the users that were migrated before taking further action.

Item	Description
0	Indicates the successful completion of the command.
1	Indicates that an error occurred.

Security

Only the root user is authorized to use this command.

Examples

1. To display the command syntax, type:

```
mkseckrb5 -h
```

2. To migrate existing users to Kerberos users, type:

```
mkseckrb5
```

3. To migrate user trojan to Kerberos user with random passwd, type:

```
mkseckrb5 -r trojan
```

Files

Item	Description
/usr/sbin/mkseckrb5	Contains the **mkseckrb5** command.

mksecldap Command

Purpose

Sets up an AIX system as an LDAP server or client for security authentication and data management.

Syntax

The syntax to set up a server is:

mksecldap **-s** **-a** *adminDN* **-p** *adminpasswd* **-S** *schematype* [**-d** *baseDN*] [**-n** *port*] [**-k** *SSLkeypath*] [**-w** *SSLkeypasswd*] [**-x** *proxyDN* **-X** *proxypasswd*] [**-u** *NONE*] [**-v** *LDAPVersion*] [**-U**] [**-j** < *ssl*|*tls*|ssl*tls*|*none*| *sslonly*>]

The syntax to set up a client is:

mksecldap **-c** **-h** *serverlist* **-a** *bindDN* **-p** *bindpwd* [**-d** *baseDN*] [**-n** *serverport*] [**-k** *SSLkeypath*] [**-w** *SSLkeypasswd*] [**-t** *cachetimeout*] [**-C** *cachesize*] [**-P** *NumberofThreads*] [**-T** *heartBeatInt*] [**-M** *searchMode*] [**-D** *defaultEntry*] [**-A** *authType*] [**-i** *databaseModule*] [**-u** *userlist*] [**-U**] [**-j** <*ssl*|*tls*>]

Description

The **mksecldap** command can be used to set up IBM Directory servers and clients for security authentication and data management.

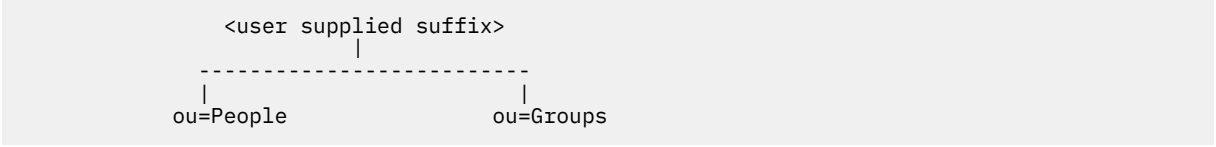
Notes:

1. The client (**-c** flag) and the server (**-s** flag) options cannot be used at the same time. When setting up a server, the **mksecldap** command might need to be run twice on that machine. Once to set up the server, and again to set up the system as a client.
2. The name and location of the LDAP server configuration file depends on the version of LDAP software installed. Refer to the LDAP software documentation of the installed release for more information.

Server Setup

Make sure that the LDAP server and the back-end IBM DB2 software are installed. You do not need to pre-configure IBM DB2 to run the **mksecldap** command for LDAP server setup. When you run the **mksecldap** command to set up the server, the command will:

1. Create a DB2® instance with **ldapdb2** as the default instance name.
2. If IBM Directory Server 6.0 or later is being configured then an LDAP server instance with the default name of **ldapdb2** is created. A prompt is displayed for the encryption seed to use to create the key stash files. The input encryption seed must be at least 12 characters.
3. Create a DB2 database with **ldapdb2** as the default database name. If a database already exists, **mksecldap** will bypass the above two steps. (This is the case when the LDAP server has been set up for other usage.) The **mksecldap** command will use the existing database to store the AIX user/group data.
4. Create the base DN (suffix) of the directory information tree (DIT). It is required that the base DN start with one of these attributes: **dc**, **o**, **ou**, **c**, **cn**. If no baseDN is supplied from the command line, the default suffix is set to **cn=aixdata** and the user/group data is placed under the **cn=aixdata** DN. Otherwise, the **mksecldap** command uses the user-supplied DN specified with the **-d** option. Users and groups will be exported to LDAP using the **sectoldif** command. The directory information tree (DIT) that will be created by default is shown below.



5. If **-u *NONE*** is not specified, then export the data from the security database files from the local host into the LDAP database. If **-u *NONE*** *is* specified, then **mksecldap** does not create the `ou=People` and `ou=Group` containers as it normally would, nor does it export users and groups. Depending on the **-S** option, the **mksecldap** command exports users/groups using one of the three LDAP schemas:
   - **AIX** - AIX schema (**aixaccount** and **aixaccessgroup** objectclasses)
   - **RFC2307** - RFC 2307 schema (**posixaccount**, **shadowaccount**, and **posixgroup** objectclasses)
   - **RFC2307AIX** - RFC 2307 schema with full AIX support (**posixaccount**, **shadowaccount**, and **posixgroup objectclasses**, plus the **aixauxaccount** and **aixauxgroup** object classes).
6. Set the LDAP server administrator DN and password.
7. Set the server to listen to a specified port if the **-n** option is used. The default port is 389. Also, TLS use this port as default port (636 for SSL).
8. Updates the **/usr/lib/security/methods.cfg** file with the LDAP module configuration. If the **-i** option is entered from the command line, it also sets a LDAPA authentication-only module and a compound loadmodule (for example, LDAPAfiles when the **-i** *files* option is specified) with LDAPA serves for authentication and the databaseModule serves for identification.
9. Create the proxy entry if the -x and -X options are specified. Create an ACL for the base DN using the proxy entry. The default ACL can be found in /etc/security/ldap/proxyuser.ldif.template. The proxy entry can be used by client systems to bind to the server (see client setup section in this file).
10. Set the server to use SSL (secure socket layer) or TLS (transport layer security) if the **-k** option is specified for secure data transfer between this server and the clients. This setup requires the **GSKIT** to be installed and creation of an SSL or TLS key.
11. Installs the **/usr/ccs/lib/libsecldapaudit.a** LDAP server plug-in. This plug-in supports AIX audit of the LDAP server.
12. Start/restart the LDAP server after all the above is done.
13. Add the LDAP server process (**slapd**) to **/etc/inittab** to have the LDAP server start after reboot.

Note: The **-U** option resets a previous setup for the server configuration file. It has no effect on the database. The first time the **mksecldap** command is run, it saves two copies of the server configuration file in the **/etc/security/ldap** directory. One is saved as the server configuration file name appended with **.save.orig** and the other is appended with **.save**. During each subsequent run of the **mksecldap** command, only the current server configuration is saved as a **.save** file. The undo option restores the server configuration file with the **.save** copy. In AIX 5.3 it is possible to invoke **mksecldap -s** in succession to create and populate multiple suffixes. If this has been performed then the **.save.orig** file will need to be manually restored in order to revert to the initial configuration file.

Client Setup

Make sure that the LDAP client fileset is installed and the LDAP server has been setup and is running. The **mksecldap** command performs the following steps during client setup:

1. Saves the LDAP server(s)' host name.
2. Saves the user base DN and group base DN of the server. If no **-d** option is supplied from command line, the **mksecldap** command searches the LDAP server for **aixaccount**, **aixaccessgroup**, **posixaccount**, **posixgroup**, and **aixauxaccount** objectclasses, and sets up the base DNs accordingly. If the server has multiple user or group bases, you must supply the **-d** option with a Relative Distinguished Name (RDN) so that the **mksecldap** command can setup the base DNs to the ones within that RDN.

   If the **posixaccount** objectclass is found during client setup, **mksecldap** will also try to search for base DNs for the following entities from the server and save any that are found:
   - hosts
   - networks
   - services
   - netgroups

- protocols
- rpc
- authorizations
- roles
- privcmds
- privdevs
- privfiles
- usrkeystore
- grpkeystore
- efscookies
- admkeystore
- domains
- domobjs

3. Determines the schema type used by the LDAP server - **AIX** specific schema, **RFC 2307** schema, **RFC 2307** schema with full AIX support, or Microsoft Services for UNIX 3.0 schema. It sets the objectclasses and attribute maps in the **/etc/security/ldap/ ldap.cfg** file accordingly. The **mksecldap** command does not recognize other schema types, so clients must be setup manually.
4. Sets SSL or TLS for secure data transfer between this host and the LDAP server. This step requires that the client SSL or TLS key and the key password are created in advance, and the server must be setup to use SSL or TLS for the client SSL or TLS to work.
5. Encrypts the bind password.
6. Saves the LDAP server bind DN and password. The DN/password pair must exist on the LDAP server. If the bind DN and password are not given, **mksecldap** uses anonymous bind. Some of the data might not be returned from the LDAP server with anonymous bind. Consult your LDAP administrator before you choose anonymous bind.
7. Sets the optionally specified configuration values as defined in the client setup flags section.
8. Optionally sets the list of users or all users to use LDAP by modifying their SYSTEM line in the **/etc/security/user** file. For more information on enabling LDAP login, see the following note.
9. Starts the client daemon process (**secldapclntd**).
10. Adds the client side daemon process to **/etc/inittab** to have this daemon start after a reboot.

Note: All client configuration data is saved to the **/etc/security/ldap/ldap.cfg** configuration file. The **-U** option resets a previous setup to the **/etc/security/ldap/ldap.cfg** file by replacing the file with the configuration stored in **/etc/security/ldap/ldap.cfg.save**. Setting the SYSTEM to LDAP for the default stanza of **/etc/security/user** only allows LDAP users to login to the system. Setting the SYSTEM to LDAP or `compat` allows both LDAP users and local users to login to the system.

Flags

For Server Setup

Item	Description
-a *AdminDN*	Specifies the LDAP server administrator DN.
-d *baseDN*	Specifies the suffix or base DN of the AIX subtree. The default is **cn=aixdata**.
-j < *ssl*|*tls*|ssl*tls*|*none*|*sslonly*>]	Specifics the encryption connection type that is used during the communication with the LDAP clients. Valid values are SSL, TLS, SSLTLS, and SSLONLY. If the **-k** and **-w** flags are specified without the **-j** flag, the default connection type is SSL.

Item	Description
-k *SSLkeypath*	Specifies the full path to the SSL or TLS key database of the server.
-n *port*	Specifies the port number that the LDAP server listens to. Default is 389 for non-SSL and 636 for SSL.
-p *adminpasswd*	Specifies the clear text password for the administrator DN.
-S *schematype*	Specifies the LDAP schema used to represent user/group entries in the LDAP server. Valid values are AIX, RFC2307, and RFC2307AIX.
-s	Indicates that the command is being run to setup the server.
-w *SSLkeypasswd*	Specifies the password for the SSL or TLS key.
-U	Specifies to undo the previous server setup to the LDAP configuration file. The database is not affected.
-u `NONE`	Specifies not to migrate users and groups from local system. The only valid value is `NONE`. Any other values are ignored. When this option is used, `mksecldap` does not create the `ou=People` and `ou=Group` containers as it normally would, nor does it export users and groups. No **-S** option is required with this option.
-v *LDAPVersion*	Denotes a specific version of the LDAP server fileset to configure. The value must be in the format *#.#* where *#* is a number. For example, `6.0`. If not specified, the **mksecldap** command configures the most recent version of the LDAP server fileset that is installed.
-X *proxypasswd*	Specifies the password for the proxy DN.
-x *proxyDN*	Specifies the DN of the proxy entry. This entry can be used by client systems to bind to this server.

For Client Setup

Item	Description
-a *bindDN*	Specifies the DN to bind to the LDAP server. The DN must exist on the LDAP server. If authtype is unix_auth, bindDN must have read access to the userPassword field on the LDAP server. Without the -a option, **mksecldap** configures anonymous bind. **Note:** Some of the data might not be retrieved from the LDAP server with anonymous bind. Consult your LDAP server administrator about using anonymous bind.
-A *authType*	Specifies the authentication mechanism used to authenticate users. Valid values are **unix_auth** and **ldap_auth**. The default is **unix_auth**. The values are defined as follows: • **unix_auth** - Retrieve user password from LDAP and perform authentication locally. • **ldap_auth** - Bind to LDAP server, sending password in clear text, for authentication. **Note:** When using **ldap_auth** type authentication, the use of SSL or TLS is strongly recommended since during authentication passwords will be sent in clear text to the LDAP server.

Item	Description
-i *databaseModule*	Specifies the configuration of LDAP as the authentication-only module (LDAPA) of a compound loadmodule. The databaseModule option specifies the database module of the compound loadmodule.
-j <*ssl*|*tls*>	Specifics the encryption connection type that is used during the communication with the LDAP server. Valid values are SSL and TLS. If the **-k** and **-w** flags are specified without the **-j** flag, the default connection type is SSL.
-c	Indicates the command is being run to setup the client.
-C *Cachsize*	Specifies the maximum number of user entries that can be used in the client-side daemon cache. Valid value is in the range 100 - 65536 for user cache. The default is 1000. The valid range for the group cache is 10-65536. The default value is 100. If you set the user cache entry in the **start-secldapclntd** command, by using the **-C** option, the group cache is set to 10% of the user cache.
-D *defaultEntryLocation*	Specifies the location of the default entry. Valid values are **ldap** and **local**. The default is **ldap**. The values are defined as follows: • **ldap** - Use the default entry in LDAP for all attribute default values. • **local** - Use the default stanza from local **/etc/security/user** file for all attribute default values.
-d *baseDN*	Specifies the base DN for the **mksecldap** command to search for the user base DN and group base DN. If not specified from the command line, the entire database is searched.
-h *serverlist*	Specifies a comma separated list of hostnames (server and backup servers).
-k *SSLkeypath*	Specifies the full path to the client SSL or TLS key.
-M *searchMode*	Specifies the set of user and group attributes to be retrieved. Valid values are **ALL** and **OS**. The default is **ALL**. The values are defined as follows: • **ALL** - Retrieve all attributes of an entry. • **OS** - Retrieve only the operating system required attributes of an entry. Non-OS attributes like telephone number, binary images etc. will not be returned. **Note:** Use OS only when entries have many non-OS required attributes or attributes with large value, e.g. binary data, to reduce sorting effort by the LDAP server.
-n *serverport*	Specifies the port number that the LDAP server is listening to.
-p *bindpasswd*	Specifies the clear text password for the bindDN used to bind to the LDAP server.
-P *NumberofTreads*	Specifies the number of threads that the client side daemon uses. Valid values are 1-256. The default value is 10.
-t *Cachetimeout*	Specifies the maximum time length that a cache entry expires. Valid values are 60-3,600 seconds. The default is 300 seconds. Set this value to 0 to disable caching.

Item	Description
-T *heartBeatInt*	Specifies the time interval of heartbeat between this client and the LDAP server. Valid values are 60-3,600 seconds. Default is 300.
-u *userlist*	Specifies the comma separated list of user names to enable for LDAP authentication. These users will have their registry and SYSTEM attributes set to use LDAP. Specify **ALL** to enable all users on the client. **Note:** Alternatively, the SYSTEM attribute in the default stanza of **/etc/security/user** can be set to LDAP, allowing only LDAP users to log in. Setting the SYSTEM attribute to LDAP or compat allows both LDAP users and local users to log in to the system.
-w *SSLkeyfilepath*	Specifies the password for the client SSL or TLS key.
-U	Specifies to undo the previous client setup to the LDAP client configuration file.

Security

A user with the **aix.security.ldap** authorization is authorized to use this command.

Examples

1. To setup a LDAP server of RFC2307AIX specific schema for users and groups, enter:

   ```
   mksecldap -s -a cn=admin -p adminpwd -S rfc2307aix
   ```

   This sets up a LDAP server with LDAP server administrator DN being **cn=admin**, password being **adminpwd**. User and group data is exported from local files to the default **cn=aixdata** suffix using RFC2307AIX schema.

2. To setup a LDAP server with a baseDN other than the default and with SSL secure communication , enter:

   ```
   mksecldap -s -a cn=admin -p adminpwd -d o=mycompany,c=us -S rfc2307 \ -k /usr/ldap/
   serverkey.kdb
    -w keypwd
   ```

   This sets up a LDAP server with LDAP server administrator DN being **cn=admin**, password being **adminpwd**. User and group data is exported from local files to the **o=mycompany,c=us** suffix using RFC2307 schema. The LDAP server uses SSL communications by using the key stored at **/usr/ldap/serverkey.kdb**. The password to the key, **keypwd**, must also be supplied.

3. To setup a LDAP server of RFC2307AIX schema type and create a proxy account, enter:

   ```
   mksecldap -s -a cn=admin -p adminpwd -d c=us -S rfc2307aix -x cn=proxy,c=us -X proxypwd
   ```

   This sets up a LDAP server with LDAP server administrator DN being **cn=admin**, password being **adminpwd**. User and group data is exported from local files to the **c=us** suffix using RFC2307AIX schema. A proxy identity is setup with DN being **cn=proxy,c=us** and password **proxypwd**. The ACL specified in **/etc/security/ldap/proxy.ldif.template** will also have been applied on the server for the **cn=proxy,c=us** DN.

4. To undo a previous server setup:

   ```
   mksecldap -s -U
   ```

   This undoes the previous setup to the server configuration file. Note, for safety reasons, this does not remove any database entries or database created by a previous setup. One has to remove the database entries/database manually if they are not needed any more.

5. To setup a client to use the **server1.ibm.com** and **server2.ibm.com** LDAP servers, enter:

```
mksecldap -c -a cn=admin -p adminpwd -h server1.ibm.com,server2.ibm.com
```

   The LDAP server administrator DN and password is supplied for this client to authenticate to the server. The **mksecldap** command contacts the LDAP server for schema type used, and sets up the client accordingly. Without the **-d** option from the command line, the entire server DIT is searched for the user base DN and the group base DN.

6. To setup the client to talk to the **server3.ibm.com** LDAP server using SSL, enter:

```
mksecldap -c -a cn=admin -p adminpwd -h server3.ibm.com -d o=mycompany,c=us
-k /usr/ldap/clientkey.kdb -w keypwd -u user1,user2
```

   This sets up a LDAP client similar to case 3, but with SSL communication. The **mksecldap** command searches the **o=mycompany,c=us** RDN for user base DN and group base DN. Account user1 and user2 are configured to authenticate through LDAP.

   Note: The **-u ALL** option enables all LDAP users to login to this client.

7. To setup a client to talk to **server4.ibm.com** and use **ldap_auth** authentication with a proxy bind, enter:

```
mksecldap -c -a cn=proxy,c=us -p proxypwd -h server4.ibm.com -A ldap_auth
```

   This sets up an LDAP client to bind to the LDAP server with the **cn=proxy,c=us** DN. Because the administrator DN is not used, the access granted to the client is dependent on the ACL setup on the LDAP server for the **cn=proxy,c=us** DN. The client is also setup to use **ldap_auth**-type authentication which sends passwords in clear text to the LDAP server for comparison.

   Note: When using **ldap_auth**-type authentication, the use of SSL or TLS is strongly recommended because during authentication passwords will be sent in clear text to the LDAP server.

8. To undo a previous client setup, enter:

```
mksecldap -c -U
```

   This undoes the previous setup to the **/etc/security/ldap/ldap.cfg** file. This does not remove the **SYSTEM=LDAP** and **registry=LDAP** entries from the **/etc/security/user** file.

9. To setup a client using LDAP as authentication-only module, and using files for user identification, enter:

```
mksecldap -c -a cn=admin -p adminpwd -h server1.ibm.com -i files -A ldap_auth
```

   This sets up a LDAPAfiles compound loadmodule, where the module LDAPA is used for user authentication and files is used for user identification. Authentication is set to ldap_auth.

Files Accessed

Item	Description
Mode	**File**
r	**/etc/passwd**
r	**/etc/group**
r	**/etc/security/passwd**
r	**/etc/security/limits**
r	**/etc/security/user (on the server)**
rw	**/etc/security/user (on the clients)**
r	**/etc/security/environ**

Item	Description
r	/etc/security/user.roles
r	/etc/security/lastlog
r	/etc/security/smitacl.user
r	/etc/security/mac_user
r	/etc/security/group
r	/etc/security/smitacl.group
r	/etc/security/roles
rw	/etc/security/login.cfg (on the server)
rw	/etc/slapd32.conf (on the server)
rw	/etc/security/ldap/ldap.cfg (on the client)

mksecpki Command

Purpose

mksecpki configures AIX PKI server components. The components of AIX PKI are Certificate Authority, Registration Authority, and Audit subsystems.

Syntax

mksecpki {**-u** *username* **-f** *reference_file* [**-p** *CA_port*] [**-H** *ldap_host*] [**-D** *dn* **-w** *password*] [**-i** *certificate_issuer_dn*] | **-U** *username*}

Description

The **mksecpki** command configures AIX PKI server components. **mksecpki** must be run after configuring an LDAP server to publish certificates. The values for the options **-H**, **-D**, **-w**, and **-i** must be the same values as the ones specified during the LDAP configuration. Otherwise, the CA will not be able to publish certificates to LDAP.

The **-u** option specifies the AIX username which will host AIX PKI. The username must follow AIX username rules. Do not use **-u** and **-U** together. The invoker of the command will be asked to provide a password for the username. **mksecpki** will create a database instance with the same name.

The **-f** option specifies the file containing the reference number and passphrase. The client certificate requests will use these exact same values while communication with the CA. The reference number and passphrase are each specified on a separate line. The following is the contents of an example **iafile**:

```
11122233
temppwd1234
```

The **-p** option specifies the port that Certificate Authority accepts the certificate requests. If no port number is given, 1077 will be assumed.

The **-H** option specifies the hostname of the LDAP server where the certificates are published to. Prior to invoking the **mksecpki** command, an LDAP server must be setup to publish certificates. Otherwise, the certificates will not be published to LDAP, however, certificate will be returned to the requestor when certificate management commands are used. If the **-H** option is not given the localhost will be used as the hostname.

The **-D** option is used to specify the directory administrators distinguished name. This must be the same one that is specified during the configuration of the LDAP server.

The **-w** option specifies the password corresponding to the administrator DN. It is an error not to specify both the admin DN and password.

The **-i** option specifies the distinguish name of the Certificate Authority issuing the certificates. This must be the same value as the one given when setting an LDAP server for publishing certificates.

The **-U** option specifies the username that hosts the AIX PKI that will be unconfigured. The command will confirm the unconfiguration before starting its operation. This option removes the username from the system. The invokers of this command will be asked if they want to remove the home directory of the username. When this command runs without errors, it displays a message indicating the successful completion. The invoker of this command is recommended to wait for this message.

Flags

Item	Description
-u *username*	Specifies the name of the username that is going to be created that will host AIX PKI server components.
-f *reference_file*	Specifies the file which contains the reference number and passphrase that is used when making a certificate creation request.
-p *CA_port*	Specifies the Certificate Authority Communication Port.
-H *ldap_host*	Specifies the LDAP host where the certificates are going to be published.
-D *adminDN*	Specifies directory administrator distinguished name (DN).
	Note: The **-D** option requires that the **-w** *password* option also be specified.
-w *password*	Specifies directory administrator password.
-i *certificate_issuer_dn*	Specifies the distinguished name of the Certificate Authority issuing certificates.
-U *username*	Specifies the username which hosts the AIX PKI that will be unconfigured.

Security

This command should grant execute (x) access only to the `root` user and members of the security group.

Examples

To configure AIX PKI server side using **pkitest.ibm.com** as the LDAP host name for publish certificates and using o=aix,c=us as the issuer name, enter:

```
$ mksecpki -u pkiuser -f iafile -p 829 -H pkitest.ibm.com -D cn=admin
-w password -i o=aix,c=us
```

where **iafile** contains the reference number and passphrase.

To unconfigure the server, enter:

```
$ mksecpki -U pkiuser
```

Files

/usr/lib/security/pki/ca.cfg

mksensor Command

Purpose

Defines a sensor or a microsensor to the resource monitoring and control (RMC) subsystem.

Syntax

To define a sensor:

`mksensor` [–n *host1*[,*host2*...] | –N { *node_file* | "–" }] [–i *seconds*] [–c *n*] [–e 0 | 1 | 2] [–u *user-ID*] [–h] [–v | –V] *sensor_name* ["]*sensor_command*["]

To define a microsensor:

`mksensor -m` [–n *host1*[,*host2*...] | –N { *node_file* | "–" }] [–i *seconds*] [–h] [–v | –V] *microsensor_name microsensor_module* [["]*microsensor_arguments*["]]

Description

The `mksensor` command defines a sensor resource to the resource monitoring and control (RMC) subsystem. A *sensor* is an RMC resource with attributes that you can monitor. You can use the event-response resource manager (ERRM) commands to set up monitoring of the sensor attributes. The response actions defined will run when a monitored sensor event occurs. This enables administrators to extend RMC monitoring capabilities without having to write a resource manager.

For sensors, the *sensor_command* parameter specifies the command or script that the sensor resource manager will run to set (and then later, update) the sensor attribute values. After the sensor attributes have been monitored, the sensor resource manager sets the attribute values. Then, at defined intervals, the sensor resource manager updates these values.

For microsensors, the *microsensor_module* parameter specifies the path name to the loadable module that the microsensor resource manager will call to set (and then later, update) the microsensor attribute values. After the microsensor attributes have been monitored, the microsensor resource manager sets the attribute values. Then, at defined intervals, the microsensor resource manager updates these values. Use the **-m** flag to create a microsensor.

Alternatively, you can use **chsensor** or **refsensor** to update the sensor or microsensor attribute values. The **lssensor** command displays values for sensor or microsensor attributes that you can set using a sensor command or a microsensor module, if the attributes are monitored. If the attributes are not monitored, **lssensor** does not display their values. To remove a sensor or a microsensor, use the **rmsensor** command.

The **mksensor** command runs on any node. To define a sensor or a microsensor on one or more nodes in a management domain or a peer domain, use the **-n** flag. Instead of specifying multiple node names using the **-n** flag, you can use the **-N** *node_file* flag to indicate that the node names are in a file. Use **-N "–"** to read the node names from standard input.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

A sensor consists of the following attributes, which can be set using a sensor command: :

Float32
The type float32 attribute for this sensor resource.

Float64
The type float64 attribute for this sensor resource.

Int32
The type int32 attribute for this sensor resource.

Int64
The type int64 attribute for this sensor resource.

Quantum
The type quantum attribute for this sensor resource.

String
The type string attribute for this sensor resource.

Uint32
The type uint32 attribute for this sensor resource.

Uint64
The type uint64 attribute for this sensor resource.

A sensor command sets attribute values by sending the values to standard output in a format that the sensor resource manager can parse. The format is *attr=value*. For example, if the sensor command sets the **Int32** attribute to 57, it writes **Int32=57** to standard output. To set more than one attribute value, the sensor command can write multiple *attr=value* pairs to standard output. The *attr=value* pairs can be on one or more lines. If the sensor command output is not in *attr=value* form, it is assumed to be a string and the value is placed in the **String** attribute.

The sensor command runs using the user ID that creates the sensor resource. Once a sensor resource is monitored, the sensor command is run at intervals specified by the **-i** flag, which is expressed in seconds. The default interval is 60 seconds. Specify a value of 0 to indicate that the sensor command is not to run at intervals. In this case, the **refsensor** command is typically used to update the sensor values.

Use the **-e** flag to control how the exit values from *sensor_command* are interpreted. Depending on this setting, when the exit value of the *sensor_command* is considered to be an error, the sensor attributes are not set and information is written to the audit log.

A microsensor consists of the following attributes, which can be set using a microsensor load module:

Float32
The type `float32` attribute for this microsensor resource

Float32Array
The type `float32` array attribute for this microsensor resource

Float64
The type `float64` attribute for this microsensor resource

Float64Array
The type `float64` array attribute for this microsensor resource

Int32
The type `int32` attribute for this microsensor resource

Int32Array
The type `int32` array attribute for this microsensor resource

Int64
The type `int64` attribute for this microsensor resource

Int64Array
The type `int64` array attribute for this microsensor resource

Quantum
The type `quantum` attribute for this microsensor resource.

String
The type `string` attribute for this microsensor resource.

StringArray
The type `string` array attribute for this microsensor resource.

UInt32
The type `uint32` attribute for this microsensor resource.

UInt32Array
The type `uint32` array attribute for this microsensor resource.

UInt64
The type `uint64` attribute for this microsensor resource.

UInt64Array
The type `uint64` array attribute for this microsensor resource.

The microsensor resource manager will make calls to the microsensor load module to set the values of the microsensor attributes. See the *Administering RSCT* for information about how to use microsensors.

Flags

–m
Specifies that the resource to be defined is a microsensor resource.

–n *host1*[,*host2*...]
Specifies one or more nodes on which the sensor should be defined. By default, the sensor is defined on the local node. This flag is only appropriate in a management domain or a peer domain.

–N { *node_file* | "–" }
Specifies that node names are read from a file or from standard input.

Use `-N` *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (#) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use `-N "–"` in a management domain or a peer domain to read the node names from standard input.

–i *seconds*
Specifies the interval at which a sensor command is run to update a sensor's attribute values or a microsensor module is run to update a microsensor's attribute valuesyup. *seconds*, which is an integer value, must be greater than or equal to `10`. The default interval is `60` seconds.

The sensor command is run at the specified interval only when the sensor resource is monitored. The microsensor module is run at the specified interval only when the microsensor resource is monitored. If the interval is set to `0`, the sensor command or microsensor module will not run automatically.

a

Using this flag is independent of using the `refsensor` command to refresh a sensor.

–c *n*
Specifies whether special handling is required for this sensor. *n* can be one of these values:

0
Indicates that no special handling is required. This is the default.

The sensor command will run at the interval that is defined for *sensor_name*. The sensor command will *not* run when monitoring begins or when the `lssensor` command is run.

1
Indicates that the sensor command will run when monitoring begins. The sensor command will also run at the interval that is defined for *sensor_name*. The sensor command will *not* run when the `lssensor` command is run.

Specfiying this value is not recommended, unless you expect the sensor command to run quickly. If the sensor command does not run quickly, it could block other requests to the sensor resource manager. These requests will not be processed until the sensor command finishes running.

2
Indicates that output from the command in the `SavedData` field is not saved permanently to `SavedData` persistent resource attributes. If this value is not specified, the sensor resource manager updates data in the registry's resource table whenever the command's standard output contains the line: `SavedData="`*any-string*`"`.

3
Indicates a combination of values 1 and 2.

4
Indicates that the sensor resource manager will run the sensor command after monitoring has stopped.

5
Indicates a combination of values 1 and 4.

6
Indicates a combination of values 2 and 4.

7
Indicates a combination of values 1, 2, and 4.

–e 0 | 1 | 2
Specifies how the sensor resource manager interprets the exit values of *sensor_command*, as follows:

0
No exit value from *sensor_command* is an error.

1
An exit value other than `0` from *sensor_command* is an error.

2
An exit value of `0` from *sensor_command* is an error.

The default value is `1`. The sensor attributes are not updated when the exit value is interpreted as an error. For an error, information is written to the audit log.

–u *user-ID*
Specifies the name of a user whose privileges will be used to run the sensor command. The user should already be defined on the system. The default value for *user-ID* is the user name that is associated with the current effective user ID.

–h
Writes the command's usage statement to standard output.

–v | –V
Writes the command's verbose messages to standard output.

Parameters

["]*microsensor_argument*["]
Specifies a string that will be passed to the microsensor module callback function. The microsensor resource manager will break the string into an array of strings based on blank characters in the microsensor argument. The microsensor argument cannot be changed once the microsensor is defined.

If the microsensor argument contains any blank characters or any special characters that can be interepreted by the shell, it must be enclosed in double quotation marks. When the microsensor argument is enclosed in double quotation marks, you must include a backslash escape character (\) before an "inner" double quotation mark. You must also include a \ before a dollar sign ($).

microsensor_module
Specifies the path name to the loadable microsensor module. A signature for the module is stored by the microsensor resource manager and is verified when the module is used. The microsensor module cannot be changed once the microsensor is defined.

microsensor_name
Specifies the name of the microsensor that is to be defined.

["]*sensor_command*["]
Specifies a command or script that the sensor resource manager will use to set the attribute values of the sensor. You should not call any of the sensor resource manager commands (`chsensor`, `lssensor`, `mksensor`, `refsensor`, or `rmsensor`) as part of this parameter.

If *sensor_command* contains any blank characters, or any special characters that can be interpreted by the shell, it must be enclosed in double quotation marks.

When *sensor_command* is enclosed in double quotation marks, you must include a backslash escape character (\) before an "inner" double quotation mark. You must also include a \ before a dollar sign ($). See Example 2 for more information.

sensor_name
Specifies the name of the sensor that is to be defined.

Security

To create sensors using this command, you need write permission for the `IBM.Sensor` resource class.

To create microsensors using this command, you need write permission for the `IBM.MicroSensor` resource class.

Permissions are specified in the access control list (ACL) file on the contacted system. See the *Administering RSCT* for details on the ACL file and how to modify it.

Exit Status

0
The command has run successfully.

1
An incorrect combination of flags and parameters has been entered.

n
Based on other errors that can be returned by the RMC subsystem.

Environment Variables

CT_CONTACT
When the `CT_CONTACT` environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If this environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled.

The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Restrictions

You should not call any of the sensor resource manager commands (`chsensor`, `lssensor`, `mksensor`, `refsensor`, or `rmsensor`) as part of the *sensor_command* parameter, as this could cause a deadlock.

Implementation Specifics

This command is part of the `rsct` fileset for the AIX operating system and `rsct-3.1.0.0-0.`*platform*`.rpm` package for the Linux, Solaris, and Windows platforms, where *platform* is `i386`, `ppc`, `ppc64`, `s390`, or `x86_64`.

Examples

1. To create a new sensor called `Sensor1` that runs the script `/usr/bin/updateSensor1`, which will update the sensor attributes every 30 seconds (once monitored), enter:

   ```
   mksensor -i 30 Sensor1 "/usr/bin/updateSensor1"
   ```

   The contents of `/usr/bin/updateSensor1` may be like:

   ```
   #!/usr/bin/perl
   my $int32 = some_fn_that_generates_i32_value;
   my $string = some_fn_that_generates_string_value;
   print "Int32=$int32 String=$string";
   exit 0;
   ```

   A sample condition could be:

   ```
   mkcondition -r IBM.Sensor -s "Name==Sensor1" -e "Int32 > 100" Sensor1Int32
   ```

   Using the response "E-mail root anytime", a start monitoring command may be:

   ```
   startcondresp Sensor1Int32 "E-mail root anytime"
   ```

2. To create a sensor called `Sensor1` with a *sensor_command* value of

   `df -m /var | sed '1d' | sed 's/%//g' | /bin/awk '{ print "Int32="$4}',`

   enter:

   ```
   mksensor Sensor1 "df -m /var | sed '1d' | sed 's/%//g' | /bin/awk \
   '{ print \"Int32=\"\$4}'"
   ```

   When *sensor_command* is enclosed in double quotation marks, you must include a backslash escape character (\) before an "inner" double quotation mark. You must also include a \ before a dollar sign ($). So in this example, the sensor command substring `"Int32="$4` becomes `\"Int32=\"\$4` when it is part of `mksensor` command.

3. To create a sensor called `Sensor3` that runs the `/usr/bin/checkhealth` script on the nodes that are listed in the `/u/joe/common_nodes` file, enter:

   ```
   mksensor -N /u/joe/common_nodes Sensor3 "/usr/bin/checkhealth"
   ```

   where `/u/joe/common_nodes` contains:

   ```
   # common node file
   #
   node1.myhost.com    main node
   node2.myhost.com    backup node
   ```

4. To create a microsensor called `IBM.msensorq` that uses the shared module `/usr/lib/msensors/msensorq` and requires the parameters `db=abc`, `confirm=yes`, `retry=yes`, and `mirror=no`, enter:

   ```
   mksensor -m IBM.msensorq /usr/lib/msensors/msensorq \
   "db=abc confirm=yes retry=yes mirror=no"
   ```

Location

/opt/rsct/bin/mksensor

mkserver Command

Purpose

Adds a subserver definition to the subserver object class.

Syntax

mkserver **-c** *CodePoint* **-s** *Subsystem* **-t** *Type*

Description

The **mkserver** command adds a subserver definition to the **Subserver** object class.

Flags

Item	Description
-c *CodePoint*	Specifies the *CodePoint* integer that identifies the subserver. This is the value by which the subsystem knows the subserver. The **mkserver** command is unsuccessful if this *CodePoint* value already exists for this subsystem. The limit for *CodePoint* storage is the same as a short integer (1 through 32,768).
-s *Subsystem*	Specifies the name that uniquely identifies the subsystem to which the subserver belongs. The **mkserver** command is unsuccessful if the *Subsystem* name is not known in the subsystem object class, or if the *Subsystem* name is that of a known subsystem in the subsystem object class but uses signals as its communication method.
-t *Type*	Specifies the name that uniquely identifies the subserver. The **mkserver** command is unsuccessful if the *Type* name is already known in the **Subserver Type** object class.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Auditing Events: If the auditing subsystem has been properly configured and is enabled, the **mkserver** command will generate the following audit record (event) every time the command is executed:

Event	Information
SRC_Addserver	Lists in an audit log subsystems that have been added and the entire Object Data Management record.

Examples

To add a subserver definition, enter:

```
mkserver  -s srctest  -t tester  -c 1234
```

This adds a subserver definition to the **Subserver Type** object class, with an owning subsystem of `srctest` and a subserver code point of 1234.

Files

Item	Description
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration object class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration object class.

mkslave Command

Purpose

Invokes the **ypinit** command to retrieve maps from an NIS controller server and starts the **ypserv** daemon to configure a worker server.

Syntax

/usr/sbin/mkslave [**-C** | **-c**] [**-O** | **-o**] [**-I** | **-B** | **-N**] *Master*

Description

The **mkslave** command invokes the **ypinit** command to retrieve maps from the controller server you specify on the command line. The **ypserv** daemon starts after the **ypinit** command has completed successfully. Use the *Master* parameter to specify the host name of the controller server. The controller server specified should already be configured and running.

You can use the System Management Interface Tool (SMIT) **smit mkslave** fast path to run this command.

Flags

Item	Description
-C	Invokes the **ypinit** command with the **-n** flag. The **mkslave** command continues on errors. This flag is the default.
-c	Stops execution when errors occur.
-O	Overwrites any maps that exist in the domain.
-o	Prevents the overwriting of maps that exist in the domain. This flag is the default.
-I	Invokes the **ypinit** command immediately but does not start the **ypserv** daemon until the next system reboot.
-N	Invokes the **ypinit** command and starts the **ypserv** daemon.
-B	Invokes the **ypinit** command, starts the **ypserv** daemon and configures the **ypserv** to start at system reboot. This flag is the default.

Examples

To invoke the **ypinit** command so that the controller server `host2` will be contacted for maps, enter:

```
mkslave -O host42
```

This command will overwrite the current maps.

Files

Item	Description
/var/yp/*DomainName* directory	Contains the NIS maps for the NIS domain.

mksmbcred Command

Purpose

Adds the credentials of the Server Message Block (SMB) client file system to the `/etc/smbcred` file. These credentials can be used for future mount operation of SMB client *shares*.

Syntax

```
mksmbcred -s server_name -u user_name [-p password]
```

Description

When you run the **mksmbcred** command, you must specify a server name and a username. If you do not specify a password as an argument with the **-p** flag, the **mksmbcred** command prompts for a password as a hidden character input. The password is encrypted, and the credentials are stored in the `/etc/smbcred` file as a *server_name*|*username*|*password* set. You can create multiple sets of credentials for the same server with different usernames. You can also create multiple sets of credentials with the same username for different servers.

Flags

-s *server_name*
Specifies the name of the remote host, which is an SMB server. The *server_name* parameter can be provided as a hostname, an IP address, or a fully qualified domain name.

-u *user_name*
Specifies the username for which credentials must be defined to access the specific remote host.

-p *password*
Specifies the password for a specific user of a specific remote host.

Exit status

0
The command completed successfully.

>0
An error occurred.

Example

To add credentials for `user1` to mount the SMB client file system on the `xxx.in.ibm.com` server, enter the following command:

```
mksmbcred -s xxx.in.ibm.com -u user1
```

Location

`/usr/sbin/mksmbcred`

Files

/etc/smbcred
Stores the credentials of the SMB client file system.

mkssys Command

Purpose

Adds a subsystem definition to the subsystem object class.

Syntax

mkssys { **-p** *Path* **-s** *Subsystem* **-u** *UserID* } [**-a** *Arguments*] [**-e** *StandardError*] [**-i** *StandardInput*] [**-o** *StandardOutput*] [**-t** *Synonym*] [**-O** | **-R**] [**-d** | **-D**] [**-q** | **-Q**] [**-K** | [**-I** *MessageQueue* **-m** *MessageMType* | **-f** *StopForce* **-n** *StopNormal* **-S**]] [**-E** *Nice*] [**-G** *Group*] [**-w** *Wait*]

Description

The **mkssys** command adds a new subsystem definition to the subsystem object class. If no flags are chosen after the **-p**, **-s**, and **-u** flags have been specified, the defaults are **-e /dev/console**, **-i /dev/console**, **-o /dev/console**, **-O**, **-d**, **-Q**, **-K**, **-E 20**, and **-w 20**.

> **Note:** Any auditing performed by the System Resource Controller (SRC) when actions are taken for the subsystem is logged against the login ID of the user who created the subsystem by using the **mkssys** command. For example, if you are logged in with root user authority, the subsystem is added with root user authority as the audit account.

Flags

Item	Description
-a *Arguments*	Specifies any arguments that must be passed to the command, started as the subsystem. These *Arguments* variables are passed by the SRC to the subsystem according to the same rules used by the shell. For example, quoted strings are passed as a single argument, and blanks outside a quoted string delimit arguments. Single and double quotes can be used.
-d	Specifies that inactive subsystems are displayed when the **lssrc -a** command (status all) request is made. By default, if the **-D** and **-d** flags are not present, the **-d** flag is used.
-D	Specifies that inactive subsystems are not displayed when status-all or status-group requests are made.
-e *StandardError*	Specifies where the subsystem *StandardError* data is placed. If the **-e** flag is not specified, the **/dev/console** file is used for standard error.
-E *Nice*	Changes the execution priority of the subsystem. Valid values are 0 through 39 (ordinary *Nice* variables are mapped to all positive numbers). If the **-E** flag is not present, the subsystem priority defaults to 20. Values between 0 and 19 are reserved for users with root authority.
-f *StopForce*	Specifies the signal sent to the subsystem when a forced stop of the subsystem is requested. Use only when the subsystem uses signals. The **mkssys** command is unsuccessful if the *StopForce* parameter is not a valid signal.
-G *Group*	Specifies that the subsystem belongs to the *Group* specified, and that the subsystem responds to all group actions on the *Group*.

Item	Description
-i *StandardInput*	Specifies where the subsystem standard input is routed. This field is ignored when the subsystem uses sockets communication. If the **-i** flag is not specified, by default the **/dev/console** file is used for standard input.
-I *MessageQueue*	Specifies that the subsystem uses message queues as the communication method. The *MessageQueue* variable specifies the message queue key for creating the message queue for the subsystem. Use the **ftok** subroutine with the subsystem path name as input to generate a unique key.
-K	Specifies that the subsystem uses sockets as its communication method. If a communication method is not specified, sockets communication is used by default.
-m *MessageMType*	Specifies the message type key the subsystem expects on packets sent to the subsystem by the SRC. Use only when the subsystem uses message queues communication.
-n *StopNormal*	Specifies the signal sent to the subsystem when a normal stop of the subsystem is requested. Use only when the subsystem uses signals communication. The **mkssys** command is unsuccessful if the *StopNormal* variable is not a valid signal.
-o *StandardOutput*	Specifies where the subsystem standard output is placed. If the **-o** flag is not specified, by default the **/dev/console** file is used for standard out.
-O	Specifies that the subsystem is not restarted if it stops abnormally. The default is no restart.
-p *Path*	Specifies the absolute path to the subsystem executable program.
-q	Specifies that the subsystem can have multiple instances running at the same time.
-Q	Specifies that multiple instances of the subsystem are not allowed to run at the same time and the subsystem is not to share the same interprocess communication (IPC) queue. If the **-q** flag is not specified, the **-Q** flag is the default.
-R	Specifies that the subsystem is restarted if the subsystem stops abnormally.
-s *Subsystem*	Specifies a name that uniquely identifies the subsystem. The **mkssys** command is unsuccessful if the subsystem name is already known in the subsystem object class.
-S	Specifies that the subsystem uses the signals communication method. You cannot define subservers for a subsystem name when your communication method is signals.
-t *Synonym*	Specifies an alternate name for the subsystem. The **mkssys** command is unsuccessful if the synonym name is already known in the subsystem object class.
-u *UserID*	Specifies the user ID for the subsystem. The *UserID* that creates the subsystem is used for security auditing of that subsystem.
-w *Wait*	Specifies the time, in seconds, allowed to elapse between a stop cancel (**SIGTERM**) signal and a subsequent **SIGKILL** signal. Also used as the time limit for restart actions. If the subsystem stops abnormally more than twice in the time limit specified by the *Wait* value, the subsystem is not automatically restarted. By default, if the **-w** flag is not present, the wait time default is 20 seconds.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Auditing Events: If the auditing subsystem has been properly configured and is enabled, the **mkssys** command will generate the following audit record (event) every time the command is executed:

Event	Information
SRC_Addssys	Lists in an audit log the name of the subsystem being added to the Object Data Manager (ODM) database and the entire ODM record.

Examples

1. To add a subsystem that uses sockets as its communication type, type the following:

   ```
   mkssys -s srctest -p /usr/lpp/srctest/srctest -u 0 -K
   ```

   This adds a subsystem definition to the subsystem object class, with a communication type of sockets, a user ID of `0` (root), and a subsystem name of `srctest`.
2. To add a subsystem that uses message queues as its communication type, type the following:

   ```
   mkssys -s srctest -p /usr/lpp/srctest/srctest -u 0 -I 123456 \ > -m 789
   ```

   This adds a subsystem definition to the subsystem object class, with a communication type of message queues, a message queue key of 123456, and a subsystem message type of 789.
3. To add a subsystem that uses signals as its communication type, type:

   ```
   mkssys -s srctest -p /usr/lpp/srctest/srctest -u 0 -S -n 30 \ > -f 31
   ```

   This adds a subsystem definition to the subsystem object class, with a communication type of signals, a stop normal signal of 30, a stop force signal of 31.
4. To add a subsystem that uses sockets as its communication type and is always passed an argument, type:

   ```
   mkssys -s srctest -p /usr/lpp/srctest/srctest -u 0 -a "-x"
   ```

   This adds a subsystem definition to the subsystem object class with a communication type of sockets and a command argument of `"-x"`.

Files

Item	Description
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration object class.
/dev/SRC	Specifies the **AF_UNIX** domain in the **socket.h** file.
/dev/.SRC-unix	Specifies the location for temporary file sockets.

mkstr Command

Purpose

Creates an error message file.

Syntax

mkstr [**-**] *MessageFile Prefix File* ...

Description

The **mkstr** command creates a file of error messages that can be removed from a single C source file or from multiple source files. Its use can reduce the size of programs that contain many error diagnostics and reduce system overhead in running such programs, because error messages are then not constantly swapped in and out of the source files.

The **mkstr** command processes each file specified by the *File* parameter, placing a massaged version of the file in a file having the name specified by the *Prefix* parameter followed by the original name.

To process the error messages in the source to the file specified by the *MessageFile* parameter, the **mkstr** command keys on the string **`error("** in the input stream. The string, starting at the **""** (two double quotation marks), is placed in the message file and followed by a null character and a new-line character. The null character terminates the message so it can be easily used when retrieved. The new-line character makes it possible to see the contents of the error message file by using the **cat** command.

The massaged copy of the input file then contains an **lseek** pointer into the file, which can be used to retrieve the message to its appropriate source file, as shown in the following example:

```
char efilname[] = "/usr/lib/pistrings";
int    efil = -1;

error(a1, a2, a3, a4)
{
    char buf[256];
    if (efil < 0) {
        efil = open(efilname, 0);
        if (efil < 0) {
oops:
           perror(efilname);
            exit(1);
        }
     }
    if (lseek(efil,(long) a1,0) <0||
         read(efil,buf, 256) <= 0)
        goto oops;
    printf(buf, a2, a3, a4);
}
```

Flags

Item	Description
-	The optional - (minus sign) causes the error messages to be placed at the end of the *MessageFile* for recompiling part of a large mkstr program.

Examples

1. To put the error messages from the current directory C source files into the file `pistrings` and to put processed copies of the source for these files into file names prefixed by `xx`, enter*:*

   ```
   mkstr pistrings xx *.c
   ```

2. To append the error messages from an additional source file into the file `pistrings`, enter:

   ```
   mkstr - pistrings xx newfile.c
   ```

Files

Item	Description
/usr/ccs/bin/mkstr	Contains the **mkstr** command.

mksysb Command

Purpose

Creates an installable image of the root volume group either in a file or onto a bootable tape.

Syntax

mksysb [**-a**] [**-A**] [**-b** *number*] [**-e**] [**-F** *filename*] [**-i**] [**-m**] [**-p**] [**-P**] [**-t** *argument*] [**-v**] [**-V**] [**-x** *file*] [**-X**] [**-Z**] [**-G** | **-N**] [**-M**] [**-T**] [**-C**] *device* | *file*

Description

The **mksysb** command creates a backup of the operating system (that is, the root volume group). You can use this backup to reinstall a system to its original state if it is corrupted. If you create the backup on tape or user defined file system (UDFS) capable media, the backup is bootable and includes the installation programs that are needed to install from the backup.

Note: If the system has a multibos environment where both instances are mounted, you can restore the backup only by using the **alt_disk_mksysb** command.

You can also use a **mksysb** image to restore another system.

The file system image is in backup-file format. The tape format includes a boot image, a bosinstall image, and an empty table of contents followed by the system backup (root volume group) image. The root volume group image is in backup-file format, starting with the data files and then any optional map files.

One of the data files that the **mksysb** command uses is the `/bosinst.data` file. If the `/bosinst.data` file does not exist, the `/var/adm/ras/bosinst.data` file is copied to / (root). The **mksysb** command always updates the `target_disk_data` stanzas in the `bosinst.data` file to match the disks currently in the root volume group of the system where the **mksysb** command is running.

If you are using a customized `/bosinst.data` file and do not want the `target_disk_data` stanzas that are updated, you must create `/save_bosinst.data_file`. The **mksysb** command does not update `/bosinst.data` if the `/save_bosinst.data_file` exists.

Notes:

1. When the **mksysb** command is running, ensure that system activity is minimal.
2. The image that the **mksysb** command creates does not include data on raw devices or in user-defined paging spaces.
3. If you are using a system with a remote-mounted `/usr` file system, you cannot reinstall your system from a backup image.
4. The **mksysb** command might not restore all device configurations for special features, such as `/dev/netbios` and some device drivers that are not shipped with the product.
5. The **mksysb** command uses the **backup** command to create an archive image. The **mksysb** command also saves the extended attributes (EA) format for any Enhanced Journaled File System (JFS2) that are being backed up. It uses the `/usr/bin/mkszfile` shell script to save this information.
6. If you remove the `/dev/ipldevice` before running the **mksysb** command, the `0301-150 bosboot` error occurs. This message, in most cases, can be ignored. Confirm the success of the **mksysb** command by the return code.
7. If you are creating a tape backup and have encrypted file systems, you must use the **-Z** flag. You cannot reinstall your system from a tape backup image that contains encrypted file systems.

To create a backup of the operating system to a CD, refer to the “mkcd Command” on page 2373. To create a backup of the operating system to a DVD, refer to the “mkdvd Command” on page 2418.

Flags

Item	Description
-a	Does not back up extended attributes or Network File System version 4 (NFS4) access control lists (ACLs).
-A	Backs up Data Management API (DMAPI) file system files.
-b *number*	Specifies the number of 512-byte blocks to write in a single output operation. When the **backup** command writes to tape devices, the default is 100 for backups by name. The write size is the number of blocks that are multiplied by the block size. The default write size for the **backup** command that writes to tape devices is 51200 (100 * 512) for backups by name. The write size must be an even multiple of the tape's physical block size.
-C	Specifies whether the `/usr/lpp/bos.alt_disk_install/boot_images/bosboot.disk.chrp` boot image can be replaced with a new boot image when you create the **mksysb** image. This flag should be used if the interim fixes that affect the kernel are installed on your system and if you plan to use the **alt_disk_mksysb** command to install the **mksysb** image. The **bos.alt_disk_install.boot_images** fileset must be installed on your system if you want to use the **–C** flag. You must specify the **–i** flag to build a new `image.data` file, when using the **–C** flag. The new `bosboot.disk.chrp` image is replaced by the original image at the end of the **mksysb** image creation. The **-C** flag is not available when you create the **mksysb** image when defining a NIM resource. A **multibos** Base Operating System (BOS) standby instance is not affected by the **-C** flag. **Note:** If required, the size of the `/usr` file system can be increased to include the new boot image by using the **-X** flag with the **-C** flag.

Item	Description
-e	Excludes files that are listed in the `/etc/exclude.rootvg` file from being backed up. The rules for exclusion follow the pattern matching rules of the **grep** command. If you want to exclude certain files from the backup, create the `/etc/exclude.rootvg` file, with an ASCII editor, and enter the patterns of file names that you do not want included in your system backup image. The patterns in this file are input to the pattern matching conventions of the **grep** command to determine which files will be excluded from the backup. If you want to exclude files that are listed in the `/etc/exclude.rootvg` file, select the `Exclude Files` field and press the Tab key once to change the default value to `yes`. For example: 1. To exclude all the contents of the directory that is called scratch, edit the exclude file to read as follows: `/scratch/` 2. To exclude the contents of the directory that is called `/tmp`, and avoid excluding any other directories that contain `/tmp` in the path name, edit the exclude file to read as follows: `^./tmp/` This excludes all contents of the **/tmp** directory, but the **/tmp** mount point for the file system is retained. It does not remove other directories or its contents such as **/var**, **/adm**, **/sw**, and **/tmp** file system. All files are backed up relative to **.** (current working directory). To exclude any file or directory for which it is important to have the search match the string at the beginning of the line, use the ^ (caret character) as the first character in the search string, followed by **.** (dot character), followed by the file name or directory to be excluded. If the file name or directory that is being excluded is a substring of another file name or directory, use the **^.** (caret character followed by dot character) to indicate that the search must begin at the beginning of the line and use the **$** (dollar sign character) to indicate that the search must end at the end of the line.
-F *filename*	Specifies a previously created **mksysb** image from which a backup tape is created. An attempt is made to make the backup tape bootable. Additionally, this flag must be used with a tape device.
-G	Excludes WPAR file systems from the system backup. This flag is not valid with **-N** flag.

Item	Description
-i	Calls the **mkszfile** command, which generates the /image.data file. The /image.data file contains details about volume groups, logical volumes, file systems, paging space, and physical volumes. This information is included in the backup for future use by the installation process. You must use the **-i** flag. Otherwise, an older /image.data file might be saved that does not contain adequate space requirements to restore the system backup. **Note:** Before you run the **mkszfile** command, ensure that enough space is available in the /tmp file to store a boot image. This space is needed during both backup and installation. To determine the amount of space that is needed in the /tmp file, enter the following command: `bosboot -q -a -d device` If you are using UDFS capable device named /dev/usbms0, you must specify /dev/cd0 as the device name because the /dev/usbms0 device is not supported by the **bosboot** command. If you use the **-X** flag with the **mksysb** command, you do not need to run the **bosboot** command to determine the amount of space needed in the /tmp file.
-m	Calls the **mkszfile** command, with the **-m** flag to generate map files. **Note:** The use of the **-m** flag causes the functions of the **-i** flag to be executed also.
-M	Creates a backup file that is intended for use with the **multibos** command. The **-M** flag backs up the **/**, **/usr**, **/var**, and **/opt** file systems. Do not use the backup to reinstall a system. You must install the bos.alt_disk_install.boot_images fileset at the same level as the system.
-N	Includes file systems that belong to a workload partition (WPAR) in the defined state in the system backup. **Note:** To be included in the backup, all file systems that belong to a WPAR in the defined state must be in the rootvg volume group.
-p	Disables software packing of the files as they are backed up. Some tape drives use their own packing or compression algorithms.
-P	Excludes files that are listed line by line in the /etc/exclude_packing.rootvg, /etc/exclude_packing.vgname, or /etc/exclude_packing.WPARname file from being packed. For example, to exclude the /etc/filesystems and /usr/bin/zcat file from being packed during the **mksysb** backup, edit the /etc/exclude_packing.*type* to add on consecutive lines /etc/filesystems and /usr/bin/zcat. In this case, the file /etc/exclude_packing.*type* must look like: `/etc/filesystems` `/usr/bin/zcat` The **-P** and **-p** flags are mutually exclusive.
-t *argument*	Specifies the path to the directory or file system that is used to create a boot image from the **mksysb** file that is specified by the -F flag. If the -t flag is not used with the -F flag, the boot image is created in the /tmp file by default. Approximately 100 MB of free space is required. After the boot image is created, this space is freed.

Item	Description
-T	Creates backup by using snapshots. This command applies only to JFS2 file systems. When you specify the **-T** flag to use snapshots for creating a volume group backup, external JFS2 snapshots are created. Snapshots allow for a point-in-time image of a JFS2 file system and thus, do not require a system to be put into a temporarily inactive state. The size of the snapshot is 2% - 15% of the size of the file system. The snapshot logical volumes are removed when backup is complete. However, snapshots are not removed if a file system already has other snapshots. Additionally, if a file system has internal snapshots, external snapshots cannot be created and thus, snapshots are not used for creating the backup of the file system. The use of the **-T** flag does not affect any JFS file systems that are present in the volume group that is being backed up. These file systems are backed up in the same manner as done previously. When you specify the **-T** flag, you must also specify the **-i** flag. If you do not specify the **-i** flag, an older `/image.data` file might not have adequate space requirements. Therefore, a failure might occur when you save data to the snapshot.
-v	Verbose mode. Lists files as they are backed up.
-V	Verifies a tape backup. This flag causes the **mksysb** command to verify the file header of each file on the backup tape and report any read errors as they occur.
-x *file*	Excludes the file systems that are listed in the file from the system backup. File system mount points must be listed one per line.
-X	Specifies to automatically expand the **/tmp** file system if necessary. The **/tmp** file system might need to be extended to make room for the boot image when creating a bootable backup to tape.
-Z	Specifies that the Encrypted File System (EFS) information for all the files, directories, and file systems is not backed up.The **-Z** flag is required if you have encrypted file systems and are creating a backup on a tape.

Attention: Use the **-x** flag with caution when you exclude the file systems from the backup of the operating system. The resulting backup might be unusable for system restoration.

Parameters

Item	Description
Device | *File*	Specifies the name of the device or file.

Exit Status

This command returns the following exit values:

Item	Description
0	The command completed successfully.
1	An error in the **mksysb** command use occurred.
2	An error in the **savevg** command use occurred. The **savevg** is a link to **mksysb**.
3	An error occurred before any file systems were mounted.
4	Exit because of trap.
5	Exit because there was no space.
6	Exit because a volume group name was not valid.

Examples

1. To generate a system backup and create an `/image.data` file (generated by the **mkszfile** command) to a tape device named `/dev/rmt0`, enter the following command:

   ```
   mksysb -i /dev/rmt0
   ```

2. To generate a system backup and create an `/image.data` file with map files (generated by the **mkszfile** command) to a tape device named `/dev/rmt1`, enter the following command:

   ```
   mksysb -m /dev/rmt1
   ```

3. To generate a system backup with a new `/image.data` file, but exclude the files in directory `/home/user1/tmp`, create the file `/etc/exclude.rootvg` containing the line `/home/user1/tmp/`, and enter the following command:

   ```
   mksysb -i -e /dev/rmt1
   ```

   This command backs up the `/home/user1/tmp` directory but not the files it contains.

4. To generate a system backup file named `/mksysb_images/node1` and a new `/image.data` file for that image, enter the following command:

   ```
   mksysb -i /mksysb_images/node1
   ```

   Note: This file is not bootable and can be installed only by using Network Installation Management (NIM).

5. After running the **mkszfile** command independently, to generate a system backup on the tape device `/dev/rmt0`, and then to verify the readability of file headers, enter the following command:

   ```
   mksysb  /dev/rmt0 -V
   ```

6. To generate a system backup file named `/mksysb_images/mksysb1` to be used with the **multibos** command, and to create an `/image.data` file for that image, enter the following command:

   ```
   mksysb -iM /mksysb_images/mksysb1
   ```

7. To generate a system backup and create an `/image.data` file (generated by the **mkszfile** command) to a UDFS capable device named `/dev/usbms0`, enter the following command:

   ```
   mksysb -i /dev/usbms0
   ```

   Note: For information backing up a volume group, see the **listvgbackup** command. To restore individual files from a volume group backup, see the **restorevgfiles** command.

Files

Item	Description
/usr/bin/mksysb	Contains the **mksysb** command.

mkszfile Command

Purpose

Saves the system state for reinstallation on the current system or another system.

Syntax

mkszfile [**-X**] [**-d** directory] [**-m**] [**-G**] [**-N**] [**-x** file]

Description

Attention: The **mkszfile** command overwrites an existing **/image.data** file with new information.

The **mkszfile** command saves the system state for reinstallation on the current system or on another system. The information saved includes the following:

- System installation information
- Logical volume information for the root volume group
- File system information.

The saved information allows the **bosinstall** routine to recreate the logical volume information as it existed before the backup.

The **mkszfile** command creates the **/image.data** file. The contents of this file are defined by the system in which the image was created. The user can edit the **/image.data** file before calling the **mksysb** command. The **mksysb** command, in turn, only backs up the file systems specified in the **/image.data** file, which reflects the requirements of the **rootvg** file system.

All the saved information is obtained using list commands. The commands are listed in the **/image.data** file as comments for the user's reference when editing this file.

Files on tape cannot be changed. However, in order to override the data files on the tape, the user can create a diskette with the desired files.

The **mkszfile** command checks to be sure there is at least 8MB of free space available in the **/tmp** file system for the boot image.

Note:

1. Before running the **mkszfile** command, ensure that enough space is available in the **/tmp** file to store a boot image. This space is needed during both backup and installation. To determine the amount of space needed in the **/tmp** file, issue one of the following commands: **bosboot -qad** *rmt* or **bosboot -qad** *ipldevice*.
2. If you remove the `/dev/ipldevice` prior to executing the **mkszfile** command, the 0301-150 bosboot error occurs. This message, in most cases, can be ignored. Confirm the success of the **mkszfile** command by the return code.

Flags

Item	Description
-m	Creates map files that specify the mapping of the logical-to-physical partitions for each logical volume in the volume group. This mapping can be used to allocate the same logical-to-physical mapping when the image is restored. The map file locations are stored in the MAPFILE field in the **/image.data** file for each logical volume. Sufficient space would exist in the **/tmp** file system for map creation because the installation routines place the maps in the **/tmp** file system before issuing the **mklv** command. For example, for the **hd7** logical volume, the location of the map file is **/tmp/vgdata/rootvg/hd7.map**. The MAPFILE field in the **/image.data** file for the **hd7** logical volume is under the entry `MAPFILE=/tmp/vgdata/rootvg/hd7.map`. The map files in the backup image are copied after the **/bosinst.data** and **/image.data** files.

Item	Description
-N	Includes file systems that belong to a workload partition (WPAR) in the defined state in the **/image.data** file. **Note:** To be included in the **/image.data** file, all file systems that belong to a WPAR in the defined state need to be in the rootvg volume group.
-X	Expands **/tmp** if needed.
-d	Write the **image.data** file to the specified directory instead of /.
-G	Excludes the WPAR file systems from the **/image.data** file. This flag is not valid with **-N** flag.
-x *file*	Excludes the file systems that are listed in the file from the **image.data** file. File system mount points must be listed one per line.

Note: Use care when excluding file systems as a resulting backup can be unusable for system restoration.

Files

Item	Description
/usr/bin/mkszfile	Contains the **mkszfile** command.

mktcpip Command

Purpose

Sets the required values for starting TCP/IP on a host.

Syntax

mktcpip { **-S** *Interface* | -h *HostName* **-a** *Address* **-i** *Interface* [**-s**] [**-m** *SubnetMask*] [**-r** *RingSpeed*] [**-t** *CableType*] [**-g** *DefaultGateway*] [**-n** *NameServerAddress* [**-d** *Domain*]] [[**-c** *Subchannel*] **-D** *Destination*] }

Description

The **mktcpip** command sets the required minimal values required for using TCP/IP on a host machine. These values are written to the configuration database.

Note: The **mktcpip** command currently supports IPv4 only.

The basic functions of the **mktcpip** command include:

- Setting the host name in both the configuration database and the running machine.
- Setting the IP address of the interface in the configuration database.
- Making entries in the **/etc/hosts** file for the host name and IP address.
- Setting the domain name and IP address of the nameserver, if applicable.
- Setting the subnetwork mask, if applicable.
- Adding a static route to both the configuration database and the running machine, if applicable.
- Starting the specified TCP/IP daemons.

You can use the System Management Interface Tool (SMIT) **smit mktcpip** fast path to run this command.

Flags

Item	Description
-a *Address*	Sets the Internet address of the host. Specify the address in dotted decimal notation. Each network interface on the host should have a unique Internet address. The following is the standard format for setting the Internet address: `127.10.31.2`
-c *Subchannel*	Specifies the subchannel address for a System/370 channel adapter.
-D *Destination*	Sets the destination address for a static route. Specify the address in dotted decimal notation. The following is the standard format for setting the destination address for a static route: `192.9.52.1`
-d *Domain*	Specifies the subdomain name that is used by the host machine. The subdomain name must be specified in the following format: `subdomain.subdomain.rootdomain`
-g *DefaultGateway*	Adds the default gateway address to the routing table. Specify the address in dotted decimal notation. The following is the standard format for setting the default gateway address: `192.9.52.0`
-h *HostName*	Sets the name of the host. If using a domain naming system, the domain and any subdomains must be specified. The following is the standard format for setting the host name: `hostname` The following is the standard format for setting the host name in a domain naming system: `hostname.subdomain.subdomain.rootdomain`
-i *Interface*	Specifies a particular network interface, for example: `tr0`
-m *SubnetMask*	Specifies the mask the gateway should use in determining the appropriate subnetwork for routing. The subnet mask is a set of 4 bytes, as in the Internet address. The subnet mask consists of high bits (1s) corresponding to the bit positions of the network and subnetwork address, and low bits (0s) corresponding to the bit positions of the host address.
-n *NameServerAddress*	Specifies the Internet address of the name server the host uses for name resolution, if applicable. The address should be entered in dotted decimal notation, as follows: `127.1.0.1`

Item	Description
-r *RingSpeed*	Specifies the ring speed for a token-ring adapter. Valid values for the *RingSpeed* variable are either 4- or 16-Mbps.
-S *Interface*	Retrieves information for System Management Interface Tool (SMIT) display.
-s	Starts the TCP/IP daemons.
-t *CableType*	Specifies cable size for Standard Ethernet or IEEE 802.3 Ethernet networks. Valid values for the *CableType* variable are **dix** for thick cable, **bnc** for thin cable, or **N/A** for Not Applicable. The **-t** *CableType* flag should be used only for Standard Ethernet (en) and IEEE 802.3 Ethernet (et) interfaces.

Examples

To set the required values for starting TCP/IP enter:

```
mktcpip -h fred.austin.century.com -a 192.9.200.4 -i en0 \
-n 192.9.200.1 -d austin.century.com -s
```

Note: Use the **mktcpip** command only to minimally configure TCP/IP for the first time. For further configuration changes, use the **smitty configtcp** fastpath.

Files

Item	Description
/usr/bin/mktcpip	Contains the **mktcpip** command.
/etc/resolv.conf	Contains the default system configuration database.
/etc/hosts	Contains the host name and IP address entries.

mkts Command

Purpose

Makes a thin server.

Syntax

mkts -i *ipaddress* **-m** *subnetmask* **-g** *gateway* [**-s** *speed*] [**-d** *duplex*] **-c** *cosi* [**-p** *size*] [**-H** | **-h**] [**-t**] [**-l**] [**-v**] [**-D**] *thinserver*

Description

The `mkts` command creates a thin server so that it can use the common image created with the `mkcosi` command. When a thin server is created, several directories are also created for the thin server to mount and use, including `/root`, `/dump`, `/home`, `/tmp`, `/shared_home`, and `/paging`. If you specify the `-l` flag when creating a thin server, the resulting thin server is a diskless client. That is, all resources are created on the server that calls the `mkts` command, except for the `/root` directory, which is created on the server storing the common image. However, if you do not specify the `-l` flag, the thin server is a dataless client. In this case, only the `/root` directory is created on the server storing the common image; all other directories are created locally on the thin server. If necessary, the 512 MB default size used for the paging can be changed by specifying a size value with the `-p` flag.

Flags

Item	Description
`-c` *cosi*	Specifies the common image for the thin server to obtain its operating system, which is required for the thin server to start up and run.
`-d` *duplex*	Specifies the duplex setting (optional). Use this setting to configure the client's network interface. This value can be full or half.
`-D`	Creates an iSCSI dump device in the Common Operating System Image (COSI) of the thin server. The name of the dump device is dump_*cosi_name*. In order to allow the **sysdumpdev** command to correctly set this new iSCSI dump device on the thin server, the thin server must boot in iSCSI mode from the COSI of the thin server.
`-g` *gateway*	Specifies the thin server gateway.
`-h`	Defines or uses home resource. The home resource is a network installation management (NIM) home resource. It is a directory that is created on a NIM master or any NIM resource server. The directory is exported to the thin server to be mounted and used. It is basically the thin server's **/home** directory.
`-H`	Defines or uses shared_home resource. The shared_home resource is a network installation management (NIM) resource. It is a directory that is shared among all thin servers. The directory is exported and mounted on the clients from the NIM master.
`-i` *ipaddress*	Specifies a thin server IP address or host name.
`-l`	Specifies whether local resources are used when configuring the thin server. If you specify this flag, all resources are created remotely from the thin server. If you do not specify this flag, only the `/root` resource is created remotely from the thin server, and all other resources are created locally on the thin server.
`-m` *subnetmask*	Specifies the thin server subnet mask.
`-p` *size*	Specifies the size (in megabytes) of the paging space for the thin server. The minimum size is 64 MB of paging space. The default size is 512 MB of the paging space. If you specify the size less than 64 MB, 512 MB is used.
`-s` *speed*	Specifies speed setting (optional). This is the communication speed to use when configuring the client's network interface. This value can be 10, 100, or 1000.
`-t`	Defines or uses the TMP resource.
`-v`	Enables verbose debug output when the `mkts` command runs.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Security

Access Control: You must have root authority to run the `mkts` command.

Examples

1. To define a thin server called `lobo` and have it use a common image called `cosi1` for its operating system with an IP address of 9.3.6.234, a subnet mask of 255.255.255.0, and a gateway of 9.3.6.1, enter:

```
mkts -i 9.3.6.234 -m 255.255.255.0 -g 9.3.6.1 -c cosi1 lobo
```

Location

`/usr/sbin/mkts`

Files

Item	Description
`/etc/niminfo`	Contains variables used by NIM.

mktun Command

Purpose

Activates tunnel(s).

Syntax

mktun [**-v 4**|**6**] [**-t** *tid_list*] [**-i**] [**-I**]

Description

Use the **mktun** command to activate tunnel(s). For IBM tunnels, this command initiates the security protocol exchanges between the local and the destination host.

Flags

Item	Description
-i	Initiation flag. If the **-i** flag is not used, all the tunnels in the tunnel database (or those listed with the **-t** flag) will be activated. If the **-i** flag is used, only the tunnels whose tunnel definitions status in the tunnel database "active" will be activated.
-I	If the **-I** flag is specified, manual tunnels will be activated.

Item	Description
-t	If the **-t** flag is specified, only the tunnel(s) that follows this flag will be activated. If the **-t** flag is not used, all tunnel(s) currently defined in the tunnel database will be activated. The *tid_list* can be a single tunnel ID or a sequence of tunnel IDs separated by "**,**" or "**-**" (1, 3, 5-7).
-v	The IP version of the tunnels to be activated. The value of **4** specifies IP version 4 tunnels. The value of **6** specifies IP version 6 tunnels. If the **-v** flag is not used, all tunnels for IP version 4 and IP version 6 will be activated.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

mkuser Command

Purpose

Creates a new user account.

Syntax

mkuser [**-R** *load_module*] [**-a**] [*Attribute*=*Value* ...] *Name*

Description

The **mkuser** command creates a new user account. The *Name* parameter must be a unique string (whose length is administrator-configurable using the chdev command) . You cannot use the **ALL** or **default** keywords in the user name. By default, the **mkuser** command creates a standard user account. To create an administrative user account, specify the **-a** flag.

To create a user with an alternate Identification and Authentication (I&A) mechanism, you can use the **-R** flag to specify the I&A load module. If you create users without the **-R** flag, you create the users locally. Load modules are defined in the **/usr/lib/security/methods.cfg** file.

The **mkuser** command does not create password information for a user. It initializes the **password** field with an * (asterisk). Later, this field is set with the **passwd** or **pwdadm** command. New accounts are disabled until the **passwd** or **pwdadm** commands are used to add authentication information to the **/etc/security/passwd** file.

You can use the System Management Interface Tool (SMIT) **smit mkuser** fast path to run this command.

The **mkuser** command always checks the target user registry to make sure the ID for the new account is unique to the target registry. You can also configure the **mkuser** command to check all user registries of the system using the **dist_uniqid** system attribute. The **dist_uniqid** system attribute is an attribute of the **usw** stanza of the **/etc/security/login.cfg** file, and can be managed using the **chsec** command.

The `dist_uniqid` system attribute has the following values:

- `never` - Does not check for ID collision against the non-target registries. This is the default setting.
- `always` - Checks for ID collision against all other registries. If collision is detected between the target registry and any other registry account creation or modification fails.
- `uniqbyname` - Checks for ID collision against all other registries. Collision between registries is allowed only if the account to be created has the same name as the existing account.

Note: ID collision detection in the target registry is always enforced regardless of the `dist_uniqid` system attribute.

The **uniqbyname** system attribute setting works well against two registries. With more than two registries, and with ID collision already existing between two registries, the behavior of the **mkuser** command is unspecified when creating a new account in a third registry using colliding ID values. The new account creation might succeed or fail depending the order in which the registries are checked.

The check for ID collision only enforces ID uniqueness between the local registry and remote registries or between remote registries. There is no guarantee of ID uniqueness between the newly created account on the remote registry and existing local users on other systems that make use of the same remote registry. The **mkuser** command bypasses a remote registry if the remote registry is not reachable at the time the command is run.

Restrictions on Creating User Names

To prevent login inconsistencies, you should avoid composing user names entirely of uppercase alphabetic characters. While the **mkuser** command supports multi-byte user names, it is recommended that you restrict user names to characters with the POSIX portable filename character set.

To ensure that your user database remains uncorrupted, you must be careful when naming users. User names must not begin with a - (dash), + (plus sign), @ (at sign), or ~ (tilde). You cannot use the keywords **ALL** or **default** in a user name. Additionally, do not use any of the following characters within a user-name string:

Item	Description
:	Colon
"	Double quote
#	Pound sign
,	Comma
=	Equal sign
\	Back slash
/	Slash
?	Question mark
'	Single quote
`	Back quote

Finally, the *Name* parameter cannot contain any space, tab, or new-line characters.

Flags

Item	Description
-a	Specifies that the user is an administrator. Only the root user can use this flag or alter the attributes of an administrative user.
username	Specifies that the user is a new user.
-R *load_module*	Specifies the loadable I&A module used to create the user.

Parameters

Item	Description
Attribute=*Value*	Initializes a user attribute. Refer to the **chuser** command for the valid attributes and values.
Name	Specifies a unique string. The length of this string is set by an administrator by using the chdev command.

Exit Status

This command returns the following exit values:

Item	Description
0	The command runs successfully and all requested changes are made.
>0	An error occurred. The printed error message lists further details about the type of failure.

Security

Access Control: This command should grant execute (x) access only to the root user and members of the security group. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the root user with the **setuid** (SUID) bit set.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

To get the full functionality of the command, besides the **accessauths**, the role should also have the following authorizations:

- **aix.security.user.audit**
- **aix.security.role.assign**
- **aix.security.group.change**
- **aix.security.user.change**

Files Accessed:

Mode	File
rw	**/etc/passwd**
rw	**/etc/security/user**
rw	**/etc/security/user.roles**
rw	**/etc/security/limits**
rw	**/etc/security/environ**
rw	**/etc/group**
rw	**/etc/security/group**
r	**/usr/lib/security/mkuser.default**
x	**/usr/lib/security/mkuser.sys**

Auditing Events:

Event	Information
USER_Create	user

Limitations

Creating a user may not be supported by all loadable I&A modules. If the loadable I&A module does not support creating a user, an error is reported.

Examples

1. To create the `davis` user account with the default values in the **/usr/lib/security/mkuser.default** file, type:

   ```
   mkuser davis
   ```

2. To create the `davis` account with `davis` as an administrator, type:

   ```
   mkuser -a davis
   ```

   Only the root user or users with the UserAdmin authorization can create `davis` as an administrative user.

3. To create the `davis` user account and set the **su** attribute to a value of `false`, type:

   ```
   mkuser su=false davis
   ```

4. To create the `davis` user account that is identified and authenticated through the LDAP load module, type:

   ```
   mkuser -R LDAP davis
   ```

Error Codes

Item	Description
0	The command is successful.
EINVAL	The username argument is not valid (containing characters not valid).
EACCES	The invoker does not have write access to the database files.
EPERM	The user identification and authentication fails if the **-a** flag is specified and the invoker is not root.
EEXIST	The user already exists.
ENAMETOOLONG	The user name is too long.
other **errno**	There are other system errors.

Files

Item	Description
/usr/bin/mkuser	Contains the **mkuser** command.
/usr/lib/security/mkuser.default	Contains the default values for new users.
/etc/passwd	Contains the basic attributes of users.
/etc/security/user	Contains the extended attributes of users.
/etc/security/user.roles	Contains the administrative role attributes of users.

Item	Description
/etc/security/passwd	Contains password information.
/etc/security/limits	Defines resource quotas and limits for each user.
/etc/security/environ	Contains the environment attributes of users.
/etc/group	Contains the basic attributes of groups.
/etc/security/group	Contains the extended attributes of groups.
/etc/security/.ids	Contains standard and administrative user IDs and group IDs.

mkuser.sys Command

Purpose

Customizes a new user account.

Syntax

mkuser.sys *Directory User Group Shell*

Description

The **mkuser.sys** command customizes the new user account specified by the *User* parameter. The **mkuser** command calls the **mkuser.sys** command after it has created and initialized the new account. The **tsm**, **login**, and **getty** commands and the **pam_mkuserhome** module call the **mkuser.sys** command at your login time if you do not have a home directory already.

The program as shipped creates the home directory specified by the *Directory* parameter, with the owner specified by the *User* parameter, the primary group specified by the *Group* parameter, and a copy of the appropriate profile for the user's shell. The shipped program can be replaced at installation by another program to customize local new-user creation. The installation-specific program should adhere to the error conventions of the supplied program.

Note: The shipped **mkuser.sys** file must not be customized directly. If a customized version is required, a new file **/etc/security/mkuser.sys.custom** must be created. The **mkuser.sys** program detects this new program and if it is present on the system, it runs it instead of the original **mkuser.sys**. The shipped **mkuser.sys** file is now a non-volatile file and must not be modified. The installation-specific program must adhere to the error conventions of the supplied program.

Security

Access Control: This command should grant read (r), write (w), and execute (x) access for the root user and members of the security group.

Files Accessed:

Mode	File
r	**/etc/passwd**
r	**/etc/security/user**

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/usr/lib/security/mkuser.sys	Contains the **mkuser.sys** command.

Note: You cannot use the **/etc/security/mkuser.sys** file to edit with the **chuser** , and **rmuser** commands. To assign default attributes such as *primary group*, *home directory* , and *login shell* to a user, use the **/etc/security/mkuser.default** file.

mkusil Command

Purpose

Creates or attaches a new user-specified installation location (USIL) instance.

Syntax

mkusil -R *RelocatePath* **-c** *Comments* [**-XFa**]

Description

The **mkusil** command creates or attaches a new USIL instance.

A user-specified installation location (USIL) is a tracked, relocated installation path that is created by the administrator. The location is tracked by the system and can be used as an alternate installation path for packages that use relocation of file sets or software. Multiple instances or versions of the same software package can be installed on a single system by delegating each installation to a separate USIL. An existing USIL instance can be attached or detached from any given system.

Each USIL instance maintains its own set of Software Vital Product Data (SWVPD) in three **installp** parts:

- *InstallRoot***/etc/objrepos**
- *InstallRoot***/usr/lib/objrepos**
- *InstallRoot***/usr/share/lib/objrepos**

Tip: Current SWVPD object classes include product, lpp, inventory, history, fix, vendor, and lag. Each USIL instance mirrors the default SWVPD structure within the relocated path.

Flags

Item	Description
-a	Attaches an existing installation as a USIL instance.
-c *Comments*	Specifies the comments to include in the USIL definition.
-F	Overwrites the existing USIL SWVPD in the target path without prompting you. It is appropriate to use this flag for USIL SWVPDs that are detached or removed by the **rmusil** command.
-R *RelocatePath*	Specifies the path to a new USIL location, which must be a valid directory. The directory cannot belong to a file system with the CIO mount option.
-X	Expands the space needed automatically.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/usr/sbin/mkusil	Contains the *mkusil* command.

mkvg Command

Purpose

Creates a volume group.

Syntax

mkvg [**-B**] [**-t** *factor*] [**-S** [**-v** *logicalvolumes*] [**-P** *partitions*]] [**-C**] [**-G**] [**-f**] [**-i**] [**-I**][**-c**] [**-X** none | SSD] [**-L** *ltgsize*] [**-n**] [**-s** *size*] [**-V** *majornumber*] [**-y** *volumegroup*] [**-M** **y**|**s**] [**-p** *mirrorpool*] [**-O** **y** | **n**] [**-N** *o*/*n*] [**-r** *y*/*n*] [**-e** **y**|**n**] [**-k** **y** | **n**] *physicalvolume ...*

Description

The **mkvg** command creates a new volume group, by using the physical volumes that are represented by the *physicalvolume* parameter. After creating the volume group, the **mkvg** command automatically varies on the new volume group by using the **varyonvg** command. The exception to this fact is when the volume group is created with the **-C** flag. When the volume group is successfully created, the volume group is not varied on automatically. Instead, the user must manually **varyon** the volume group.

The **mkvg** command by default creates a volume group that can accommodate 255 logical volumes and 32 physical volumes (disks). These limits can be extended by specifying either the **-B** or **-S** flag.

The **mkvg** command attempts to determine a proper partition size (**-s**) and factor (**-t**) if none is specified on the command line.

Note:

1. The physical volume is checked to verify that it is not already in another volume group. If the **mkvg** command determines that the physical volume belongs to a volume group that is varied on, it exits without creating the volume group. If the **mkvg** command determines the physical volume belongs to a volume group that is not varied on, the force option (**-f**) must be used to create the volume group. When using the force option, the previous contents of the physical volume are lost, so the user must use caution when using the force option.
2. To use this command, you must either have root user authority or be a member of the **system** group.
3. When creating the default volume group type (with a maximum of 32 PVs) or the big volume group type (with a maximum of 128 PVs), there is a limitation of 1016 physical partitions per PV. When specifying the physical partition size (**-s**), make sure that the value is set large enough so that 1016 physical partitions per PV limit is not violated. For example, a partition size of at least 16 MB would be needed to create a volume group with a 10-GB disk. Using a factor size (**-t**) of 2, a smaller partition size of 8 MB can be used. If a factor value is specified, the maximum number of PVs that can be included in the volume group is MaxPVs/factor.
4. Whenever you create a volume group, the operating system automatically does a varyon. However, if you create a volume group with the **-C** flag, the system does not autovaryon the volume group at the

end of the Concurrent Capable volume group creation. Instead, the **mkvg** command notifies you to manually **varyonvg** the volume group in either non-concurrent or concurrent mode.

5. This command fails to add a disk to the volume group if the disk indicates that it is managed by a third-party volume manager. To override and clear the disk of the third-party volume manager use **chpv -C** *HDiskName*.
6. Only Enhanced Concurrent Capable volume groups are created when the **-c** or **-C** flags are specified.
7. You must not mix 4 KB block physical volumes (PV) with PV blocks of other sizes. The block size of all PVs in the volume group must be the same. You can not import the volume group that is created with 4 KB block PVs on a version of AIX that does not support 4 KB block PVs.
8. Bad block relocation policy of a volume group is not supported on a volume group that is created with 4 KB block PVs.

Because the VGDA space has been increased substantially, every VGDA update operation (creating a logical volume, changing a logical volume, adding a physical volume, and so on) might take considerably longer to run.

You can use the System Management Interface Tool (SMIT) **smit mkvg** fast path to run this command.

Flags

Item	Description
-B	Creates a **big** type of volume group. This type can accommodate up to 128 physical volumes and 512 logical volumes. **Note:** Because the VGDA space has been increased substantially, every VGDA update operation (creating a logical volume, changing a logical volume, adding a physical volume, and so on) might take considerably longer to run.
-c	Same as **-C** flag. Only Enhanced Concurrent Capable volume groups are created.
-C	Creates an Enhanced Concurrent Capable volume group. Only use the **-C** flag with the PowerHA SystemMirror enhanced scalability (ES). It is not usable on volume groups and systems that do not use the PowerHA SystemMirror ES product. Use this flag to create an Enhanced Concurrent Capable volume group. **Note:** 1. Enhanced Concurrent volume groups use Group Services. Group Services ships with PowerHA SystemMirror ES and must be configured before activating a volume group in this mode. 2. Only Enhanced Concurrent Capable volume groups are supported when running with a 64-bit kernel. Concurrent Capable volume groups are not supported when running with a 64-bit kernel. 3. Enhanced Concurrent Capable volume groups have multinode varyon protection enabled. See the **-N** flag for details about multinode varyon protection.

Item	Description
-e *y*|*n*	Enables the `Critical PVs` option of the volume group. This flag is available in IBM AIX 7.2 with Technology Level 1, or later. **y** Enables the `Critical PVs` option of the volume group. If write request failures occur in the mirrored logical volume, the PV is marked as missing and it stops sending I/O requests to the failed mirrored logical volume. If the `Critical PVs` option is enabled in a volume group, you can import only the volume group into IBM AIX 7.2 with Technology Level 1, or later. **n** The `Critical PVs` option is not used. This is the default value.
-f	Forces the volume group to be created on the specified physical volume unless the physical volume is part of another volume group in the Device Configuration Database or a volume group that is active.
-G	Same as **-B** flag.
-i	Reads the *PhysicalVolume* parameter from standard input.
-I	Creates a volume group that can be imported to AIX Version 6.1. The *LTGSize* behaves as if the volume group had been created before AIX Version 6.1. If the logical volumes are later created with a strip size that is larger than the supported strip size on AIX Version 6.1 or AIX Version 6.1 (a strip size that is multiplied by the number of disks in an array equals the stripe size), then attempting to import the volume group back to AIX Version 6.1 or AIX Version 6.1 is not supported.
-k y|**n**	Enables the data encryption option in the volume group. The **-k** flag is available in IBM AIX 7.2 with Technology Level 5, or later. You can specify the following values for this flag: **y** Enables the data encryption option in the volume group. If the data encryption option is enabled in a volume group, you can import the volume group into an AIX LPAR that is running AIX 7 with 7200-05, or later. **n** Does not enable the data encryption option in the volume group. This is the default value.
-L *ltgsize*	For volume groups created on AIX Version 6.1 without the **-I** flag, the **-L** flag is ignored. When the volume group is varied on the logical track group size is set to the common maximum transfer size of the disks. For volume groups created on AIX Version 6.1 with the **-I** flag or for volume groups that are created before AIX Version 6.1, the logical track group size is set to the *ltgsize*, which must be 128, 256, 512, or 1024. In addition, it must be less than or equal to the maximum transfer size of all disks in the volume group. The default *ltgsize* is 128-KB.
-M y|**s**	Enables mirror pool strictness for the volume group. **y** Mirror pools must be used on each logical volume in the volume group. **s** Super-strict mirror pools are enforced on this volume group.

Item	Description
-N *o*|*n*	**o** Creates a volume group that is allowed to varyon in non-concurrent mode in more than one node at the same time. This is the default value. **n** Creates a volume group that is not allowed to varyon in non-concurrent mode in more than one node at the same time. This volume group can no longer be imported on a version of the AIX operating system that does not support the **-N** flag.
-n	Specifies that the volume group is not automatically available during a system restart. The default value activates the volume group automatically.
-O *y / n*	Enables the infinite retry option of the logical volume. **n** The infinite retry option of the logical volume is not enabled. The failing I/O of the logical volume is not retried. This is the default value. **y** The infinite retry option of the logical volume is enabled. The failed I/O request is retried until it is successful. **Note:** The infinite retry option is not supported in the Geographic Logical Volume Manager (GLVM) environment.
-p *mirrorpool*	Assigns each of the physical volumes that are being added to the specified mirror pool. After mirror pools are enabled in a volume group, the volume group can no longer be imported into a version of AIX that does not support mirror pools.
-P *partitions*	Total number of partitions in the volume group, where the *Partitions* variable is represented in units of 1024 partitions. Valid values are 32, 64, 128, 256, 512 768, 1024 and 2048. The default is 32 k (32768 partitions). The **chvg** command can be used to increase the number of partitions up to the maximum of 2048 k (2097152 partitions). This option is only valid with the **-S** option.
-r *y*|*n*	Enables the `Critical VG` option of the volume group. The **-r** flag can have the following values: **y** The `Critical VG` option of the volume group is enabled. If the volume group is created with the `Critical VG` option turned on, any I/O request failure starts writing the logical volume manager (LVM) metadata to check the state of the disk before returning the I/O failure. If the rootvg volume group is set to the `Critical VG` option and if the volume group looses access to quorum set of disks (or all disks if quorum is disabled), instead of forcing the volume group to an offline state, the node crashes and a message is displayed on the console. **n** The `Critical VG` option of the volume group is not enabled. It is the default value.

Item	Description
-S	Creates a **scalable** type of volume group. By default, this volume group can accommodate up to 1024 physical volumes, 256 logical volumes, and 32768 physical partitions. To increase the number of logical volumes, use the **-v** option. To increase the number of physical partitions, use the **-P** option. **Note:** Increasing **maxlvs** and **maxpps** beyond the default values for a scalable volume group can significantly increase the size of the VGDA proportionately. The **maxlvs** and **maxpps** values must be increased only as needed because they cannot be decreased. Meanwhile, as the VGDA space increases all VGDA update operations (creating a logical volume, changing a logical volume, adding a physical volume, and so on) can take longer and longer to run.
-s *Size*	Sets the number of megabytes in each physical partition, where the *Size* variable is expressed in units of megabytes from 1 (1 MB) through 131072 (128 GB). The *Size* variable must be equal to a power of 2 (example 1, 2, 4, 8). The default value for 32 and 128 PV volume groups is the lowest value to remain within the limitation of 1016 physical partitions per PV. The default value for scalable volume groups is the lowest value to accommodate 2040 physical partitions per PV.
-t *factor*	Changes the limit of the number of physical partitions per physical volume, which is specified by *factor*. The *factor* must be 1 - 16 for 32 PV volume groups and 1 and 64 for 128 PV volume groups. The maximum number of physical partitions per physical volume for this volume group changes to *factor* x 1016. The default is the lowest value to remain within the physical partition limit of *factor* x 1016. The maximum number of PVs that can be included in the volume group is MaxPVs/factor. The **-t** option is ignored with the **-S** option.
-V *majornumber*	Specifies the major number of the volume group that is created.
-v	Number of logical volumes that can be created. Valid values are 256, 512, 1024, 2048 and 4096. The default is 256. The **chvg** command can be used to increase the number of logical volumes up to the maximum of 4096. This option is only valid with the **-S** option. The last logical volume is reserved for metadata.
-X none|**SSD**	Enables PV type restriction for the volume group. This option allows a volume group to be created with a specific restriction based on the PV type. "none" is the default value. "SSD" requires that all PVs in the volume group must be SSD media type PVs. When the PV restriction is turned on, the **mkvg** command verifies that all PVs meet this condition. Once a PV restriction is turned on, the volume group can no longer be imported on a version of AIX that does not support PV type restrictions. **none** As an option, there is no PV restriction. volume group can be formed on any disk type. This is the default value. **SSD** As an option, volume group is restricted to SSD PV types. Disks that are listed in physical volume argument must be of type SSD.

Item	Description
-y *volumegroup*	Specifies the volume group name rather than having the name generated automatically. Volume group names must be unique system wide and can range from 1 to 15 characters. The name cannot begin with a prefix already defined in the **PdDv** class in the Device Configuration database for other devices. The volume group name that is created is sent to standard output. The volume group name can only contain only the following characters: "A" through "Z," "a" through "z," "0" through "9," or "_" (the underscore), "-" (the minus sign), or "." (the period). All other characters are considered invalid.

Security

Note:

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To create a volume group that contains 3 physical volumes with partition size set to 1 megabyte, type:

   ```
   mkvg  -s 1 hdisk3 hdisk5 hdisk6
   ```

   The volume group is created with an automatically generated name, which is displayed and available at system restart time.

   ```
   mkvg -s 2 -t 2 -y newvg hdisk1
   ```

   The volume group newvg is created with a physical partition size of 2-MB and maximum number of physical partitions per physical volume of 2032. The configuration mentioned in the example means that the size of hdisk1 can not be larger than 4064-MB (2032*2)

2. To create a volume group that can accommodate a maximum of 1024 physical volumes and 2048 logical volumes, type:

   ```
   mkvg -S -v 2048 hdisk6
   ```

Files

Item	Description
/usr/sbin	Directory where the **mkvg** command resides.
/tmp	Directory where the temporary files are stored while the command is running.
/dev	Directory where the character device entry for the volume group is created.

mkvgdata Command

Purpose

Creates a file containing information about a volume group for use by the **savevg** and **restvg** commands.

Syntax

mkvgdata [**-X**] [**-m**] [**-x** *file*] *VGName*

Description

The **mkvgdata** command creates a file containing information about a volume group for use by the **savevg** and **restvg** commands. The information includes the list of logical volumes, file systems and their sizes, and the volume group name. One of the following files is created, depending on the type of volume group:

Item	Description
/image.data	Created for information about the root volume group (**rootvg**). The **savevg** command uses this file to create a backup image that can be used by the **bosinstall** routine to reinstall the volume group to the current system or to a new system. The **mkvgdata** command overwrites this file if it already exists. The **/image.data** file is located in the **/** directory.
*vgname***.data**	Created for information about a user volume group. The *vgname* variable reflects the name of the volume group. The **savevg** command uses this file to create a backup image that can be used by the **restvg** command to reinstall the user volume group. The **mkvgdata** command overwrites this file if it already exists. The *vgname***.data** file is located in the **/tmp/vgdata/***vgname* directory, where *vgname* is the volume group name.

The information in either of these files can be edited by the user before issuing the **savevg** command.

Flag

Item	Description
-m	Creates map files that specify the mapping of the logical-to-physical partitions for each logical volume in the volume group. This mapping can be used to allocate the same logical-to-physical mapping when the image is restored. The map file locations are stored in the MAPFILE field in the **/image.data** file for each logical volume. Sufficient space would exist in the **/tmp** file system for map creation because the installation routines place the maps in the **/tmp** file system before issuing the **mklv** command. For example, for the **hd7** logical volume, the location of the map file is **/tmp/vgdata/rootvg/hd7.map**. The MAPFILE field in the **/image.data** file for the **hd7** logical volume is under the entry `MAPFILE=/tmp/vgdata/rootvg/hd7.map`. The map files in the backup image are copied after the **image.data** or *vgname***.data** files.
-X	Expands **/tmp** if needed.
vgname	Name of volume group to backup.
-x *file*	Excludes the file systems that are listed in the file from the output file. File system mount points must be listed one per line.

Note: Use care when excluding file systems because a resulting backup can be unusable for system restoration.

Files

Item	Description
/image.data	Created when the volume group is **rootvg**.
/tmp/vgdata/*vgname***/***vgname***.data**	Created when the volume group is not **rootvg** and where *vgname* is the name of the volume group.

mkvirprt Command

Purpose

Makes a virtual printer.

Syntax

mkvirprt [**-A** *AttachmentType*] **-d** *QueueDevice* **-n** *Device* **-q** *PrintQueue* **-s** *DataStream* **-t** *PrinterType* **-T**

mkvirprt -A *AttachmentType*

Description

The **mkvirprt** command creates a virtual printer definition and assigns it to the specified print queue and queue device. A virtual printer definition is a set of attribute values that describe a particular data stream for a particular printer. Before a print job can be queued with the **enq** command, **qprt** command, **lp** command, or **lpr** command, a virtual printer definition must be created for the printer's print queue and queue device.

Printers that support only one printer data stream, such as the 4201-3 Proprinter III, need only one virtual printer defined. Printers that support multiple printer data streams, such as the IBM 4216-31 Page Printer II, need a virtual printer defined for each data stream.

To create a virtual printer definition for a printer attached to an ASCII terminal, use the **-T** flag with the **mkvirprt** command.

After a virtual printer definition is created, its attribute values can be displayed with the **lsvirprt** command and changed with the **chvirprt** command.

The **mkvirprt** command becomes interactive if only the -A flag is specified with the command. Prompts are issued requesting the necessary parameter values. Prerequisite spooler queues and spooler queue devices are generated automatically, and all virtual printer definitions needed for the printer are defined with a single invocation of the **mkvirprt** command for the specified attachment type.

When the first prompt asks for a device name, if the device name entered is not that of a printer, or if an * (asterisk) precedes the device name, a list of printers is displayed. Otherwise, the printer type is assumed to be the same as that of the device.

Also, when a prompt asks for a print queue name, the queue name entered may optionally be followed by a colon and a queue device name. If no queue device name is provided, the queue device name is assumed to be the same as the device name.

> **Note:** Queue and device names must begin with an alphabetic character.

You can use the System Management Interface Tool (SMIT) **smit mkvirprt** fast path to run this command.

Flags

Item	Description
-A *AttachmentType*	Specifies the type of printer attachment. The most common values for the *AttachmentType* variable value are: **Attachment Type** Represents **local** Locally connected printers **remote** Remote print queues **ascii** Printers attached to an ASCII terminal **file** Print output redirected to a regular file. This flag is optional, and if the **-A** flag is not specified the default attachment type is **file**. If the **-A** flag is the only flag specified on the command line, the **mkvirprt** command goes into interactive mode and executes steps specified in the corresponding **.config** file.
-d *QueueDeviceName*	Specifies the name of an existing queue device to which the virtual printer is assigned.
-n *DeviceName*	Specifies the name of the printer device. Device names include **lp0** for printer 0, **lp1** for printer 1, and so on.
-q *PrintQueueName*	Specifies the special file name of an existing print queue to which the virtual printer is to be assigned. Note that you do not have to specify the path name to the file, such as the **/dev/lp0** file, you just need to specify **lp0**.
-s *DataStreamType*	Specifies the printer data stream type. Data stream types include: **Type** Description **asc** Extended ASCII **ps** PostScript **pcl** Hewlett-Packard PCL **630** Diablo 630 **855** Texas Instruments 855 **gl** Hewlett-Packard GL **kji** Kanji
-t *PrinterType*	Specifies the printer type. Printer types include 4201-3**,** ti2115**,** and so on.
-T	Specifies that the printer is attached to an ASCII terminal.

Examples

1. To make a virtual printer for the `asc` printer data stream for the 4029 printer attached locally, enter:

```
mkvirprt -A local -d mypro -n lp0 -q proq -s asc -t 4019
```

2. To make a virtual printer for a printer connected to an ENA 4033 network adapter, and to be prompted for the parameter values, enter:

```
mkvirprt -A ena
```

Files

Item	Description
/usr/sbin/mkvirprt	Contains the **mkvirprt** command.
/etc/qconfig	Contains configuration files.
/usr/lib/lpd/pio/predef/*	Contains predefined printer attribute files.
/var/spool/lpd/pio/@local/custom/*	Contains customized virtual printer attribute files.
/usr/lib/lpd/pio/etc/*.attach	Contains attachment type files.
/usr/lib/lpd/pio/etc/*.config	Contains the configuration file for the printer.
/var/spool/lpd/pio/@local/ddi*	Contains digested virtual printer attribute files.

mkwpar Command

Purpose

Creates a system workload partition (WPAR), or a WPAR specification file.

Syntax

/usr/sbin/mkwpar [**-a**] [**-A**] [**-b** *devexportsfile*] [**-c**] [**-C**] [**-E** *directory*] [**-d** *directory*] [**-B** *wparbackupdevice*] [**-D** *attribute=value* ...] ... [**-F**] [**-g** *vg*] [**-h** *hostname*] [**-H** *architecture*] [**-i**] [**-I** *attribute=value* ...] [**-k**]... [**-l**] [**-L** *attribute=value*...] [**-M** *attribute=value* ...] ... [**-N** *attribute=value* ...] ... [**-P**] [**-r**] [**-R** *attribute=value* ...] [**-S** *attribute*[+|-]=*value* ...] [**-t**] [**-T** attribute=value ...] [**-s**] [**-u** *userscript*] [**-X** *attribute=value* ...] [**-U** [*uuid*]] { **-n** *wparname* [**-p** [*name*]] [**-e** *existingwparname* **-W** | **-f** *infile*] [**-o** *outfile* [**-w**]] | **-p** [*name*] [**-n** *wparname*] [**-e** *existingwparname* **-W** | **-f** *infile*] [**-o** *outfile* [**-w**]] | **-f** *infile* [**-n** *wparname*] [**-p** [*name*]] [**-o** *outfile* [**-w**]] | **-w** **-o** *outfile* [**-n** *wparname*] [**-p** [*name*]] [**-e** *existingwparname* **-W** | **-f** *infile*] }

Restriction:

- White space must be included between a flag and its argument for *attribute=value* type flags. The **mkwpar** command is not supported on the TCB systems. Regardless of locale, only ASCII characters are allowed as arguments to **mkwpar**, **chwpar**, or **wparexec**.
- You must not run the **mkwpar** command during the AIX Live Update operation.

In addition to the previous command restrictions, more restrictions follow for the WPAR name:

- Must not be more than 25 bytes.
- Must not contain white space or any of the following symbols:

```
= : / ! ; ` ' " < > ~ & ( ) * + [ ] , . ^ 0 { } | \
```

- Must not start with hyphen (-) or 0.

Description

The **mkwpar** command builds the infrastructure to prepare a system workload partition for use. This command includes the following tasks:

- Creating the configuration data of the workload partition in the workload partition database
- Creating and populating file systems of the workload partition
- Creating an SRC subsystem for the init process of the workload partition
- Defining the resource control profile of the workload partition through Workload Manager

The following options are also available:

- Writing a specification file to simplify creation of other, similar workload partitions
- Starting the workload partitions
- Specifying whether the workload partitions must be automatically started on system start or when **/etc/rc.wpars** is started
- Specifying WPAR specific routing, by using the **-i** and **-I** flags

The **mkwpar** command supports advanced logical volume and file system options by specifying the **image.data** file as an argument to the **mkwpar -L** flag.

The **mkwpar** command supports creating a rootvg WPAR, in which the root file systems are located solely in WPAR storage devices.

Flags

Item	Description
-a	Automatically resolves conflicting static settings if required. Resolvable settings are base directory, host name, and network configuration.
-A	Specifies that the workload partition must be started each time **/etc/rc.wpars** is run, which is added to the global **/etc/inittab** to run each time that the system starts. The default is not to start the workload partition automatically. **Tip:** The workload partition is started immediately upon completion of the **mkwpar** command. To start the workload partition immediately, use the **-s** flag.
-b *devexportsfile*	Specifies an alternative file to use as the master device exports file. This file must match the format of a Device Exports File. If you do not specify a file name, **/etc/wpars/devexports** is used.
-B *wparbackupdevice*	Specifies a device that contains a workload partition backup image. This image is used to populate the workload partition file systems. The *wparBackupDevice* parameter is a workload partition image that is created with the **savewpar**, **mkcd**, or **mkdvd** command. The **-B** flag is used by the **restwpar** command as part of the process of creating a workload partition from a backup image. **Note:** The **-B** flag is mutually exclusive with the **-p** flag.
-c	Configures the workload partition to be checkpointable. This option is valid only when more checkpoints or restart software are installed and configured. When you specify this flag, any file systems that are associated with only this flag (for example, through the **-M** flag) must be remote (for example, **vfs=nfs**).
-C	Creates a versioned workload partition. This option is valid only when more versioned workload partition software is installed.
-d *directory*	Specifies a base directory for the workload partition. If you do not specify a directory name, **/wpars/<wparname>** is used.

Item	Description
-D [**devname**=*device name* | **devid**=*device identifier*] [**rootvg**=*yes* | *no*] [**devtype**=[*clone* | *pseudo* | *disk* | *adapter* | *cdrom* | *tape*]]	Configures exporting or virtualization of a global device into the workload partition every time the system starts. You can specify more than one **-D** flag to allocate multiple devices. Separate the attribute=value by blank spaces. You can specify the following attributes for the **-D** flag: **devname=device name** Specifies the device name to allocate to the workload partition. For pseudo and clone type devices, this command is the full path to the device (that is, /dev/pty10). For storage type devices, it is the logical device short name. **devid=device identifier** Specifies the unique device identifier of a disk type device to allocate to the workload partition. This attribute applies only to disk, cdrom, or tape type devices. **devtype=[clone | pseudo | disk | adapter | cdrom | tape]** Specifies the device type of the device to allocate to the workload partition. **rootvg= [yes | no]** Used to indicate whether the specified disk device is to be used as a rootvg workload partition device. If the **rootvg** attribute is not specified, the command takes the default of number.
-e *existingwparname*	Uses an existing workload partition as the source for specification data. This flag is mutually exclusive with the **-f** flag. Any values that you specify using other **mkwpar** flags override those from the existing workload partition.
-E	Specifies a directory which contains additional filesets to install when a versioned workload partition is created. If you do not specify a directory name, `/usr/sys/inst.images` is used. This option is used only during creation of a versioned WPAR.
-f *infile*	Indicates a specification file from which default values are read. This flag is mutually exclusive with the **-e** flag. Any values that you specify by using other **mkwpar** flags, override those flags from the loaded specification file.
-F	Forces the command to continue rather than fail for most error conditions.
-g *vg*	Indicates the default volume group. If you do not specify a value, rootvg is used. This volume group is used for each localfs file system whose volume group is not specified by using the *vg* parameter of the **-M** flag. The volume group for file systems that you specified in the **image.data** file, supersedes the volume group that is specified with the **-g** flag.
-h *hostname*	Specifies a host name for the workload partition. you do not specify a value, the **mkwpar** command uses the workload partition name for the host name.
-H *architecture*	Creates an architecture compatible workload partition. The valid architecture values are *{pwr4, ppc970, pwr5, pwr6, pwr7, and pwr8}*. The architecture value must be lower than, or equal to, the system hardware level. The applications in the workload partition are presented with the lowest common denominator of the specified architecture. If the workload partition is checkpointable, the workload partition must be able to migrate between systems with hardware levels greater than, or equal to, the workload partition architecture. **Note:** Values *pwr5* and *ppc970* are not compatible with each other. You cannot create a *ppc970* compatible WPAR on a POWER5 processor-based system even though the *ppc970* processor preceded the POWER 5 processor.
-i	Enables WPAR specific routing for the workload partition. A default route is not created automatically. The **-I** flag is used to specify routes, including the default route. By default, outgoing network traffic from a workload partition is routed as if it is being sent from the global environment: • Traffic between addresses that are hosted on the same global system is sent through the loopback interface. • Routing table entries that are configured in the global system, including the default route, are used to transmit workload partition traffic. If you enable WPAR specific routing by specifying the **-i** flag, the workload partition creates and uses its own routing table for outgoing traffic. Routing entries are created automatically for each of the network addresses of the workload partition to accommodate broadcast, loopback, and subnet routes. For more information about the network attributes, see the **-N** flag. You can create explicit additions to the routing table of the workload partition using the **-I** flag.
-I *attribute=value ...*	Adds routing table entries to those tables that are automatically created when WPAR specific routing is in effect. You can specify more than one **-I** flag to configure multiple routes. Using the **-I** flag automatically enables WPAR specific routing as described under the **-i** flag. You can specify the following attributes with the **-I** flag:

Item	Description	
	rtdest=*destination*	(Required) Identifies the host or network to which you are directing the route. You can specify the value by using either symbolic name or numeric address. You can use the keyword **default** to specify a default route. For more information about the route **rtdest** attribute, see the *Destination* parameter of the **route** command.
	rtgateway=*gateway*	(Required) Identifies the gateway to which packets are addressed. You can specify the value by using either symbolic name or numeric address.
	rtnetmask=*A.B.C.D*	Specifies the network mask to the destination address.
	rtprefixlen=*n*	Specifies the length of a destination prefix, which is the number of bits in the netmask. The value must be a positive integer.
	rttype={*net*|*host*}	Forces the **rtdest** attribute to be interpreted as the specified type.
	rtinterface=*if*	Specifies the interface, for example, `en0`, to associate with the route so that packets are sent by using the interface when the route is chosen.
	rtfamily =*{inet|inet6}*	Specifies the address family.
-k	Specifies the path to a user provided post installation customization script. The script is run in the global environment after the WPAR is created while WPAR file systems are mounted. The post customization script is called with the WPAR name as the first argument, and the WPAR base directory as the second argument. If the script exits with a nonzero return code, a warning is printed, but the **mkwpar** command has not failed.	
-l	Creates private and writable versions of the **/usr** and **/opt** file systems.	
-L [**image_data**= *imagedatafile*] [**shrink**= {**yes**|**no**}] [**ignore_maps**= {**yes**|**no**}]	**image_data** Specifies the path to the **image.data** file to be used for logical volume and file system options. The format of the **image.data** file is described in *Files Reference* and the **/usr/lpp/bosinst/image.template** file. File system specifications in the **image.data** file supersede file system specifications in the **Specifications** File. The **-c** flag and **-L image_data=** flags are mutually exclusive. **shrink** Specifies that the **LV_MIN_LPS** attribute, rather than the **LPs** attribute, must be used to determine the number of logical partitions for the logical volume. The **LV_MIN_LPS** attribute is from the **lv_data** stanzas from the file that the **image_data** attribute specifies. This attribute can minimize the amount of disk space that is required for a workload partition file system. This attribute has no effect if the **image_data** attribute is not specified. **ignore_maps** Specifies that the **MAPFILES** attribute must not be used to provide a disk mapping for the logical volumes that are associated with a workload partition. The **MAPFILES** attribute is from the **lv_data** stanzas from the file that the **image_data** attribute specifies. This attribute has no effect if the **image_data** attribute is not specified. **ignore_lvs** Specifies that the information from the **lv_data** stanzas is not used when the **image.data** file is being processed. The logical volumes are created with the default characteristics when the file systems are created. This attribute has no effect if the **image.dita** attribute is not specified. If this attribute is specified, the **ignore_maps** attribute is ignored.	

Item	Description
-M directory=*dir* [**vfs**=*type*] [**size**=*sizespec*] [**vg**=*volumegroup*] [**logname**=*loglv*] [**dev**=*devicepath*] [**host**=*remotehost*] [=] [**mountopts**=*mountopts*]	Specifies mount configuration attributes. Attributes must be separated by a blank space. You can specify more than one **-M** flag. By default, the **/usr** and **/opt** file systems of the workload partition are mounted over the global **/usr** and **/opt** file systems in read-only mode. The **/proc** file system of the workload partition is mounted over the global **/proc** file system in read/write mode. New logical volumes that are created in **rootvg** for **/, /var, /tmp,** and **/home**. The default settings for a specified file system can be overridden by using the **-M** flag with the **directory** attribute set to the file system name. You can specify more file systems with additional **-M** flags. The **directory** attribute denotes the directory within the workload partition where the device must be mounted. File system specifications in the **-M** flag supersede file system specifications in the **image.data** file. There are 4 basic workload partition mount forms: **localfs** Disk-based file system (**vfs**=jfs or **vfs**=jfs2) to be created at the location that is specified by the value of the **directory** within the directory structure of the workload partition. If you specify a **dev** attribute, it denotes an existing logical volume in the global environment, which is to be used to host the file system. For localfs file systems, you must specify the **size** attribute. Other optional attributes, which are of the form *attr=value*, include those attributes in the following list: **logname** Specifies the log device to use for this file system. This attribute must be specified only if the default log device that the file system uses is insufficient. • For vfs=jfs2, the default is to use an inline log. • For vfs=jfs, the default is that the file system uses an existing log device if available. Otherwise, it creates one. When the **logname** attribute is being specified, make sure that the named log device exists. **mode** Specifies the octal permission mode to assign to the base directory of this file system. The default is 755. **size** Specifies the size of the file system that is created in a format acceptable to the **crfs** command. **vg** Specifies the volume group in which the file system (if no existing logical volume device is specified by using the **dev** attribute) is created. If you do not specify a value, the volume group that is specified in the **-g** flag is used. If you do not specify the **-g** flag, **rootvg** is assumed. Specifies other options to pass to the **crfs** command when the file system is being created. Options are passed directly to the **crfs** command so the value must be in the form that is required by the **crfs** command.

Item	Description
-M directory=*dir* [**vfs**=*type*] [**size**=*sizespec*] [**vg**=*volumegroup*] [**logname**=*loglv*] [**dev**=*devicepath*] [**host**=*RemoteHost*] [=] [**mountopts**=*mountopts*] (continued)	**Restriction:** Do not specify any options to the **crfs** command that correspond to the flags in the **mkwpar** command. The **mkwpar** command must not be specified by using the attribute because incorrect results might occur: • **-a logname**=*lvname* (logname) • **-a size**=*value* (size) • **-d** *device* (dev) • **-g** *volumegroup* (vg) • **-m** *mountpoint* (directory) • **-v** *vfstype* (vfs) For more information, see **crfs** documentation for further information about the **crfs** command. **mountopts** Specifies the mount options (corresponding to the "options" attribute in an /etc/filesystems stanza). If you do not specify a mount option, by default, no mount flags are used. Option values that you can specify correspond to the **-o** options of the **mount** command. **namefs** Specifies that the global directory that is specified by the **dev** attribute is mounted over the directory that is specified by the directory attribute in the file system structure of the workload partition. The only other attribute that is applicable to a **namefs** mount is **mountopts**. For the **namefs** type, you cannot map the /, `/var`, `/opt`, `/usr`, `/tmp`, or `/proc` file system of a workload partition with write privileges to a real /, `/var`, `/opt`, `/usr`, `/tmp`, or `/proc` file system. The **namefs** mount can also be used with rootvg workload partitions. In this case, the content of the namefs mount is not saved by using the **savewpar** command. **nfs** Specifies that the directory that is specified by the **dev** attribute on the system that is exported by the **host** attribute is mounted over the workload partition **directory**. The only other attribute that is applicable to a nfs mount is **mountopts**. **Requirement:** The global system and the workload partition must both have root permissions to the NFS device. You can give the global and the WPAR root permission to the NFS device, when you export the NFS mount, by specifying the root access for the host names of both the global system and the workload partition. When an NFS device is mounted, you cannot map the **/, /var, /opt**, or **/usr** file system of a workload partition with write privileges to a real **/, /var, /opt**, or **/usr** file system. **directory** Specifies that the directory that is specified by the **directory** attribute is added to the file system structure of the workload partition. No file system is created. Use this attribute to reduce the number of file systems to manage in a workload partition, such as by eliminating the separate file systems for **/tmp** and **/var**. Ensure that the size of the containing file system is adjusted accordingly. **Note:** A directory mount cannot be used for **/usr** or **/opt**.
-n *wparname*	Specifies the name for the workload partition to be created. You must specify a name, either by using the **-n** flag, or in a specification file by using the **-f** flag, unless the **-p** name or both **-w** and **-o** flags are used.

Item	Description
-N *attribute=value*	Specifies network configuration attributes. Separate the *attribute=value* pairs by blank spaces. You can specify more than one **-N** flag to configure multiple IP addresses. You must always specify the **address** or the **address6** attribute when you use the **-N** flag. Any other values that are not specified are taken from the settings of the global system. If you do not specify the **-N** flag, the **mkwpar** command attempts to discover an appropriate IP address for the workload partition. To do that, the **mkwpar** command performs the **gethostbyname** subroutine on the workload partition host name (specified with the **-h** flag). If no **-N** flag is specified and no host name is specified, the **mkwpar** command attempts to discover the IP address by performing the **gethostbyname** subroutine on the workload partition name (specified with the **-n** flag). If you can find an address on the same subnet as any global interface, use that interface settings with the resolved IP address to create the default network entry. You can specify the following attributes for the **-N** flag: • **interface**= *if* or **interface**=*namemappedif* • **address**=*A.B.C.D* • **netmask**=*A.B.C.D* • **broadcast**=*A.B.C.D* • **address6**=*S*:*T*:*U*:*V*:*W*:*X*:*Y*:*Z* • **prefixlen**=*n* The name-mapped interface is defined in the **/etc/wpars/devmap** file. You can specify the mapping between the name-mapped interface and the system interface as follows: `# The comments start with '#'` `# Each line contains a pair of name-mapped interface` `# and real interface separated by tab or blank spaces.` `foo en0` `goo en1` `soo en2` To define an IPv6 network configuration, specify the **-N** flag with the **address6**, **prefixlen**, and **interface** attributes. The **address6** attribute is a 128-bit address. The address is represented by eight 16-bit integers that are separated by colons. Each integer is represented by 4 hex digits. Leading zeros can be skipped, and consecutive null 16-bit integers can be replaced by two colons (one time per address). The **prefixlen** attribute is the number of high-order bits that are used to mask the IPv6 address and to comprise the prefix. The value of the **prefixlen** attribute ranges from 0 through 128. Each **-N** flag can accept either IPv4 attributes, or IPv6 attributes, but not both.
-o *outfile*	Indicates an output path and file name to which to write specification data. This specification file can then be used to create a workload partition later, by using the **-f** flag.
-O	This flag is used to force an existing volume group to be overwritten on a particular set of devices, specified with the **-D rootvg=yes** flag directive. If not specified, the overwrite value defaults to FALSE. This flag must be specified only once, as its setting is applied to all devices specified with the **-D rootvg=yes** flag directive.
-p [*name*]	Indicates that the workload partition's file systems exist and must be preserved, which means the root part must not be populated. You must specify the existing file systems to the **mkwpar** command in one of the following two ways: • Use the *name* parameter to specify an existing mount group in **/etc/filesystems**. Such a mount group usually exists because a previous workload partition was removed by using **rmwpar -p**. If you specify the *name* parameter, it cannot match the name of an existing workload partition on the system. If you specify the **-d** flag, the mount points of the file systems are adjusted accordingly. If you do not specify the **-d** flag, the base directory of the workload partition is determined based on the mount points that are associated with the discovered file systems. • If the file systems are not defined in **/etc/filesystems**, use the **-p** flag with the **-M** flag or mount stanzas in the specification file to define the attributes of the file systems. If you specify the **-p** flag with the *name* parameter, and no workload partition name is provided by using other means (for example, the **-n** flag or **general.name** in the specification file), you can also use the *name* parameter as the workload partition name. **Note:** The **-p** flag is mutually exclusive with the **-B** flag. The **-p** flag can be used to re-create a versioned workload partition. The workload partition's file systems must be preserved. To re-create such a versioned workload partition, the **-f infile** flag is included to provide a specification file that is saved from the original workload partition. The specification file is required to preserve the **ostype** value of the workload partition.
-P	Sets the root password for the workload partition. The **mkwpar** command prompts you for the password interactively.

Item	Description
-r	Duplicates the network name resolution configuration from the global system. The following files, if they exist, are copied into the workload partition: • **/etc/resolv.conf** • **/etc/hosts** • **/etc/netsvc.conf** • **/etc/irs.conf** • **/etc/networks** If the NSORDER environment variable is defined in the calling environment, it is added to the workload partition's `/etc/environment` file.
-R *attribute=value*	Allows specification of resource control attributes. You specify only one **-R** flag. Most resource controls are similar to those resource control attributes that are supported by Workload Manager. You can use the following attributes: **active={yes|no}** • **Active=no** means that the resource attributes are defined but the resource controls are not activated when the WPAR is started. • **Active=yes** means that the resource control attributes are activated when the WPAR starts. **Tip:** If this field is set to 'no', performance metrics such as processor and memory usage are not available by using such commands as **topas** and **wlmstat**, either inside and outside of the workload partition. **rset**=*rset* Configures the workload partition to use a resource set that was created by the **mkrset** command. **shares_CPU**=*n* Specifies the number of processor shares that are available to the workload partition. **CPU**=*m*%-*SM*%,*HM*% Specifies the percentage processor limits for the processes of the workload partition. **shares_memory**=*n* Specifies the number of memory shares that are available to the workload partition. **memory**=*m*%-*SM*%,*HM*% Specifies the percentage memory limits for the processes of the workload partition. **procVirtMem**=*n*[M|MB|G|GB|T|TB] Specifies the maximum amount of virtual memory that a single process can consume. Processes that exceed the specified limit are terminated. The valid units are megabytes (M or MB), gigabytes (G or GB), and terabytes (T or TB). The minimum limit that is allowed is 1 MB. The maximum limit that can be specified is 8796093022207M, 8589934591G, or 8388607T. If you set the value to -1 (no units), the limit is disabled. See Workload Manager limits File. **totalVirtMem**=*n*[M|MB|G|GB|T|TB] The maximum amount of virtual memory that can be consumed by the WPAR as a whole. Processes that cause the specified limit to be exceeded are terminated. The valid range and units are the same as **procVirtMem**. If you set the value to -1 (no units), the limit is disabled. See Workload Manager limits File. **totalProcesses**=*n* Specifies the total number of processes that are allowed in the workload partition. See Workload Manager limits File. **totalPTYs**=*n* Specifies the total number of pseudo terminals that are allowed in the workload partition. See pty Special File. **totalLargePages**=*n* Specifies the number of large pages that can be allowed for the workload partition. See Large Pages. **pct_msgIDs**=*n*% Specifies the percentage of the maximum number of message queue IDs of the system that are allowed in the workload partition. See Message Queue Kernel Services. **pct_semIDs**=*n*%

Item	Description
	Specifies the percentage of the maximum number of semaphore IDs of the system that are allowed in the workload partition. **pct_shmIDs**=*n*% Specifies the percentage of the maximum number of shared memory IDs of the system that are allowed in the workload partition. See Shared Memory. **pct_pinMem**=*n*% Specifies the percentage of the maximum pinned memory of the system that can be allocated to the workload partition. See Support for pinned memory. **totalThreads**=*n* Specifies the total number of threads that are allowed in the workload partition. See Workload Manager limits File.
rootvg=*yes*|*no*	Used to indicate whether the specified disk device is to be used as a rootvg WPAR device. If the **rootvg** option is not specified, the command takes the default of *no*.
-s	Starts the workload partition after it is created.
-S secfile = */path/to/secattrs* **privs[+|-]** = *list*	Configures the set of privileges that can be assigned to processes that are running in a system workload partition. You can provide privileges in a specification file (see the **-f** flag), in a separate security attributes file through **-S secfile=***/path/to/secattrs*, or on the command line by using the **-S privs=***list* flag. If you do not provide security attributes through one of these mechanisms, the **/etc/wpars/secattrs** file is used by default. When you use a separate security attributes file (either the default file or the file that is supplied through **-S secfile**), this file is read once when the workload partition is created to determine the privileges that are associated with the workload partition. Subsequent changes to the file have no effect on existing workload partitions. The default security attributes file */etc/wpars/secattrs* must not be modified directly as it might be overwritten in the future. If you use a base list of privileges from a specification file or security attributes file (including the default), individual privileges can be added to or removed from the list by specifying **-S privs+=***list*, **-S privs-=***list*, or both. Separate attributes must be separated by a blank space and must be unique, which means **secfile=, privs=, privs+=**, and **privs-=** cannot be specified more than once. Privileges must be comma-separated (without spaces) and must be unique. Attributes are processed in the following order regardless of the order that is specified in either the command line or the specification file: 1. The first attribute to be processed is the **privs** attribute without the **+** or **-** modifier. For example, **privs=PV_AZ_READ,PV_AZ_ADMIN**. If this attribute is found, no other attributes can be used. 2. The next attribute to be processed is the **secfile** attribute. See the security stanza of the Specification File Format for details on the format of this file. 3. If none of the attributes that are listed previously are specified, the **/etc/wpars/secattrs** file is used to populate the list of privileges. 4. The next attribute to be processed is an attribute with the **+** modifier. For example, **privs+=PV_DAC_UID,PV_AZ_ROOT**. This command adds the specified privileges to the list of privileges that are specified in the security file. 5. The final attribute to be processed is an attribute with the **-** modifier. For example, **privs-=PV_AZ_ROOT**. This command removes the specified privileges from the list of privileges that are specified in the security file. **Tip:** If you specify the **-S** flag on the command line, any security attributes in the specification file are ignored.

Item	Description
-u *userscript*	Specifies the path to a user script to be run by workload partition commands at various administration points. The parameter of the **-u** flag can be a string that is enclosed in quotation marks, including more arguments to be passed to the script. The first component of the parameter of the **-u** flag must be an absolute path to an existing executable file. The script is started in the following manner: `/path/to/userScript <action> <wparName>` The first argument indicates the administrative action that is being performed, as follows: **WPAR_LOAD** A script runs in the global environment after the kernel is configured, and before the tracked process is created. If the script returns a value other than zero, the workload partition cannot be started. **WPAR_START** A script runs in the global environment as soon as the workload partition becomes active. For system workload partitions, the script runs after the device configuration is complete. For application workload partitions, the script runs as soon as the tracked process is started. In the latter case, this code path can be run asynchronously by a dissociated process with its standard I/O streams closed or redirected. Internal messaging must be handled accordingly, and the script must account for the fact that short-lived workload partitions might be stopped or stopping at any point during the execution of the script. If the script returns a value other than zero, a warning is logged, but no other behavior changes. **WPAR_STOP** A script runs in the global environment after all workload partition processes finish before the kernel is unconfigured. **Note:** This code path can be started by a dissociated process with its standard I/O streams closed or redirected to SRC logs. If the script returns a value other than zero, a warning is logged, but no other behavior changes The second argument is the name of the workload partition. The script can use the **lswpar** command to obtain any other necessary configuration data.
-U [*Workload Partition UUID*]	Specifies the Workload Partition UUID. If you do not specify the value, the UUID is automatically generated for the corresponding Workload Partition.
-w	Writes the specification file only. Used with the **-o** flag, the **-w** flag causes the **mkwpar** command to quit after the new specification file is written, without actually creating the workload partition.
-W	Filters the bootset related information from the WPAR specification file. When the **-W** flag is used with the **-e** flag to generate a specification file from an existing WPAR, the resulting specification file does not include the bootset related attributes, such as bootset and bootlist.
-X [**exportfile**=*/path/to/file* |[**kext**=[*/path/to/extension*|*ALL*]] [**local**=*yes* | *no*] [**major**=*yes* | *no*]	Configures exporting kernel extensions that will be allowed to load inside a workload partition. You can specify more than one **-X** flag to allocate multiple kernel extensions. Separate the **attribute**=*value* by blank spaces. This flag is not valid for application workload partitions. You can specify the following attributes for the **-X** flag: **exportfile=*/path/to/file*** Specify a file containing valid extension stanzae that will be exported. An extension stanza should contain at least the **kext** attribute. The **local** and **major** attribute can also be specified in the stanza which are described below. The **exportfile** attribute is mutually exclusive with the **kext** attribute. It is also mutually exclusive with the local and major attribute because these can be specified for each extension stanza in the **exportfile**. This is a file that can be created by a user to use with **exportfile**=*/path/to/file* for **mkwpar** and **chwpar**. It can contain multiple extension stanzas. The **kext** attribute is required for each extension stanza. The local and major are optional as they both have default value **no**. The **exportfile** will look similar to the following. extension: `major = "yes"` `local = "no"` `kext = "/usr/lib/drivers/ldterm"` **kext=*/path/to/extension*** Specify a kernel extension that will be exported. This is a kernel extension located in the global system's filesystem. The keyword **ALL** can also be specified. This will allow a workload partition to load any extension. When **ALL** is specified, the **local** and **major** attributes are restricted to **local**=*yes* and **major**=*no*. Additional **-X** flags can be specified to overide the restricted local and major values. The **kext** attribute is mutually exclusive with the **-X exportfile** attribute. **local=*yes*/*no*** Specifying **local**=*yes* will make an instance of the kernel extension accessible to only the workload partition that is loading it. Specifying **local**=*no* will share the instance of the kernel extension loaded in the global system. By default, **local**=*no*. **major=*yes*/*no*** This attribute should only be used for kernel extensions that have an associated device major. By default, **major**=*no*.

Item	Description
-t	Copies the file systems from the rootvg volume groups from a system backup image specified by the **-B** flag or the global system.
-T [**preserve_private** ={**yes**|**no**}] [**preserve_wpars** ={**yes**|**no**}]	Controls behavior when copying the file systems from a rootvg volume group or system backup.
preserve_private ={**yes**|**no**}	Controls whether filesets that are designated as not visible within WPARs remain in a WPAR that is created by copying a rootvg volume group from a system backup or global system. The default value is no.
preserve_wpars ={**yes**|**no**}	Indicates whether the file systems associated with WPARs are available in the source system in a WPAR, which is created by copying a rootvg volume group from a global system. The default value is no.

Security

Access Control: Only the root user can run this command.

Examples

1. To create a workload partition called `roy`, enter the following command:

   ```
   mkwpar -n roy -N address=192.168.0.51
   ```

   All values that are not specified are generated or discovered from the global system settings.
2. To create a workload partition based on an existing specification file, enter the following command:

   ```
   mkwpar -f /tmp/wpar1.spec
   ```
3. To create a modified copy of a specification file with a new IP address, host name, and workload partition name (without creating a workload partition), enter the following command:

   ```
   mkwpar -f /tmp/wpar1.spec -N address=219.168.45.132 -h www.flowers.com -n wpar2
   -o /tmp/wpar2.spec -w
   ```
4. To create a specification file, which is based on an existing workload partition, enter the following command:

   ```
   mkwpar -e wpar1 -o /tmp/wpar2.spec -w
   ```
5. To recreate a workload partition that was previously removed with the **rmwpar -p** command, enter the following command:

   ```
   mkwpar -p wparname
   ```
6. To create a rootvg workload partition, enter the following command:

   ```
   mkwpar -n test -D devname=hdisk1 rootvg=yes -O
   ```
7. To create a rootvg workload partition called wpar1 with the storage device on an adapter, enter the following command (assuming that hdisk3 is attached to the adapter, fcs2):

   ```
   mkwpar -n wpar1 -D devname=fcs2 -D devname=hdisk3 rootvg=yes
   ```
8. To create a specification file from an existing workload partition, without including bootset related information, enter the following command:

   ```
   mkwpar -e <existing wparname> -W -w -o <path to spec file>
   ```
9. To create a WPAR with a default route, enter the following command:

   ```
   mkwpar -n wparB -N address=192.162.1.2 interface=en0 netmask=255.255.255.0 -i
   -I rtdest=0.0.0.0 rtgateway=192.162.1.1
   ```

10. To create a WPAR with its own routing table but no default route, enter the following command:

```
mkwpar -n wparA -N address=192.152.1.2 interface=en0 netmask=255.255.255.0 -i
```

Files

Item	Description
/etc/wpars/devexports	Default device export control file for workload partitions.
/etc/wpars/secattrs	Default security file for workload partitions.
/usr/samples/wpars/sample.spec	An annotated workload partition specification file.

mkwpardata Command

Purpose

Creates a file containing information about a workload partition for use by the **savewpar** and **restwpar** commands.

Syntax

mkwpardata [**-X**] [**-m**] *WparName*

Description

The **mkwpardata** command creates a file containing information about a workload partition (WPAR) for use by the **savewpar** and **restwpar** commands. The information includes the list of logical volumes, file systems and their sizes, the list of volume groups, and the WPAR name. The following files are created:You can edit the information in the file before issuing the **savewpar** command.

Flags

Item	Description
-m	Creates map files that specify the mapping of the logical-to-physical partitions for each logical volume in the WPAR. This mapping can be used to allocate the same logical-to-physical mapping when the image is restored. The map file locations are stored in the MAPFILE field in the **image.data** file for each logical volume. For example, for the **hd7** logical volume, the location of the map file is **/tmp/wpardata/***WparName***/hd7.map**. The MAPFILE field in the **image.data** file for the **hd7** logical volume is under the entry **MAPFILE=/tmp/wpardata/***WparName***/hd7.map**. The map files in the backup image are copied after the **image.data** file.
-X	Expands the **/tmp** file system if needed.

Parameters

Item	Description
WparName	Specifies the name of workload partition to back up.

Files

Item	Description
/tmp/wpardata/*WparName***/image.data**	Created for general and storage information about a WPAR. The *WparName* variable reflects the name of the WPAR. The **savewpar** command uses this file to create a backup image that can be used by the **restwpar** command to reinstall the WPAR. The **mkwpardata** command overwrites this file if it already exists. The **image.data** file is located in the **/tmp/wpardata/***WparName* directory, where *WparName* is the workload partition name.

mm Command

Purpose

Prints documents formatted with memorandum macros.

Syntax

mm [**-M** *Media*] [**-c**] [**-e**] [**-E**] [**-t**] [**-12**] [**-T***Name*] { *File* ... | **-** }

Description

The **mm** command formats documents that use the **nroff** command and the **mm** macro package. The **mm** command has flags that specify preprocessing by the **tbl** and **neqn** commands and postprocessing by various terminal-oriented output filters. The proper pipelines and the required flags for the **nroff** command are generated depending on the flags that are selected.

Notes:

1. Use the **-o***List* flag of the **nroff** command to specify ranges of output pages. Remember that if the **mm** command is called with the **-e**, **-t**, or **-** (minus sign) flags together with the **-o***List* flag, and if the last page of the document is not specified by the *List* variable, you may receive a `broken pipe` message. This message is not an indication of any problem and can be ignored.
2. The **mm** command calls the **nroff** command with the **-h** flag. With this flag, the **nroff** command assumes that the workstation has tabs set every 8 character positions.
3. If you use the **-s** flag of the **nroff** command (to stop between pages of output), use a linefeed (rather than the Enter key or a newline character) to restart the output. The **-s** flag of the **nroff** command does not work with the **-c** flag of the **mm** command or if the **mm** command automatically calls the **col** command**.**
4. Providing inaccurate information to the **mm** command about the kind of workstation its output is to be printed on will produce unsatisfactory results. However, if you are redirecting output to a file, use the **-T37** flag. Then, use the appropriate workstation filter when you print the file.

To obtain a list of **mm** command flags, enter the command name with no parameters. The flags can occur in any order, but they must come before the *File* parameter. Any other flags (for instance, **-r***ANumber*) are passed to the **nroff** command.

Flags

Item	Description
-M *Media*	Specifies a paper size in order to determine the amount of imageable area on the paper. Valid values for the *Media* variable are: **A4** Specifies a paper size of 8.27 X 11.69 inches (210 X 297 mm). **B5** Specifies a paper size of 6.93 X 9.84 inches (176 X 250 mm). **EXEC** Specifies a paper size of 7.25 X 10.5 inches (184.2 X 266.7 mm). **LEGAL** Specifies a paper size of 8.5 X 14 inches (215.9 X 355.6 mm). **LETTER** Specifies a paper size of 8.5 X 11 inches (215.9 X 279.4 mm). This is the default value. **Note:** The *Media* variable is not case sensitive.
-c	Calls the **col** command. Note that the **col** command is called automatically by the **mm** command for the following terminal names. The following devices can be specified by the **-T***Name* flag, the **$TERM** shell variable, or by using the default: • **ppds** • **lp** • **2631** • **8510**
-e	Calls the **neqn** command; also causes the **neqn** command to read the **/usr/share/lib/pub/eqnchar** file. See the **eqnchar** file format.
-E	Calls the **-e** flag of the **nroff** command.
-t	Calls the **tbl** command.
-12	Uses 12-pitch font. Use this when the **$TERM** shell variable is set to 300, 300s, 450, or 1620. (The pitch switch on the DASI 300 and 300s workstations must be manually set to 12 if this flag is used.)
-T*Name*	Uses the workstation type specified by the *Name* variable. By default, the **mm** command uses the value of the **$TERM** shell variable from the environment as the value of the *Name* variable. If the **$TERM** shell variable is not set, the **mm** command uses **lp** (the generic name for printers that can underline and tab). If several workstation types are specified, the last one listed applies.
-	Forces input to be read from standard input.

Parameters

Item	Description
File	Specifies the file that the **mm** command formats.

Examples

1. When the **$TERM** shell variable is set in the environment to the **hplj** command, the following two command lines are equivalent:

```
mm -t -rC3 File
tbl File | nroff -mm -Thplj -h -rC3
```

2. The **mm** command reads the standard input when you specify a **-** (minus sign) flag instead of a value for the *File* variable. This option allows you to use the **mm** command as a filter, as follows:

```
cat File | mm -
```

Note: Using other files together with a **-** (minus sign) flag leads to undesired results.

Environment Variables

Item	Description
$TERM	Specifies the terminal names.

Files

Item	Description
/usr/share/lib/pub/eqnchar	Contains special character definitions for the **eqn** command and the **neqn** command.

mmt Command

Purpose

Typesets documents.

Syntax

mmt [**-M** *Media*] [**-a**] [**-c**] [**-e**] [**-g**] [**-p**] [**-t**] [**-z**] [**-T***Name* | **-D***Destination*] [*File* | **-**]

Description

Similar to the **mm** command, the **mmt** command typesets its input using the **troff** command, rather than formatting it with the **nroff** command. The **mmt** command uses the **mm** macro package. There are flags to specify preprocessing by the **tbl**, **pic**, **eqn**, and **grap** commands. The proper pipelines, required parameters, and flags for the **troff** command and the **mm** macro package are generated, depending on the flags selected.

There are several flags that are specific to the **mmt** command. Any other parameters or flags (for instance, **-r***ANumber* or **-a**) that you give the **mmt** command are passed to the **troff** command. You can put flags in any order, but they must be listed before any input files. *File* specifies the file that the **mmt** command formats. If you do not give *File* parameters or other flag variables, the **mmt** command prints a list of its flags.

The **mmt** command, unlike the **troff** command, automatically pipes its output to a postprocessor, unless specifically requested not to do so. The user should not specify a postprocessor when using the **mmt** command. The precedence is as follows:

1. The **-z** flag; no postprocessor is used.
2. The **-T***Name* flag.
3. The **TYPESETTER** environment variable is read.
4. The default is set to **ibm3816**.

The **mmt** command reads standard input when you specify a **-** (minus sign) instead of any *File* parameters.

Use the **-o***List* flag of the **troff** command to specify ranges of pages to be output.

> **Note:** If you call the **mmt** command with one or more of the **-e**, **-c**, **-t**, **-p**, **-g**, and - (minus sign) flags together with the **-o***List* flag of the **troff** command, you may receive a `broken pipe` message if the last page of the document is not specified by the *List* variable. This broken pipe message is not an indication of any problem and can be ignored.

Flags

Item	Description
-M *Media*	Specifies a paper size in order to determine the amount of imageable area on the paper. Valid values for the *Media* variable are: **A4** Specifies a paper size of 8.27 X 11.69 inches (210 X 297 mm). **A5** Specifies a paper size of 5.83 X8.27 inches (148 X 210 mm). **B5** Specifies a paper size of 6.93 X 9.84 inches (176 X 250 mm). **EXEC** Specifies a paper size of 7.25 X 10.5 inches (184.2 X 266.7 mm). **LEGAL** Specifies a paper size of 8.5 X 14 inches (215.9 X 355.6 mm). **LETTER** Specifies a paper size of 8.5 X 11 inches (215.9 X 279.4 mm). This is the default value. **Note:** The *Media* variable is not case sensitive.
-a	Displays readable **troff** output to the terminal.
-c	Preprocesses the input files with the **cw** command.
-e	Calls the **eqn** command; also causes the **eqn** command to read the **/usr/share/lib/pub/eqnchar** file (see the **eqnchar** file format).
-g	Calls the **grap** command, which in turn calls the **pic** command.
-p	Calls the **pic** command.
-t	Calls the **tbl** command.
-z	Starts no output filter to process or redirect the output of the **troff** command.
-D*Destination*	Directs the output to a device specified by the *Destination* variable. Supported destination devices for English-language output is 4014, which is the Tektronix 4014 terminal by way of the **tc** command.

Item	Description
-T*Name*	Creates output for a **troff** device as specified by the *Name* variable. The output is sent through the appropriate postprocessor.. The default value is **ibm3816**. Possible Name variables are: **ibm3812** 3812 Pageprinter II. **ibm3816** 3816 Pageprinter. **hplj** Hewlett-Packard LaserJet II. **ibm5587G** 5587-G01 Kanji Printer multi-byte language support. **psc** PostScript printer. **X100** AIXwindows display.
-	Forces input to be read from standard input.

mmtu Command

Purpose

Displaying, adding, and deleting maximum transfer unit (MTU) values used for path MTU discovery.

Syntax

mmtu { **-a** *Value* | **-d** *Value* | **-s** }

Description

Use the **mmtu** command to display, add, and delete maximum transfer unit (MTU) values to the list of potential path MTU values. Path MTU discovery uses the list of potential path MTU values to detect the path MTU. The list of potential path MTU values is only used when there are routers in the path that do not comply with RFC 1191. The user must have administrative authority to add or delete MTU values.

Note: The **-a** and **-d** flags used to modify the list of potential path MTU values are disallowed when executed within workload partitions.

Flags

Item	Description
-a *Value*	Adds the new MTU to the list of potential path MTU values.
-d *Value*	Deletes the value from the list of potential path MTU values.
-s	Displays the current list of potential path MTU values.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To add a value to the list of potential path MTU values, enter:

```
mmtu -a mtu-value
```

2. To delete a value from the list of potential path MTU values, enter:

```
mmtu -d mtu-value
```

3. To display the contents of the list of potential path MTU values, enter:

```
mmtu -s
```

Files

Item	Description
/usr/sbin/mmtu	Contains the **mmtu** command.

mobip6ctrl Command

Purpose

Configures and manages the mobile IPv6 home agent and correspondent node functionality.

Syntax

mobip6ctrl [**-c**] [**-R**] [**-b**] [**-S** { **0** | **1** }] [**-n** { **0** | **1** }] [**-l** *LifeTime*] [**-a** | **-d** *HomeAddress CareOfAddress MyAddress*]

Description

The **mobip6ctrl** command is used to configure and manage the mobile IPv6 home agent and correspondent node. It can enable and disable NDP proxy and IP security checking, and it can be used to display or modify the mobile IPv6 binding cache.

NDP proxy must be enabled if the system is configured as a home agent. This allows the home agent to intercept packets addressed to mobile nodes that are not currently on their home network.

IP security checking enables checking to ensure that IP security is used for the Binding Update and Binding Acknowledgement messages sent for mobile IPv6. Because these two types of messages have the ability to affect the routing of packets addressed to a mobile node, they would represent a significant security vulnerability if not protected by IP security. If checking is enabled, the mobile IPv6 home agent or correspondent node will discard any Binding Update or Binding Acknowledgement packets that are not protected by IP security.

The mobile IPv6 binding cache on a home agent or correspondent node maps home addresses to the current care-of addresses for each mobile node. This allows the home agent to tunnel traffic to the mobile node at its current location, and allows a correspondent node to send packets directly to a mobile node at its current location. The **mobip6ctrl** command can be used to view the binding cache or manually edit it for debugging purposes.

Normally, this command is used from the **/etc/rc.mobip6** script when mobile IPv6 has been configured using system management.

Flags

Item	Description
-a *HomeAddress CareOfAddress MyAddress*	Adds this entry to the binding cache.
-b	Displays all binding cache entries.
-c	Compatibility option which enables the support of the mobiles implementing the draft #13 of the *Mobility support in IPv6* specification. Using this option, the home agent or correspondent node will accept the binding update messages sent using a Destination Option and using an Authentication Header (AH) to protect these packets with IPsec.
-d*HomeAddress CareOfAddress MyAddress*	Delete this entry from the binding cache.
-l *LifeTime*	Specifies the default life time value for binding cache entries in seconds.
-n 0 | **1**	Activates or deactivates NDP proxy capabilities. A value of 1 activates the NDP proxy capabilities, and a value of 0 disables NDP proxy capabilities. The default value is 0.
-R	Resets all the binding cache entries.
-S 0 | **1**	Enables or disables checking to ensure that IP security is used for all Binding Update and Binding Acknowledgement packets. A value of 1 enables checking, and a value of 0 disables checking. The default value is 0.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Security

You must be the root user or a member of the system group to execute this command.

Examples

1. The following example enables NDP proxy and IP security checking for mobile IPv6:

```
mobipctrl -S 1 -n 1
```

2. The following example displays all entries in the binding cache:

```
mobip6ctrl -b
```

The output from this command looks similar to the following:

```
                BINDING CACHE LIST (1 elem)

Home Address.........: 3ffe:300:20:1102::217
Care-Of Address......: 3ffe:300:20:1101::217
My Address...........: 3ffe:300:20:1102::223
Life time............: 518
Time since last usage: 50
Rate limit time......: 0
```

```
Retransmit count.....: 0
Sequence number......: 14
Registered by me.....: 1
Prefix length........: 64
```

mobip6reqd Daemon

Purpose

Provides the Mobile IPv6 home agent daemon.

Syntax

To run the daemon using the System Resource Controller:

startsrc -s mobip6reqd

To run the daemon without using the System Resource Controller:

mobip6reqd

Description

The **mobip6reqd** daemon must be running in order for the system to function as a mobile IPv6 home agent. This daemon enables the home agent to perform NDP proxying for mobile nodes. The daemon is normally started automatically by the **/etc/rc.mobip6** script if the mobile IPv6 home agent has been enabled using system management.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Security

You must be the root user or a member of the system group to execute this command.

Examples

1. The following example starts the **mobip6reqd** daemon:

```
startsrc -s mobip6reqd
```

2. The following example stops the **mobip6reqd** daemon:

```
stopsrc -s mobip6reqd
```

monacct Command

Purpose

Performs monthly or periodic accounting.

Syntax

/usr/sbin/acct/monacct [**-X**] [*Number*]

Description

The **monacct** command performs monthly or periodic accounting. The intervals are set in the **crontab** file. You can set the **cron** daemon to run the **monacct** command once each month or at some other specified time period. The **monacct** example shows how to set up this command for use with the **cron** daemon. See the **crontab** command for more information about setting up **cron** files.

The *Number* parameter indicates which month or other accounting period to process. The default value of the *Number* parameter is the current month. The **monacct** command creates summary files in the **/var/adm/acct/fiscal** file and restarts summary files in the **/var/adm/acct/sum** file, the cumulative summary to which daily reports are appended.

> **Note:** You should not share accounting files among nodes in a distributed environment. Each node should have its own copy of the various accounting files.

Flags

Item	Description
-X	Processes all available characters for each user name instead of truncating to the first 8 characters. The **-X** flag will also cause the **monacct** command to use the **/var/adm/acct/sumx** and **/var/adm/acct/fiscalx** directories instead of the **/var/adm/acct/sum** and **/var/adm/acct/fiscal** directories.

Security

Access Control: This command should grant execute (x) access only to members of the administrative group.

Example

To produce automatically a monthly accounting report, add the following to the **/var/spool/cron/crontabs/root** file:

```
15 5 1 * * /usr/sbin/acct/monacct
```

This example shows the instructions that the **cron** daemon will read and act upon. The **monacct** command will run at 5:15 (15 5) the first day of each month (1). This command is only one of the accounting instructions normally given to the **cron** daemon.

Files

Item	Description
/usr/sbin/acct	Contains the accounting commands.
/var/adm/acct/fiscal	Contains accounting data files.
/var/adm/acct/sum	Cumulative daily accounting records.
/var/spool/cron/crontabs	Contains the commands to be run by the **cron** daemon at regularly scheduled intervals.

mon-cxma Command

Purpose

Monitor status of 128-port asynchronous subsystem and attached devices.

Syntax

To Display All 128-Port Adapters:

mon-cxma

To Display Syntax or Slots and Bus Information:

mon-cxma { **-h** | **-x** }

To Display Specific Slots and Bus Information:

mon-cxma { [**-l** [*LogFile*] [**-f** [*DeviceFile*]] [**-s** [*SlotNumber*]] [**-b** [*BusNumber*]] }

Description

The **mon-cxma** command is a software tool which provides a means to monitor the status of serial devices and remote async nodes (RAN) attached to the IBM 128-port asynchronous adapter. It is used for subsystem problem determination and can be accessed locally and remotely via modem. The only restriction on modem access is that the modem can not be physically attached to the 128-port adapter being monitored.

When the user enters the **mon-cxma** command at the command line, it automatically detects and displays all available 128-port adapters in the system. The bus and slot location within the system is displayed for each adapter and the user can select adapter to monitor.

You can use the System Management Interface Tool (SMIT) **smit 128psync** fast path to advance directly to the "128-Port Asynchronous Adapter" menu. When run from SMIT, the **mon-cxma** command automatically displays all available 128-port adapters in the system.

Flags

Item	Description
-b [*BusNumber*]	Specifies the bus number of the device. Valid values for *BusNumber* are 0 to (n-1), where n is the number of buses the system has.
-f [*DeviceFile*]	Specifies the device special file. Use this file to look at a specific device driver without having to make a selection. The default device special file is **/dev/cxma0**.
-h	Shows syntax information.
-l [*LogFile*]	(Lowercase L) Specifies the file to be used as the log. Use this file to store information from the screen when the IMAGE key is pressed. The default log file is **/tmp/mon-cxma.log**.
-s [*SlotNumber*]	Specifies the slot number of the device. Valid values for *SlotNumber* are 0 to (n-1), where n is the number of slots the system has.
-x	Shows the POS (Programmable Select Option) register values for all the slots and buses.

Note: -x and **-h** ignore other options.

Security

Access Control: Root authority required to run this command.

Auditing Events: N/A

Examples

1. To run the mon-cxma command using the SMITfastpath, enter:

```
smit 128psync
```

2. To display all 128-port adapters, enter:

```
/usr/lbin/tty/mon-cxma
```

Files

Item	Description
/usr/lbin/tty/mon-cxma	Contains the **mon-cxma** command.
/tmp/mon-cxma.log	Contains the log file.

monitord Daemon

Purpose

Communicates with the License Use Management server and requests a concurrent-use license for each countable login.

Syntax

monitord [**-t** *Minutes*] [**-v** *Version*.*Release*]

Description

The operating system has multiple ways to access the system, and each of them has a different behavior upon exit. The **monitord** daemon provides a common interface to the License Use Management **netlsd**. **monitord** communicates with the License Use Management server and requests a concurrent-use license for each countable login.

> **Note:** The License Use Management licensing mechanism is used only if the system has the *floating license mode* enabled.

After user logout, **monitord** requests **netlsd** to release the specific license the user was using, in order to make it available for further logins.

monitord is started when the **chlicense -f on** command is used to enable the *floating license mode*. When the *floating license mode* is enabled, **monitord** is started upon system startup via an entry in **/etc/inittab**. The default (invoked without **-t** option) is an interval of fifteen minutes.

The entry in **/etc/inittab** looks like the following:

```
monitord:2:once:/usr/sbin/monitord >/dev/console 2>&1
```

Flags

Item	Description
-t *Minutes*	Sets the value in minutes of the heartbeat interval. A value of 0 sets an infinite interval. The default is fifteen minutes.
-v *Version*.*Release*	Enables the *floating license mode* for a license of the specified *Version* and *Release*.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

moo Command

Purpose

Starts the number-guessing game.

Syntax

moo

Description

The **moo** command picks a combination of four random, non-repeating numbers. You guess four numbers at the `your guess?` prompt. Each correct number in an incorrect position in the four number combination scores "cow." Each correct number in the correct position in the four number combination scores a "bull." For example:

```
your guess?
1470
bulls = 0  cows = 1
your guess?
```

In this example, one of the four numbers (1, 4, 7, and 0) is correct but in the incorrect position. None of the numbers are correct and in the correct position.

To quit the game, press the Interrupt (Ctrl-C) or End Of File (Ctrl-D) key sequence.

File

Item	Description
/usr/games	Contains the system's games.

more Command

Purpose

Displays file contents one screen at a time.

Syntax

more [**-c**] [**-d**] [**-e**] [**-H**] [**-i**] [**-l**] [**-N**] [**-s**] [**-u**] [**-v**] [**-z**] [**-n** *Number*] [**-p** *Subcommand*] [**-t** *Tagstring*] [**-W** *Option*] [**-x** *Tabs*] [*File* ...]

Description

The **more** command reads files and displays the text one screen at a time. The command pauses after each screen and prints the word `More` at the bottom of the screen. If you then press a carriage return,

the **more** command displays an additional line. If you press the space bar, the **more** command displays another full screen of text.

Note: On some terminal models, the **more** command clears the screen, instead of scrolling.

Instead of naming files to read, you can either redirect or pipe standard output, such as a long directory listing, to the **more** command. The command adds a % (percent sign) to its prompt when reading from a file rather than a pipe. This provides the percentage of the file (in characters, not lines) that the **more** command has read.

The **more** command sets the terminal to NOECHO mode so the output can be continuous. With the exception of the / and ! subcommands, commands that are typed do not normally show up on the terminal. If the standard output is not a terminal, the **more** command will act just like the **cat** command, except that a header will be printed before each file in a series.

Environment Variables

Environment variables affect the way the **more** command works. You can set some environment characteristics in the **/etc/environment** file and system profile files, such as the **.ksh**, **.csh**, and **.profile** files.

The **more** command uses the **TERM** variable to determine terminal characteristics. If this variable is NULL or not set, the command uses the default terminal type. The **/usr/share/lib/terminfo** directory contains definitions for terminal characteristics.

By default, the **more** command window size is 2 lines less than what the system terminal is capable of. The command sets the default window size based on the **LINES** variable. Also, you can easily adjust the window size for each run of the command by adding the **-n** flag.

Use the **MORE** variable to customize the **more** command with your preferred configuration each time the system starts. This variable accepts **more** command flags.

Flags

Flags Description

Item	Description
-c	Prevents the screen from scrolling, which makes text easier to read as the **more** command writes to the screen. The system ignores the **-c** flag if the terminal cannot clear to the end of a line.
-d	Prints a message, appended to the `More` prompt at the bottom of the screen, about which keys continue, quit, and provide help for the **more** command. Displays error messages rather than ringing the terminal bell if an unrecognized command is used. This is helpful for inexperienced users.
-e	Exits automatically after displaying the last line of the last file.
-H	Disables the search pattern highlighting feature by default.
-i	Searches for patterns without considering uppercase and lowercase.
-l	Pauses after detecting a page break in the input. If the `-l` flag is not used, the **more** command pauses to accept commands after any line containing a ^L (CTRL-L) character. Also, if a file begins with a FORMFEED, the screen is cleared before the file is printed.
-N	Suppresses line numbering. The default display, with line numbers, can slow the **more** command's performance on very large input files. The line numbering feature displays the line number in the **=** subcommand and passes the line number to the editor (if it is the **vi** editor).

Flags Description *(continued)*

Item	Description
-n *Number*	Configures the **more** command to display the specified number of lines in the window. Without the **-n** flag, the **more** command defaults to two lines less than what the terminal is capable of. For example, on a 24-line terminal, the default is 22 lines. The `-n` option overrides any values obtained from the environment.
-p *Subcommand*	Starts the **more** command and specified subcommand for each *File* operand. For example, **more -p 50j text1 text2** displays the **text1** file at the fiftieth line; then does the same for the **text2** file when you finish the first. See "Subcommands" for descriptions of **more** subcommands. If the command is a positioning command, such as a line number or a regular expression search, set the current position to represent the final results of the command, without writing any intermediate lines of the file. For example, the two commands: `more -p 1000j` *`filename`* `more -p 1000G` *`filename`* are functionally the same and will start the display with the current position at line 1000, passing the lines that j would write and would scroll off the screen if it had been issued during the file examination. If the positioning command is unsuccessful, the first line in the file will be the current position.
-s	Reduces multiple blank lines in the output to only one blank line. The **-s** flag is particularly helpful in viewing output from the **nroff** command.
-t *Tagstring*	Displays the portion of the file that contains the specified tag. This flag works only on files containing tags created with the **ctags** command.
-u	Prevents the **more** command from treating a backspace character as a printable control character (displayed as a ^H (CTRL-H)), suppressing backspacing, underlining, or creating reverse video text for underlined information in a source file. The **-u** flag also forces the **more** command to recognize a carriage-return character, if it exists, at the end of a line.
-v	Suppresses the graphical translation of nonprinting characters. Without the **-v** flag, the **more** command graphically interprets all non-ASCII and most control characters, except Tab, Backspace, and Return. For example, if you do not use the **-v** flag, the **more** command displays the non-ASCII characters Ctrl-x as ^X and x as `M-x`.

Flags Description *(continued)*

Item	Description
-W *Option*	Provides the specified *Option* to the **more** command as an extension: **notite** Prevents the **more** command from sending the terminal initialization string (either the **ti termcap** or the **smcup terminfo** capability) before displaying the file. This option also prevents the **more** command from sending the terminal de-initialization string (either the **te termcap** or the **rmcup terminfo** capability) before exiting. **tite** Causes the **more** command to send the initialization and de-initialization strings. This is the default. These options control whether the **more** command sends the initialization strings described, which, for certain terminals (such as some xterms), cause the **more** command to switch to an alternative screen. The effect of switching screens is to erase the display of the file you were viewing.
-x *Tabs*	Sets tab stops at the specified *Tabs* position. The default tab setting is 8 columns.
-z	Graphically displays the Tab, Backspace, and Return control characters. With the **-z** flag, the **more** command translates the Backspace character as ^H, Return as ^M, and Tab as ^I.

Subcommands

The **more** command accepts subcommands when the command pauses and as parameters for the **-p** flag. Many subcommands take an optional integer, symbolized here by *K*, which you must enter before the subcommand, with no space between. The **more** command, in the paused state, processes subcommands immediately and does not require you to press the Enter key.

The **more** command uses the following subcommands:

more command

Item	Description
h	Displays a help screen that describes the **more** subcommands.
v	Starts the **vi** editor, editing the current file in the current line.
r or **^L**	Refreshes the display.
R	Refreshes the display and removes buffered input.
[*K*](Spacebar)	Moves forward *K* lines when you press the spacebar. If you do not give a value for *K*, pressing the spacebar displays the next full screen by default. This spacebar subcommand is the same as [*K*]**f** or [*K*]**^F** or [*K*]**z**.
[*K*]**f** or [*K*]**^F** or [*K*]**z**	Moves forward *K* lines, or one full screen if you do not give a value for *K*.
[*K*]**b** or [*K*]**^B**	Moves backward *K* lines, or one full screen if you do not give a value for *K*.
[*K*]**d** or [*K*]**^D**	Moves forward *K* lines, or half a screen if you do not give a value for *K*. If you give a value for *K*, the **more** command sets the **d** and **u** scroll size to *K* lines for the session.

Item	Description
[*K*]**u** or [*K*]**^U**	Moves backward *K* lines, or half a screen if you do not give a value for *K*. If you give a value for *K*, the **more** command sets the **d** and **u** scroll size to *K* lines for the session.
[*K*]**j** or [*K*](Enter) or [*K*]**^E**	Moves forward *K* lines, or one line if you do not give a value for *K*.
[*K*]**k** or [*K*]**^Y**	Moves backward *K* lines, or one line if you do not give a value for *K*.
[*K*]**g**	Moves to the beginning of the file, unless you give a line number for *K*. The default for *K* is line number 1.
[*K*]**G**	Moves to the last line in the file, unless you give a line number for *K*. The default for *K* is the last line in the file.
[*K*]**p** or [*K*]**%**	Moves to the point in the file that is *K* percent of the total file. The default for *K* is one percent, or the first line in the file.
m*a-z*	Marks the current position in the file with the specified letter.
'*a-z*	(Single quote) Moves to the position marked with the specified letter.
''	(Two single quotes) Returns to the position from which the last large movement (moving more than a page) command was run. If no such movements have been made, returns to the beginning of the file.
[*K*]**/***pattern*	(Slash) Searches forward, from the current position, for the specified occurrence of the specified pattern of characters. The default value for *K* is the first occurrence.
[*K*]**/!***pattern*	(Slash, exclamation mark) Searches forward, from the current position, for the specified occurrence of a line that does not contain the specified pattern of characters. The default value for *K* is the first occurrence.
[*K*]**?***pattern*	(Question mark) Searches backward, from the current position, for the specified occurrence of the specified pattern of characters. The default value for *K* is the first occurrence.
[*K*]**?!***pattern*	(Question mark, exclamation mark) Searches backward, from the current position, for the specified occurrence of a line that does not contain the specified pattern of characters. The default value for *K* is the first occurrence.
[*K*]**n**	Repeats the last search, specifying an occurrence of the pattern (or an occurrence *not* containing the pattern if the search subcommand included **!**). The default value for *K* is the first occurrence.
:a	Lists the file or files you named in the **more** command line.

more command *(continued)*

Item	Description
:f or **^G** or **=**	Displays information about the current file: • file name • order of the file in the list of files • current line number • current position in the file, given as a percentage • current byte number and total bytes to display.
:e[*File*] or **E**[*File*]	Examines the specified file, provided you named it in the **more** command line.
[*K*]**:n** or [*K*]**N**	Examines either the next file (if you do not give a value for *K*) or the file *K* number of positions forward in the list of files you named in the **more** command line.
[*K*]**:p** or [*K*]**P**	Examines either the previous file (if you do not give a value for *K*) or the file *K* number of positions backward in the list of files you named in the **more** command line.
:t *Tagstring*	Displays the portion of the file that contains the specified tag. This subcommand works only on files containing tags created with the **ctags** command. The **:t** subcommand is the interactive version of the **-t** flag.
:q or **q** or **Q**	Exits the **more** command.
:!*command* or **!***command*	Starts the specified command in a new shell.
H	Toggles the search pattern highlighting feature on or off.

Exit Status

This command returns the following exit values:

Exit Status

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To view a file named `myfile`, enter:

```
more myfile
```

2. To view output from the **nroff** command, enter:

```
ls -l | more
```

3. To view each file starting at its last screen, enter:

```
more -p G file1 file2
```

4. To view each file with the 100th line at the current position, enter:

```
more -p 100 file1 file2
```

Typically, the current position in a **more** command display is the third line on the screen. In this example, the first line on the screen is the 98th line in the file.

5. To view each file starting with the first line that contains the `foo` string, enter:

```
more -p /foo file1 file2
```

The **more** command displays the line in the current position, the third line on the screen.

Files

Files

Item	Description
/usr/share/lib/terminfo	Indicates the terminal information database.

mosy Command

Purpose

Converts the ASN.1 definitions of Structure and Identification of Management Information (SMI) and Management Information Base (MIB) modules into objects definition files for the **snmpinfo** command.

Syntax

mosy -o *output_defs_file* [**-s**] *inputfile* ...

mosy -x *output_desc_file* [**-o** *output_defs_file*] [**-s**] *inputfile* ...

mosy -c *output_c_file* [**-x** *output_desc_file*] [**-o** *output_defs_file*] [**-s**] *inputfile* ...

Description

The **mosy** command reads in the ASN.1 definitions of SMI and MIB modules and produces objects definition files in specific formats. The resulting objects definition files are used by the **snmpinfo** command.

The *inputfile* parameter files are required to be in the **smi.my** or **mibII.my** format. Sample files are the **/usr/samples/snmpd/smi.my** and **/usr/samples/snmpd/mibII.my** files. See the **smi.my** and the **mibII.my** files for information on the required format of the file specified by the *inputfile* parameter.

The **mosy -o** command is used to create the objects definition file specified by the *output_defs_file* parameter for the **snmpinfo** command. This file is normally the **/etc/mib.defs** file.

The objects definition file can be created with one pass of the **mosy** compiler if the **smi.my** and **mibII.my** files are both specified as *inputfile* parameters. The **smi.my** file must precede the **mibII.my** file on the command line.

The **mosy -o** command can also be used to create subfiles. If subfiles are created separately from the SMI and MIB modules, you must concatenate the various subfiles before the **snmpinfo** command can successfully use the resultant **mib.defs** file. The SMI subfile must be at the top of the final objects definition file.

You can add objects definitions for experimental MIB modules or private-enterprise-specific MIB modules to the **/etc/mib.defs** file, but you must first obtain the private MIB module from the vendor that supports those MIB variables.

To update the **/etc/mib.defs** file to incorporate a vendor's private or experimental MIB objects definitions, create a subfile and then concatenate that subfile to the existing MIB II **/etc/mib.defs** file. See example 3.

Flags

Item	Description
-c *output_c_file*	Creates a C code file.
-o *output_defs_file*	Defines the path and file name of the MIB objects definition file for the **snmpinfo** command. There is no default path and file name for this flag. If this flag is not specified, the objects definition file is not created.
-s	Suppresses the conversion verification messages. If this flag is not specified, the conversion verification messages are printed to standard output.
-x *output_desc_file*	Creates a description file in **mib.desc** file format.

Parameters

Item	Description
inputfile	Defines the ASN.1 object definitions module for input to the **mosy** compiler. This file can be formatted according to either the **smi.my** or **mibll.my** file format.

Examples

1. To create an objects definition file for use by the **snmpinfo** command with one pass of the **mosy** command, enter:

   ```
   mosy -o /etc/mib.defs /usr/samples/snmpd/smi.my
   /usr/samples/snmpd/mibII.my
   ```

   In this example, **/usr/samples/snmpd/smi.my** and **/usr/samples/snmpd/mibII.my** are both specified as input files and the resultant objects definition file is the **/etc/mib.defs** file.

2. To create objects definition subfiles, enter:

   ```
   mosy -o /tmp/smi.obj /usr/samples/snmpd/smi.my
   mosy -o /tmp/mibII.obj /usr/samples/snmpd/mibII.my
   cat /tmp/smi.obj /tmp/mibII.obj > /etc/mib.defs
   ```

   In this example, the first command creates an SMI objects file, `/tmp/smi.obj`, from the **/usr/samples/snmpd/smi.my** file. The second command creates the MIB objects definition file, `/tmp/mibII.obj`, from the **/usr/samples/snmpd/mibII.my** file. The final command concatenates the subfiles, placing the SMI objects definition file first in the resultant **/etc/mib.defs** file.

3. To add private enterprise specific MIB objects definitions to an existing **/etc/mib.defs** file for use by the **snmpinfo** command, enter:

   ```
   mosy -o /tmp/private.obj /tmp/private.my
   cat /etc/mib.defs /tmp/private.obj > /tmp/mib.defs
   mv /tmp/mib.defs /etc/mib.defs
   ```

   In this example, the first command creates the `/tmp/private.obj` objects definition file. The second command concatenates the **/etc/mib.defs** MIB objects definition file with the `/tmp/private.obj` private MIB file and places the concatenated contents into the `/tmp/mib.defs` temporary MIB objects definition file. The final command moves the temporary file to the **/etc/mib.defs** file for use by the **snmpinfo** command.

4. To create a description file in **/tmp/smi.desc**, a C code file named **/tmp/smi.c**, and a Management Information Base (MIB) definition file named **/tmp/smi.defs**, enter:

```
mosy  -x /tmp/smi.desc -c /tmp/smi.c -o /tmp/smi.defs -s smi.my mibII.my
```

Files

Item	Description
/etc/mib.defs	Defines the Management Information Base (MIB) variables the SNMP agent should recognize and handle. The format of the **/etc/mib.defs** file is required by the **snmpinfo** command.
/usr/samples/snmpd/smi.my	Defines the ASN.1 definitions by which the SMI is defined as in RFC 1155.
/usr/samples/snmpd/mibII.my	Defines the ASN.1 definitions for the MIB II variables as defined in RFC 1213.

mount Command

Purpose

Makes a file system available for use.

Syntax

mount [**-f**] [**-n** *node*] [**-o** *options*] [**-p**] [**-r**] [**-v** *vfsname*] [**-F** *AltFSfile*] [**-t** *type* | [*device* | *node***:***directory*] *directory* | **all** | **-a**] [**-V** [*generic_options*] *special_mount_points*]

Description

The **mount** command instructs the operating system to make a file system available for use at a specified location (the mount point). In addition, you can use the **mount** command to build other file trees made up of directory and file mounts. The **mount** command mounts a file system expressed as a device using the *device* or *node***:***directory* parameter on the directory specified by the *directory* parameter. After the **mount** command has finished, the directory specified becomes the root directory of the newly mounted file system.

Only users with root authority or are members of the system group and have write access to the mount point can issue file or directory mounts. The file or directory may be a symbolic link. The **mount** command uses the real user ID, not the effective user ID, to determine if the user has appropriate access. System group members can issue device mounts, provided they have write access to the mount point and those mounts specified in the **/etc/file systems** file. Users with root user authority can issue any **mount** command.

Users can mount a device provided they belong to the system group and have appropriate access. When mounting a device, the **mount** command uses the *device* parameter as the name of the block device and the *directory* parameter as the directory on which to mount the file system.

If you enter the **mount** command without flags, the command displays the following information for the mounted file systems:

- the node (if the mount is remote)
- the object mounted
- the mount point
- the virtual-file-system type
- the time mounted

- any mount options

If you specify only the *directory* or *node*:*directory* parameter, the **mount** command takes it to be the name of the directory or file on which a file system, directory, or file is usually mounted (as defined in the **/etc/file systems** file). The **mount** command looks up the associated device, directory, or file and mounts it. This is the most convenient way of using the **mount** command, because it does not require you to remember what is normally mounted on a directory or file. You can also specify only the device. In this case, the command obtains the mount point from the **/etc/file systems** file.

The **/etc/file systems** file should include a stanza for each mountable file system, directory, or file. This stanza should specify at least the name of the file system and either the device on which it resides or the directory name. If the stanza includes a mount attribute, the **mount** command uses the associated values. It recognizes five values for the mount attributes: **automatic**, **true**, **false**, **removable**, and **readonly**.

The **mount all** command causes all file systems with the **mount=true** attribute to be mounted in their normal places. This command is typically used during system initialization, and the corresponding mount operations are referred to as automatic mount operations.

By default, the **mount** command runs the **wlmcntrl** command to refresh the current assignment rules in the kernel after mounting the file system. In some situations (such as when many file systems are mounted at once, or when a rule for an inaccessible remote mount is present in the workload manager configuration), calling **wlmcntrl** automatically after mount might be undesirable.

If you wish to override this behavior, set the environment variable *MOUNT_WLMCNTRL_SELFMANAGE* to any value. This will avoid calling the **wlmcntrl** command during the mount operation. You must manually run `wlmcntrl -u -d ""` to refresh the current assignment rules in the kernel. For more information, see **wlmcntrl** command.

Note:

1. If the **cdromd** CD and DVD **automount** daemon is enabled, those devices will be automatically mounted as specified in the **/etc/cdromd.conf** file. Use the **cdumount** or **cdeject** command to unmount an automatically mounted CD or DVD. Use **stopsrc -s cdromd** to disable the CD/DVD **automount** daemon.
2. For CacheFS, the remote file system that is to be cached locally must be exported such that the root ID of the local system is not remapped on the remote host to nobody (or the ID that the remote host uses as the anonymous user). For example, if host A were to export a file system **/F**, which would be mounted with CacheFS on host B, then the **/etc/exports** on host A would need to have an entry similar to:

   ```
   /F -rw,root=B
   or
   /F -ro,root=B
   ```

   depending on the mount options used for the local CacheFS mount.
3. Mounting a JFS file system on a read-only logical volume is not supported.
4. Mounting a JFS2 file system with EAv1 on Trusted AIX system converts the file system to EAv2.

Using mount on a JFS2 File System

The **mount** command can also be used to access a snapshot of a JFS2 file system as a directory tree. The snapshot on *device* is mounted read-only at *directory*. A snapshot can only be mounted once. When mounting a JFS2 file system with snapshots, the snapshots are activated.

You can use the System Management Interface Tool (SMIT) **smit mount** fast path to run this command.

Note: If the **mount** command encounters a Journaled File System (JFS) or Enhanced Journaled File System (JFS2) which was not unmounted before reboot, a replay of any JFS or JFS2 log records is attempted. In order to move a compatible JFS file system to a system running an earlier release of the operating system, the file system must always be unmounted cleanly prior to its movement. Failure to unmount first may result in an incompatible JFS log device. If the movement results in an unknown log device, the file system should be returned to the system running the latter operating system release, and **fsck** should be run on the file system.

Flags

Item	Description
-a	Mounts all file systems in the **/etc/file systems** file with stanzas that contain the **true** mount attribute.
all	Same as the **-a** flag.
-f	Requests a forced mount during system initialization to enable mounting over the root file system.
-F *AltFSfile*	Mounts on a file of an alternate file system, other than the **/etc/file systems** file.
-n *node*	Specifies the remote node that holds the directory to be mounted. The node can be specified as a colon-separated IPv6 address. If this is done with the `node:directory` format, the colon-separated IPv6 address must be enclosed in square brackets.
-p	Mounts a file system as a removable file system. While open files are on it, a removable mounted file system behaves the same as a normally mounted file system. However, when no files are open (and no process has a current directory on the file system), all of the file system disk buffers in the file system are written to the medium, and the operating system forgets the structure of the file system.
-r	Mounts a file system as a read-only file system, regardless of its previous specification in the **/etc/file systems** file or any previous command-line options.
-t *type*	Mounts all stanzas in the **/etc/file systems** file that contain the **type=***type* attribute and are not mounted. The *type* parameter specifies the name of the group.
-v *vfsname*	Specifies that the file system is defined by the *vfsname* parameter in the **/etc/vfs** file.

File System Specific Options

Item	Description
-o *options*	Specifies options. Options entered on the command line should be separated only by a comma. The following file system-specific options do not apply to all virtual file system types: **atime** Turns on access-time updates. If neither **atime** nor **noatime** is specified, **atime** is the default value. **bsy** Prevents the mount operation if the directory to be mounted over is the current working directory of a process. **cio** Specifies the file system to be mounted for concurrent readers and writers. I/O on files in this file system will behave as if they had been opened with **O_CIO** specified in the **open()** system call. Using this option will prevent access in any manner other than CIO. It is impossible to use cached I/O on a file system mounted with the cio option. This means that mapping commands such as `mmap()` and `shmat()` will fail with EINVAL when used on any file in a file system mounted with the `cio` option. One side-effect of this is that it is impossible to run binaries out of a `cio` mounted file system, since the loader may use `mmap()`. **Note:** When you mount the file system by using the `cio` option, all applications must manage the serialization of files. Quotas are not supported by the `cio` option because quotas have their own serialization code.

Item	Description
	dev Specifies that you can open devices from this mount. If neither **dev** nor **nodev** is specified, **dev** is the default value.
	dio Specifies that I/O on the file system will behave as if all the files had been opened with **O_DIRECT** specified in the **open()** system call. **Note:** Using the **-odio** or **-ocio** flags can help performance on certain workloads, but users should be aware that using these flags will prevent file caching for these file systems. Because readahead is disabled for these file systems, this may decrease performance for large sequential reads.
	fmode=*octal* Specifies the mode for a file and directory. The default is 755.
	gid=gid Specifies the GID that is assigned to files in the mount. The default is **bin**.
	log=*lvname* Specifies the full path name of the file system logging logical volume name where the following file-system operations are logged.
	log=*NULL* Turns off logging for JFS2 file systems. JFS2 depends on the log for metadata consistency, so if the system abnormally stops during JFS2 metadata operations, the file system cannot be recovered to a consistent state upon reboot. In these cases, the file system must be recreated.  **Attention:** Because of the risk of data loss, use this flag with caution.
	maxpout=*value* Specifies the pageout level for files on this file system at which threads should be slept. If **maxpout** is specified, **minpout** must also be specified. Value must be non-negative and greater than **minpout**. The default is the kernel **maxpout** level.
	minpout=*value* Specifies the pageout level for files on this file system at which threads should be readied. If **minpout** is specified, **maxpout** must also be specified. Value must be non-negative. The default is the kernel **minpout** level.
	noatime Turns off access-time updates. Using this option can improve performance on file systems where a large number of files are read frequently and seldom updated. If you use the option, the last access time for a file cannot be determined. If neither **atime** nor **noatime** is specified, **atime** is the default value.
	nocase Turns-off case mapping. This is useful for CDROMs using the ISO 9660:1998/HSG standard.
	nodev Specifies that you cannot open devices from this mount. This option returns a value of **ENXIO** if a failure occurs. If neither **dev** nor **nodev** is specified, **dev** is the default value.

Item	Description
	noguard Mount the filesystem regardless of the current mountguard setting which would otherwise guard the filesystem against unsupported concurrent mounts in a PowerHA or other clustering environment. If mountguard is enabled by the **chfs** or **crfs** command, the filesystem cannot be mounted if it appears to be mounted on another node or system. Specifying the noguard option temporarily overrides the mountguard setting.
	norbr Mounts the file system without the release-behind-when-reading capability. If none of the release-behind options are specified, **norbrw** is the default value.
	norbrw Mounts the file system without both the release-behind-when-reading and release-behind-when-writing capabilities. If none of the release-behind options are specified, **norbrw** is the default value.
	norbw Mounts the file system without the release-behind-when-writing capability. If none of the release-behind options are specified, **norbrw** is the default value.
	nosuid Specifies that execution of **setuid** and **setgid** programs by way of this mount is not allowed. This option returns a value of **EPERM** if a failure occurs. If neither **suid** nor **nosuid** is specified, **suid** is the default value.
	rbr Mount file system with the release-behind-when-reading capability. When sequential reading of a file in this file system is detected, the real memory pages used by the file will be released once the pages are copied to internal buffers. If none of the release-behind options are specified, **norbrw** is the default. **Note:** When **rbr** is specified, the **D_RB_READ** flag is ultimately set in the **_devflags** field in the **pdtentry** structure.
	rbw Mount file system with the release-behind-when-writing capability. When sequential writing of a file in this file system is detected, the real memory pages used by the file will be released once the pages written to disk. If none of the release-behind options are specified, **norbrw** is the default. **Note:** When **rbw** is specified, the **D_RB_WRITE** flag is set.
	rbrw Mount file system with both release-behind-when-reading and release-behind-when-writing capabilities. If none of the release-behind options are specified, **norbrw** is the default. **Note:** If **rbrw** is specified, both the **D_RB_READ** and the **D_RB_WRITE** flags are set.

Item	Description

remount

Changes the mount options of a mounted file system. For JFS2 file systems, you can specify the following mount options with the remount option to change the settings of a mounted file system. For any mount options not specified, no change is made to the current corresponding settings of the file system.

atime, **noatime**; **dev**, **nodev**; **maxpout**, **minpout**; **rbr**, **norbr**; **rbw**, **norbw**; **rbrw**, **norbrw**, **rw**, **ro**, **rox**; **suid**, **nosuid**.

Note:

1. External-snapshot mounted file systems cannot be remounted to read-write file systems.
2. You cannot use the **rw** and **ro** remount options on a file system that is managed by data management application programming interface (DMAPI).

For NFS, there are three types of mount requests.

duplicate mount

If the node, object, mount point, and the options that are specified in the **mount** command are the same as those for an existing mount, the **mount** command returns information about a successful mount, but a new mount is not created.

new mount

If the remount option is not specified, the **mount** command creates a new mount. If the node, object, mount point, or the constant options that are specified in the **mount** command are different than those for the existing mounts, the **mount** command fails if the remount option is specified.

remount

If the node, object, and mount point are the same as those for a top-most mount, but the remount options are different, the remount operation modifies the mount options of an existing mount. In this case, NFS performs the remount operation.

A top-most mount does not have another mount on top of it. For remount requests, the following options can be modified: **acdirmax**, **acdirmin**, **acregmax**, **acregmin**, **actimeo**, **fastattr**, **grpid**, **hard**, **intr**, **noac**, **nocto**, **nodev**, **nointr**, **nosuid**, **posix**, **retrans**, **ro**, **rsize**, **rw**, **secure**, **sec**, **soft**, **timeo**, **wsize**, **biods**, **extraattr**, **nodircache**, **prefer**, **otwattr**, **maxgroups**, and **proto**. Other options are classified as constant options.

ro

Specifies that the mounted file is read-only, regardless of its previous option specification in the **/etc/file systems** file or any previous command-line options. The default value is **rw**.

rw

Specifies that the mounted file is read/write accessible, regardless of its previous option specification in the **/etc/file systems** file or any previous command-line options. The default value is **rw**.

snapshot

Specifies the *device* to be mounted is a snapshot. The snapped file system for the specified snapshot must already be mounted or an error message will display.

Item	Description
	snapto=*snapshot* Specifies the location to start a snapshot with the value of *snapshot* when mounting the specified JFS2 file system. The *snapshot* parameter specifies the name of an internal snapshot if the *snapshot* parameter does not included a forward slash (/), that is, no path information.
	suid Specifies that execution of **setuid** and **setgid** programs by way of this mount is allowed. If neither **suid** nor **nosuid** is specified, **suid** is the default value.
	upcase Changes case mapping from default lowercase to uppercase. This is useful for CDROMs using the ISO 9660:1998/HSG standard.
	uid=uid Specifies the UID that is assigned to files in the mount, the default is bin.
	wrkgrp=workgroup Specifies the workgroup that the SMB server belongs.

NFS Specific Options

Item	Description
-o *options*	Specifies options. Options you enter on the command line should be separated only by a comma, not a comma and a space. The following NFS-specific options do not apply to all virtual file system types:
	acdirmax=*n* Holds cached attributes for no more than *n* seconds after directory update. The default is 60 seconds.
	acdirmin=*n* Holds cached attributes for at least *n* seconds after directory update. The default is 30 seconds.
	acl Requests using the Access Control List RPC program for this NFS mount. If the **acl** option is used, the ACL RPC program is used only if the NFS server provides it. The default is **noacl**.
	acregmax=*n* Holds cached attributes for no longer that *n* seconds after file modification. The default is 60 seconds.
	acregmin=*n* Holds cached attributes for at least *n* seconds after file modification. The default is 30 seconds.
	actimeo=*n* Sets minimum and maximum times for regular files and directories to *n* seconds. If this option is set, it overrides any settings for the **acregmin**, **acregmax**, **acdirmin**, and **acdirmax** options.
	bg Attempts mount in background if first attempt is unsuccessful. The default value is **fg**.

Item	Description

biods=*n*

Sets the maximum number of `biod` threads that perform asynchronous I/O RPC requests for an NFS mount. The maximum value that can be set is 128. Values greater than 128 are limited to 128 within the NFS client. The NFS client dynamically manages the number of running `biod` threads up to the maximum based on activity. The default maximums for the different NFS protocols are 7 for NFS version 2 and 32 for NFS version 3 and NFS version 4. These defaults are subject to change in future releases.

cio

Specifies the file system to be mounted for concurrent readers and writers. I/O on files in this file system will behave as if they had been opened with **O_CIO** specified in the **open()** system call. Using this option will prevent access in any manner other than CIO. It is impossible to use cached I/O on a file system mounted with the cio option. This means that mapping commands such as `mmap()` and `shmat()` will fail with EINVAL when used on any file in a file system mounted with the `cio` option. One side-effect of this is that it is impossible to run binaries out of a `cio` mounted file system, since the loader may use `mmap()`.

Note: When you mount the file system by using the `cio` option, all applications must manage the serialization of files. Quotas are not supported by the `cio` option because quotas have their own serialization code.

cior

Specifies to allow read-only files to open in the file system. I/O on files in this file system will behave as if they had been opened with **O_CIO | O_CIOR** specified in the **open()** system call. Using this option will prevent access in any manner other than **O_CIO | O_CIOR** and read-only. An attempt to open with **O_CIO** only will also fail. This option can only be used in conjunction with `cio`.

dio

Specifies that I/O on the file system will behave as if all the files had been opened with **O_DIRECT** specified in the **open()** system call.

Note: Using the **-odio** or **-ocio** flags can help performance on certain workloads, but users should be aware that using these flags will prevent file caching for these file systems. Because readahead is disabled for these file systems, this may decrease performance for large sequential reads.

fastattr

Bypasses the requirement that files currently being written will be sent to the server before the attributes of the file is read. This option is to be used with caution, since it will cause the client to assume that the file data that has not yet reached the server will be written without problem. In case of write errors, the client and server will have different opinions on what the size of the file really is. Likewise, a client will not be aware of attribute changes to the file being made by another client, so this option must not be used in environments where two clients are writing to the same files.

fg

Attempts mount in foreground if first attempt is unsuccessful. **fg** is the default value.

grpid

Directs any file or directory created on the file system to inherit the group ID of the parent directory.

Item	Description

hard
Retries a request until server responds. The option is the default value.

intr
Allows keyboard interrupts on hard mounts.

llock
Requests that files lock locally at the NFS client. NFS network file locking requests are not sent to the NFS server if the **llock** option is used.

maxgroups=*n*
Indicates that NFS RPC calls using **AUTH_UNIX** may include up to *n* member groups of information. Using this option to increase the number of member groups beyond the RPC protocol standard of 16 will only work against servers that support more than 16 member groups. Otherwise, the client will experience errors.

Values below 16 or greater than 64 will be ignored. By default, the protocol standard maximum of 16 is adhered to. AIX NFS servers will accept and process **AUTH_UNIX** credentials with up to 64 groups starting with AIX 5L Version 5.2 with the 5200-01 Recommended Maintenance package. The actual number of member groups sent by the NFS client is dependent on the number of groups the involved user is a member of, and may be limited by the length of the NFS client's hostname (which is included in the **AUTH_UNIX** information).

noac
Specifies that the **mount** command performs no attribute or directory caching. If you do not specify this option, the attributes (including permissions, size, and timestamps) for files and directories are cached to reduce the need to perform over-the-wire **NFSPROC_GETATTR** Remote Procedure Calls (RPCs). The **NFSPROC_GETATTR** RPC enables a client to prompt the server for file and directory attributes. The **acregmin**, **acregmax**, **acdirmin**, and **acdirmax** options control the length of time for which the cached values are retained.

noacl
Specifies not to use the Access Control List RPC program for this NFS mount request. The default is **noacl**.

nointr
Specifies no keyboard interrupts allowed on hard mounts.

port=*n*
Sets server Internet Protocol (IP) port number to n. The default value is the 2049.

posix
Requests that pathconf information be exchanged and made available on an NFS Version 2 mount. Requires a mount Version 2 **rpc.mountd** at the NFS server.

proto=[udp|tcp]
Specifies the transport protocol. The default is **tcp**. Use the **proto=[udp|tcp]** option to override the default.

proto=udp cannot be specified if **vers=4**.

Item	Description

retrans=*n*
Sets the number of NFS transmissions to *n*. The default value is 5. The `retrans` setting determines how many times the NFS client retransmits a given UDP RPC request to an NFS server for file system operations. The `retrans` setting is not used during communication with the NFS server `rpc.mountd` service when processing NFS version 2 and 3 mounts. Retries to `rpc.mountd` are controlled with the `retry mount` option.

retry=*n*
Sets the number of times the mount is attempted to *n*; the default value is 1000. When the retry value is 0, the system makes 10,000 attempts.

rsize=*n*
Sets the read buffer size to *n* bytes. Beginning with AIX Version 6.1, the default value is 64 KB and the maximum value is 512 KB when using Version 3 and Version 4 of the NFS protocol.

secure
Specifies that the **mount** command uses Data Encryption Standard (DES) for NFS transactions. Data Encryption Standard (DES) is not supported in NFS Version 4, use *krb5* instead.

sec=*flavor*[:*flavor*...]
Specifies a list of security methods that may be used to access files under the mount point. Allowable flavor values are:

sys
UNIX authentication. This is the default method.

dh
DES authentication. Data Encryption Standard (DES) is not supported in NFS Version 4, use *krb5* instead.

krb5
Kerberos. Authentication only.

krb5i
Kerberos. Authentication and integrity.

krb5p
Kerberos. Authentication, integrity, and privacy.

The **secure** option may be specified, but not in conjunction with a **sec** option. The **secure** option is deprecated and may be eliminated in a future release. Use **sec=dh** instead.

sec=[*flavor1:...:flavorn*]
The `sec` option specifies the security flavor list for the NFS mount. The available flavors are `des`, `unix`, `sys`, `krb5`, `krb5i`, and `krb5p`. This option only applies to AIX 5.3 or later.

shortdev
Specifies that you are mounting a file system from a host that does not support 32-bit device special files.

soft
Returns an error if the server does not respond. The default value is **hard**.

Item	Description
	timeo=*n* Sets the Network File System (NFS) time out period to *n* tenths of a second. For TCP mounts, the default timeout is 100, which equals 10 seconds. For UDP mounts, the default timeout is 11, which equals 1.1 seconds, but varies depending on the NFS operation taking place. For UDP mounts, the timeout will increase for each failed transmission, with a maximum value of 20 seconds. Each transmission will be attempted twice, after which the timeout value is updated. The `timeo` option does not apply to communication from the NFS client to the `rpc.mountd` service on NFS servers. A timeout of 30 seconds is used when making calls to `rpc.mountd`.
	vers=[2|3|4] Specifies NFS version. The default is the version of NFS protocol used between the client and server and is the highest one available on both systems. If the NFS server does not support NFS Version 3, the NFS mount will use NFS Version 2. Use the `vers=[2|3|4]` option to select the NFS version. By default, the NFS mount will never use NFS Version 4 unless specified. The `vers=4` only applies to AIX 5.3 or later.
	wsize=*n* Sets the write buffer size to *n* bytes. Beginning with AIX Version 6.1, the default value is 64 KB and the maximum value is 512 KB when using Version 3 and Version 4 of the NFS protocol.

CacheFS Specific Options

The CacheFS-specific version of the **mount** command mounts a cached file system; if necessary, it NFS-mounts its back file system. It also provides a number of CacheFS-specific options for controlling the caching process.

To mount a CacheFS file system, use the **mount** command with the **-V** flag followed by the argument. The following **mount** flags are available.

The following arguments to the **-o** flag are specifically for CacheFS mounts. Options you enter on the command line should be separated only by a comma, not a comma and a space.

Note: The **backfstype** argument must be specified.

Item	Description
-o	Specifies options.
	acdirmax=*n* Specifies that cached attributes are held for no more than *n* seconds after directory update. Before *n* seconds, CacheFS checks to see if the directory modification time on the back file system has changed. If it has, all information about the directory is purged from the cache and new data is retrieved from the back file system. The default value is 60 seconds.
	acdirmin=*n* Specifies that cached attributes are held for at least *n* seconds after directory update. After *n* seconds, CacheFS checks to see if the directory modification time on the back file system has changed. If it has, all information about the directory is purged from the cache and new data is retrieved from the back file system. The default value is 30 seconds.

Item	Description
	acregmax=*n*
	Specifies that cached attributes are held for no more than n seconds after file modification. After *n* seconds, all file information is purged from the cache. The default value is 30 seconds.
	acregmin=*n*
	Specifies that cached attributes are held for at least *n* seconds after file modification. After *n* seconds, CacheFS checks to see if the file modification time on the back file system has changed. If it has, all information about the file is purged from the cache and new data is retrieved from the back file system. The default value is 30 seconds.
	actimeo=*n*
	Sets **acregmin**, **acregmax**, **acdirmin**, and **acdirmax** to *n*.
	backfstype=*file_system_type*
	The file system type of the back file system (for example, nfs).
	backpath=*path*
	Specifies where the back file system is already mounted. If this argument is not supplied, CacheFS determines a mount point for the back file system.
	cachedir=*directory*
	The name of the cache directory.
	cacheid=*ID*
	ID is a string specifying a particular instance of a cache. If you do not specify a cache ID, CacheFS will construct one.
	demandconst
	Enables maximum cache consistency checking. By default, periodic consistency checking is enabled. When you enable **demandconst**, it checks on every read and write. **Note:** If this option is used the first time a specific CacheFS is mounted, then the option must also be specified for subsequent mounts. There is state information stored in the cache control files that enforces consistent use of this option.
	local_access
	Causes the front file system to interpret the mode bits used for access checking instead or having the back file system verify access permissions. Do not use this argument with secure NFS.
	noconst
	Disables cache consistency checking. By default, periodic consistency checking is enabled. Specify **noconst** only when you know that the back file system will not be modified. Trying to perform cache consistency check using **cfsadmin-s** will result in error. **demandconst** and **noconst** are mutually exclusive. **Note:** If this option is used the first time a specific CacheFS is mounted, then the option must also be specified for subsequent mounts. There is state information stored in the cache control files that enforces consistent use of this option.

Item	Description
	purge Purges any cached information for the specified file system. **Note:** If this option is used the first time a specific CacheFS is mounted, then the option must also be specified for subsequent mounts. There is state information stored in the cache control files that enforces consistent use of this option. **rw | ro** Read-write (default) or read-only. **suid | nosuid** Allows (default) or disallows set-uid execution **write-around | non-shared** Writes modes for CacheFS. The write-around mode (the default) handles writes the same as NFS does; that is, writes are made to the back file system, and the affected file is purged from the cache. You can use the non-shared mode when you are sure that no one else will be writing to the cached file system. **Note:** If this option is used the first time a specific CacheFS is mounted, then the option must also be specified for subsequent mounts. There is state information stored in the cache control files that enforces consistent use of this option. **mfsid** Turns on global view. In NFS v4 system, you can traverse through the exported namespace on the server side. You need to specify this option to go over the file system. **Restriction: mfsid** is an option if the backend file system for CacheFS is NFS v4.
-V	Mounts a CacheFS file system.

Server Message Block (SMB) client file system specific options

Item	Description
-o *options*	Specifies options for mounting the SMB client file system. Options that you enter on the command line must be separated only by a comma. Do not insert a space before or after a comma. The following options are available for the SMB client file system: **fmode** Sets a file or directory to octal mode for access permissions. The default value is 755. **uid** Assigns a user ID to files during the mount operation. The default value is `root`. **gid** Assigns a group ID to files during the mount operation. The default value is `system`. **wrkgrp** Specifies the workgroup to which the SMB server belongs. This parameter is mandatory to mount the SMB client file system. **port** Specifies the port number. The valid values are 445 and 139. The default value is 445. Port 139 is supported only when the specified server address is in IPv4 format. **Note:** `encryption` option is not supported when the port specified is 139. **pver** Specifies version of the SMB protocol that is used to communicate with the SMB server. The valid values are `2.1,3.0.2` and `auto`. For the value `auto`, the SMB protocol version `2.1` or version `3.0.2` is used based on the specified SMB server. **signing** Specifies whether the file system in the SMB client needs digital signature for communication with the SMB server filesystem. The valid values are `enabled` and `required`. When this parameter is set to `enabled`, the file system in the SMB client does not digitally sign the data packets unless the file system in the SMB server needs digital signatures for communication with the file system in the SMB server. When this is set to `required`, the file system in the SMB client must digitally sign the data packets for communication with the file system in the SMB server. If you do not specify the value for the `signing` parameter by using the **mount** command, a default value is used from the tunable parameter values of the kernel that are set by using the **smbctune** command. **secure_negotiate** Specifies whether the file system in the SMB client needs secure dialect negotiation capability. SMB Dialect 3.0.2 implements secure dialect negotiation to protect against security-downgrade attacks. The valid values are `desired`, `required`, and `disabled`. If you do not specify the value by using the **mount** command, a default value is used from tunable parameter values of the kernel that are set by using the **smbctune** command. **encryption** Specifies whether the file system in the SMB client requires data encryption. The valid values are `desired`, `required`, and `disabled`. If you do not specify the value by using the **mount** command, a default value is used from the tunable parameter values of the kernel that are set by using the **smbctune** command. **Note:** `encryption` option is not supported when the port specified is 139.

If the options that are used with the **mount** command (`pver, signing, secure_negotiate, or encryption`) are unspecified by using the **-o** flag, the default values for the **mount** command options are initialized by using the new values of the kernel tunable parameters (`smbc_protocol_version,`

`smbc_signing, smbc_secure_negotiate, smbc_encryption`). The kernel tunable parameters are initialized from tunable parameters defined in the `smbctune.conf` file. These parameters can also be modified by using the **smbctune** command.

The following table shows the kernel tunable parameters of the **mount** command and the corresponding kernel tunable parameters that can be set in the `smbctune.conf` file:

Options of the -o flag (mount command)	Corresponding kernel tunable parameter of the smbctune.con file	Valid values
pver	smbc_protocol_version	2.1, 3.0.2, auto
signing	smbc_signing	enabled, required
secure_negotiate	smbc_secure_negotiate	desired, required, disabled
encryption	smbc_encryption	desired, required, disabled

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list the mounted file systems, enter the following command:

   ```
   mount
   ```

   This command produces output similar to the following:

   ```
   node   mounted          mounted over  vfs    date              options
   ----   -------          ------------ ---  ------------   -------------------
          /dev/hd0         /            jfs   Dec 17 08:04   rw, log  =/dev/hd8
          /dev/hd3         /tmp         jfs   Dec 17 08:04   rw, log  =/dev/hd8
          /dev/hd1         /home        jfs   Dec 17 08:06   rw, log  =/dev/hd8
          /dev/hd2         /usr         jfs   Dec 17 08:06   rw, log  =/dev/hd8
   sue    /home/local/src  /usr/code    nfs   Dec 17 08:06   ro, log  =/dev/hd8
   ```

   For each file system, the **mount** command lists the node name, the device name, the name under which it is mounted, the virtual-file-system type, the date and time it was mounted, and its options.

2. To mount all default file systems, enter the following command:

   ```
   mount all
   ```

   This command sequence mounts all standard file systems in the **/etc/file systems** file marked by the **mount=true** attribute.

3. To mount a remote directory, enter the following command:

   ```
   mount -n nodeA /home/tom.remote /home/tom.local
   ```

   This command sequence mounts the `/home/tom.remote` directory located on nodeA onto the local `/home/tom.local` directory. It assumes the default *VfsName* parameter=**remote**, which must be defined in the **/etc/vfs** file.

4. To mount a file or directory from the **/etc/file systems** file with a specific type, enter the following command:

   ```
   mount -t remote
   ```

This command sequence mounts all files or directories in the **/etc/file systems** file that have a stanza that contains the **type=remote** attribute.

5. To CacheFS-mount the file system which is already NFS-mounted on **/usr/abc**, enter the following command:

```
mount -V cachefs -o backfstype=nfs,backpath=/usr/abc,
cachedir=/cache1 server1:/user2 /xyz
```

The lines similar to the following appear in the **/etc/mnttab** file after the mount command is executed:

```
server1:/user2 /usr/abc nfs
/usr/abc /cache1/xyz cachefs backfstype=nfs
```

6. To mount a snapshot, enter the following command:

```
mount -o snapshot /dev/snapsb /home/janet/snapsb
```

This command mounts the snapshot contained on the **/dev/snapsb** device onto the **/home/janet/snapsb** directory.

7. To mount a file system and create a snapshot, enter the following command:

```
mount -o snapto=/dev/snapsb /dev/sb /home/janet/sb
```

This command mounts the file system contained on the **/dev/sb**device onto the **/home/janet/sb** directory and creates a snapshot for the file system on the **/dev/snapsb**device.

8. To access files on an SMB server as a local file system, enter the following command:

```
mount -v cifs -n pezman/user1/pass1 -o uid=201,fmode=750 /home /mnt
```

9. To mount an SMB client file system as a local mount point, enter the following command:

```
mount -v smbc -n llm140.xyz.com/cec102usr1/Passw0rd     \
-o wrkgrp=SMB_21.FVT,port=445,signing=required /some_share /mnt
```

Where, `llm140.xyz.com` is the Windows server, `cec102usr1` is the Kerberos user name, `Passw0rd` is the password of the Kerberos user, `SMB_21.FVT` is the workgroup, `some_share` is the share point on the Windows system, and `/mnt` is the local mount point.

10. To remount the mounted read-only JFS2 file system to a read-write file system, enter the following command:

```
mount –o remount,rw fsname
```

11. To mount all on a file **/tmp/fs1** of an alternate file system, enter the following command:

```
mount -F /tmp/fs1 all
```

Files

Item	Description
/etc/file systems	Lists the known file systems and defines their characteristics.
/etc/vfs	Contains descriptions of virtual-file-system types.

mountd Daemon

Purpose

Answers requests from clients for file system mounts.

Syntax

/usr/sbin/rpc.mountd [**-n**] [**-N**] [**-x**][**-r**]

Description

The **mountd** daemon is a Remote Procedure Call (RPC) that answers a client request to mount a file system. The **mountd** daemon finds out which file systems are available by reading the **/etc/xtab** file.

In addition, the **mountd** daemon provides a list of currently mounted file systems and the clients on which they are mounted. You can display this list by using the **showmount** command.

The **mountd** daemon listens for requests on the port specified in the **/etc/services** file for the service **mountd**. If the **/etc/services** file does not specify a port, one will be chosen when the daemon starts. For example, adding the lines:

```
mountd   6666/tcp
mountd   6666/udp
```

will cause **mountd** to listen for requests on port 6666.

Examples

The **mountd** daemon is started from the **/etc/rc.nfs** file. The **mountd** daemon can be started and stopped by the following System Resource Controller (SRC) commands:

```
startsrc -s rpc.mountd
stopsrc -s rpc.mountd
```

To change the parameters passed to the **mountd** daemon, use the **chssys** command. For example:

```
chssys -s rpc.mountd -a Argument
```

The change will not take effect until the daemon is restarted.

Flags

Item	Description
-n	Allows clients that use older versions of NFS to mount file systems. This option makes the system less secure. It is the default.
-N	Deny mount requests originating from non-privileged ports. This is the opposite of using the **-n** flag, and is not enabled by default.
-x	Allows mount request without reverse lookup check.
-r	Turn off the **/etc/rmtab** file updates.

Files

Item	Description
/etc/exports	Lists the directories that the server can export.
/etc/inetd.conf	Defines how the **inetd** daemon handles Internet service requests.
/etc/xtab	Lists currently exported directories.
/etc/services	Defines the sockets and protocols used for Internet services. Contains information about the known services used in the DARPA Internet network.

mpcstat Command

Purpose

Displays operational information about a Multi-Protocol Over ATM (MPOA) Client.

Syntax

mpcstat [**-a** **-c** **-e** **-i** **-m** **-r** **-s** **-t** **-v**] [Device_Name]

Description

This command displays Multi-Protocol Over ATM (MPOA) Client operational information gathered by a specified MPOA Client device. If a MPOA Client (MPC) device name is not entered, information for the available MPC appear. You can use the flags to narrow down your search to specify specific categories of information such as Configuration, Egress Cache Entries, Ingress Cache Entries, MPOA Servers, Shortcut Virtual Connections, and Statistics, or you can elect to have all of the information categories display.

You can also toggle debug tracing on or off and reset statistics counters.

Parameters

Item	Description
Device_Name	The name of the MPOA Client device name, for example, *mpc0*.

Flags

Item	Description
-a	Requests that all of the MPOA Client information appear. Note that this flag does not reset statistics counters or toggle trace. If a flag is not entered, the **-a** flag is the default flag.
-c	Requests the configuration
-e	Requests the egress (incoming) cache
-i	Requests the ingress (outgoing) cache.
-m	Requests the list of MPOA Servers in use.
-r	Resets the statistics counters after reading.
-s	Requests the statistics counters.
-t	Toggles full debug trace on or off.
-v	Requests the list of Shortcut Virtual Connections.

The following information appears for all valid calls and contains the following fields:

Device Name
Displays the device name of the MPOA Client.

MPC State
Displays the current state of the MPOA Client.

Example States:

```
Idle              Registering with the ELAN.
Initializing      Registering with the switch.
Operational       Fully operational.
Network Down      Network is currently unavailable.
```

MPC Address
Displays the MPOA Client's 20-byte ATM address for a specific ATM adapter port device name. The adapter port device name is also displayed.

Elapsed Time
Displays the real time period which has elapsed since statistics were last reset.

MPC Configuration
Selected with the **-a** or **-c** flags. Displays the network administrator's pre-configured attributes for the MPOA Client, or the values provided by a Lan Emulation Configuration Server (LECS).

MPC Egress Cache
Selected with the **-a** or **-e** flags. Displays the current egress cache entries. Included are the state of the entry, its Level-3 address, and ATM shortcut address, as well as additional descriptive values associated with each entry.

Example States

```
Active          Has active shortcut connection.
Purging         Purging the egress MPOA Server entry.
DP Purge        Purging the remote MPOA Client data plane.
Inactive        No current activity on shortcut connection.
```

MPC Ingress Cache:
Selected with the **-a** or **-i** flags. Displays the current ingress cache entries. Included are the state of the entry, its Level-3 address, and ATM shortcut address, as well as additional descriptive values associated with each entry.

Example States

```
Flow Detect     Waiting for packet threshold to enable shortcut.
Resolution      Packet threshold reached, resolving shortcut.
Hold Down       Shortcut resolution failed, waiting for retry.
Resolved        Shortcut resolution sequence complete.
```

MPOA Server List
Selected with the **-a** or **-m** flags. Displays a list of MPOA Servers currently known by this MPC. Included in each entry are the name of the LE Client that identified the MPS, the MPS ATM address, and the MPS LAN MAC address.

MPC Statistics
Selected with the **-a** or **-s** flags. Displays the current Transmit, Receive, and General statistics for this MPOA Client.

Shortcut Virtual Connection
Selected with the **-a** or **-v** flags. Displays the current list of shortcut virtual circuits in use by the MPOA client. Included are virtual path and channel values, VC state, ATM device name, as well as additional descriptive values associated with each entry.

Example States:

```
Idle            Call idle.
Signalling      Call placed but not established.
Operational     Call connected; data path valid.
Released        Call released.
Retry           Temporary call failure; will be retried.
Hold Down       Call failure; will be suspended for hold down period.
```

Exit Status

If an invalid *Device_Name* is specified, this command produces error messages stating that it could not connect to the device. Note that MPOA is a protocol extension to the ATM LAN Emulation protocol, and must have a corresponding and available LE Client to be operational. Examples of an invalid device error message are:

```
MPCSTAT: Device is not an MPOA device.
MPCSTAT: No LEC device with MPOA enabled.
MPCSTAT: Device is not available.
```

mpio_get_config Command

Purpose

Displays information about the DS3000 or the DS4000® subsystem that is based on multiple path I/O (MPIO) and the hdisks that are associated with the subsystem.

Syntax

mpio_get_config [**-v**] [**-a**] **-A** | **-l** *<hdisk#>* | **-?** | **-h**

Description

The **mpio_get_config** command displays information about the MPIO-based DS3000 or DS4000 subsystem and the hdisks that are associated with the subsystem.

Specifically, the command displays the information about the subsystem, including the assigned name of the subsystem, the worldwide name of the subsystem, and a list of hdisks in the **Available** state that are associated with the subsystem.

The following information about the hdisk# is displayed:

- hdisk name
- LUN number
- Current ownership
- Preferred path
- Adapter information
- User-assigned label for the volume

Flags

Item	Description
-A	Lists information for all attached subsystems.
-l	Lists information for the subsystem that includes the hdisk that is specified by the *hdisk#* parameter.
-v	Lists additional information about the controller and partition.
-a	Lists the adapter information.

Parameters

Item	Description
hdisk#	Specifies the hdisk name.

Examples

1. To display information about the subsystem that **hdisk11** is a member of, enter the following command:

```
mpio_get_config -l hdisk11
```

The system displays a message similar to the following message:

```
Storage Subsystem Name = 'Twister'

hdisk#           LUN #   Ownership          User Label
hdisk11              0   A (preferred)      1_disk_0
hdisk12              1   A (preferred)      1_disk_1
hdisk13              2   A (preferred)      1_disk_2
hdisk14              3   A (preferred)      1_disk_3
```

2. To display the information about the subsystem that **hdisk11** is a member of, along with the adapter information, enter the following command:

```
mpio_get_config -a -l hdisk11
```

The system displays a message similar to the following message:

```
Storage Subsystem Name = 'Twister'
hdisk#           LUN #   Ownership           Adapter  User Label
hdisk11              0   A (preferred)        fscsi0  1_disk_0
hdisk12              1   A (preferred)      Inactive  1_disk_1
hdisk13              2   A (preferred)      Inactive  1_disk_2
hdisk14              3   A (preferred)      Inactive  1_disk_3
```

3. To display information about all the attached subsystems, enter the following command:

```
mpio_get_config -A
```

The system displays a message similar to the following message:

```
Storage Subsystem worldwide name: 60ab80026982e000045f255d7
Storage Subsystem Name = 'Twister'

hdisk#           LUN #   Ownership          User Label
hdisk11              0   A (preferred)      1_disk_0
hdisk12              1   A (preferred)      1_disk_1
hdisk13              2   A (preferred)      1_disk_2
hdisk14              3   A (preferred)      1_disk_3
```

4. To display additional information about all the attached subsystems, enter the following command:

```
mpio_get_config -A -v
```

The system displays a message similar to the following message:

```
Frame id 0:
Storage Subsystem worldwide name: 60ab80026982e000045f255d7
Controller count: 2
Partition count: 1
Partition 0:
Storage Subsystem Name = 'Twister'
hdisk#           LUN #   Ownership          User Label
hdisk11              0   A (preferred)      1_disk_0
hdisk12              1   A (preferred)      1_disk_1
hdisk13              2   A (preferred)      1_disk_2
hdisk14              3   A (preferred)      1_disk_3
```

mpstat Command

Purpose

Collects and displays performance statistics for all logical processors in the system.

Syntax

`mpstat` [{ `-d` | `-i` | `-s` | `-a` | `-h` | `-v` | `-E` }] [`-w`][**`-O`** *Options*] [`-@` *wparname*] [*interval* [*count*]]

mpstat [**-X** [-o *filename*]] [interval[count]]

Restriction: The *wparname* parameter is restricted to use inside workload partitions.

Description

The `mpstat` command collects and displays performance statistics for all logical processors in the system. Users can define both, the number of times the statistics are displayed, and the interval at which the data is updated.

The *interval* parameter specifies the amount of time in seconds between each report. If you do not specify the *interval* parameter, the **mpstat** command generates a single report that contains statistics for the time since system startup and then exits. You can specify the *count* parameter only with the *interval* parameter. If you specify the *count* parameter, its value determines the number of reports that are generated and the number of seconds apart. If you specify the *interval* parameter without the *count* parameter, reports are continuously generated. Do not specify a value of zero to the *count* parameter.

The **mpstat** command with no options generates a single report that contains the performance statistics for all logical processors since boot time.

When the `mpstat` command is invoked, it displays two sections of statistics. The first section displays the System Configuration, which is displayed when the command starts and whenever there is a change in the system configuration. The second section displays the Utilization Statistics which are displayed in intervals and at any time the values of these metrics are deltas from previous interval.

The following information is displayed in the system configuration section:

lcpu
Indicates the number of online logical processors.

ent
Indicates the entitled processing capacity in processor units. This information is displayed only when the partition type is shared.

mode
Indicates whether the partition processor capacity is capped or uncapped allowing it to consume idle cycles from the shared pool. Dedicated LPAR is capped or donating.

rset
Indicates the resource-set type (regular or exclusive) that is associated with the WPAR. This information is displayed only when there is a resource set that is associated with the WPAR.

The performance statistics displayed by `mpstat` are listed below:

CPU
(All flags) Logical processor ID.

Note: The logical processor ID that is associated with the resource set of a WPAR is prefixed by an asterisk (*) when you run the **mpstat** command inside a WPAR with the **-s** or **-@** flag.

min
(Default, `-a` flag) Minor page faults (page faults with no IO).

maj
(Default, `-a` flag) Major page faults (page faults with disk IO).

mpcs
(`-a`, `-i` flag) Number of `mpc` send interrupts.

mpcr
(`-a`, `-i` flag) Number of `mpc` receive interrupts.

mpc
(Only default) Total number of inter-processor calls .

dev
(`-a`, `-i` flag) Number of device interrupts.

soft
(`-a`, `-i` flag) Number of software interrupts.

dec
(-a, -i flag) Number of decrementer interrupts.

ph
(-a, -i flag) Number of phantom interrupts.

int
(Only default) Total number of interrupts.

cs
(Default, -a flag) Total number of context switches.

ics
(Default, -a flag) Total number of involuntary context switches.

bound
(-a, -d flag) Total number of threads that are bound.

rq
(Default, -a, -d flag) Run queue size.

push
(-a, -d flag) Number of migrations due to starvation load balancing .

S3pull
(-a, -d flag) Number of migrations outside the scheduling affinity domain 3 due to idle stealing.

S3grd
(-a, -d flag) Number of dispatches from global runqueue, outside the scheduling affinity domain 3.

mig
(Only default) Total number of thread migrations (to another logical processor).

S0rd
(**-a**, **-d** flag) The percentage of thread redispatches within the same logical processor with scheduling affinity domain 0.

S1rd
(**-a**, **-d** flag) The percentage of thread redispatches within the same physical processor or core with scheduling affinity domain 1.

S2rd
(-a, -d flag) The percentage of thread redispatches within the same chip set, but not within the same processor core with scheduling affinity domain 2.

S3rd
(-a, -d flag) The percentage of thread redispatches within the same MCM (multiple chip module) , but not within the same chip set with scheduling affinity domain 3.

S4rd
(-a, -d flag) The percentage of thread redispatches on different MCMs within the same CEC or Plane with scheduling affinity domain 4.

S5rd
(-a, -d flag) The percentage of thread redispatches on a different CEC or Plane with scheduling affinity domain 5.

S3hrd
(-a, -d flag) The percentage of local thread dispatches on this logical processor.

S4hrd
(-a, -d flag) The percentage of near thread dispatches on this logical processor.

S5hrd
(-a, -d flag) The percentage of far thread dispatches on this logical processor.

lpa
(Only default) Logical processor affinity. The percentage of logical processor re-dispatches within the scheduling affinity domain 3.

sysc
(Default, -a flag) Number of system calls.

us
(Default, -a flag, -v flag) The percentage of physical processor utilization that occurred while executing at the user level (application).

If the -v flag is used, then utilization is based on the virtual processor.

sy
(Default, -a flag, -v flag) The percentage of physical processor utilization that occurred while executing at the system level (kernel).

If the -v flag is used, then utilization is based on the virtual processor.

wa
(Default, -a flag, -v flag) The percentage of time that the logical processor was idle during which it had an outstanding disk I/O request.

If the -v flag is used, then utilization is based on the virtual processor.

id
(Default, -a flag, -v flag) The percentage of time that the logical processor was idle and it did not have an outstanding disk I/O request.

If the -v flag is used, then utilization is based on the virtual processor.

pc
(Default, -a flag, -h flag, -v flag) The number or fraction of physical processor consumed. It is displayed in both a shared partition and a dedicated partition. For the default flag in the dedicated partition, it is not displayed when both donation and simultaneous multithreading are disabled.

The pc of the cpuid U row represents the number of unused physical processors.

%ec
(Default, -a flag) The percentage of entitled capacity consumed by the logical processor. The %ec of the ALL CPU row represents the percentage of entitled capacity consumed. Because the time base over which this data is computed can vary, the entitled capacity percentage can sometimes exceed 100%. This excess is noticeable only with small sampling intervals. The attribute is displayed only in a shared partition.

ilcs
(-a, -d, -h flag) Number of involuntary logical processor context switches, displayed only in shared partition. For the -h and -a flags, it is also displayed in dedicated partition.

vlcs
(-a, -d, -h flag) Number of voluntary logical processor context switches. Displayed only in shared partition. For the -h and -a flags, it is also displayed in dedicated partition.

lcs
(Default) Total number of logical processor context switches. Displayed only in shared partition or when a dedicated partition is donating.

%idon
(-a, -h flag) Shows the percentage of physical processor utilization that occurs while explicitly donating idle cycles. Displayed only in dedicated partition that is donating.

%bdon
(-a, -h flag) Shows the percentage of physical processor utilization that occurs while donating busy cycles. Displayed only in dedicated partition that is donating.

%istol
(-a, -h flag) Shows the percentage of physical processor utilization that occurs while the Hypervisor is stealing idle cycles. Displayed only in dedicated partition.

%bstol
(-a, -h flag) Shows the percentage of physical processor utilization that occurs while the Hypervisor is stealing busy cycles. Displayed only in dedicated partition.

%nsp
(-a, -d flag) Shows the current average processor speed as a percentage of nominal speed. Displayed only if the hardware uses Scaled Processor Utilization Resource Register (SPURR).

vcpu
(-v flag) Shows the virtual processor id.

lcpu
(-v flag) Shows the logical processor id.

pbusy
(-v, -E flag) Shows the percentage of physical time during which the physical processor is busy.

VTB
(-v flag) Shows the time taken for a virtual processor in milliseconds.

physc
(-E flag) Shows the number of physical processors that are used by the logical processor.

freq
(-E flag) Shows the operating frequency in GHz.

scaled physc
(-E flag) Shows the number of physical processors that are used by the logical processor based on SPURR.

The `mpstat` command displays all the above statistics for every logical processor in the partition. When running a shared partition, a special processor row with the `cpuid U` can be displayed when the entitled processing capacity has not entirely been consumed.

The `mpstat` command also displays a special processor row with the `cpuid ALL` which shows the partition-wide utilization. On that row, except for uncapped partitions with current physical processor consumption above their entitled capacity, the percentages are relative to the entitled processing capacity. For dedicated partitions, the entitled processing capacity is the number of physical processors. And for a WPAR , the processors present in the associated resource set, if any, are indicated by asterisks (*) only when the **-s** or **-@** flag is used.

When the `-s` flag is specified, the `mpstat` command reports simultaneous multithreading utilization, if it is enabled. This report displays the virtual processor engines utilization and utilization of each thread (logical processor) associated with the virtual processor engine.

If `mpstat` is running in a dedicated partition and simultaneous multithreading is enabled, then only the thread (logical processor) utilization is displayed.

If `mpstat` is running on an interval based mode, then it would be average value calculated per second.

Flags

Item	Description
`-a`	Displays all the statistics.
`-d`	Displays detailed affinity and migration statistics for AIX threads and dispatching statistics for logical processors.
`-i`	Displays detailed interrupts statistics.
`-s`	Displays simultaneous multithreading threads utilization, this flag is available only when `mpstat` runs in a simultaneous multithreading enabled partition.
`-h`	Displays pc and processor switches, with stolen and donation statistics for dedicated partitions.
`-w`	Displays wide column output, switches to wide output mode. Default is 80 column output mode.

Item	Description
@ *wparname*	Displays the statistics for the specified WPAR.
-O *Options*	Specifies the command option. **-O options=value...** Following are the supported options: • **sortcolumn** = Name of the metrics in the **mpstat** command output • **sortorder** = [asc|desc] • **topcount** = Number of CPUs to be displayed in the **mpstat** command sorted output
-X	Generates the XML output. The default file name is **mpstat_DDMMYYHHMM.xml** unless you specify a different file name by using with the **–o** option.
-o	Specifies the file name for the XML output.
`-v`	Displays utilization statistics at the virtual processor level. **Note:** The **-v** flag is available only for POWER8 processors, and later.
`-E`	Displays SPURR-based utilization metrics on a SPURR-capable processor.

Note:

1. The `-a`, `-d`, and `-i` flags implicitly turn on wide-column output.
2. Inside a WPAR, the **-@** flag reports statistics of all processors.
3. Processor statistics that are displayed inside a WPAR is always system wide.
4. Only **-o** option is allowed with **-X** option.

Parameters

Item	Description
interval	Specifies the interval between the iterations. If *interval* is not specified, just one snapshot of metrics is displayed which actually reports the values from the time system is up. If *interval* is specified, the tool waits for that duration before printing the first set of data. Each set of data is followed by a separation line, a line with average values for each columns (except the processor, which is replaced by ALL), followed by an empty line.
count	Specifies number of iterations. If *interval* is specified and *count* is not specified then `mpstat` runs infinitely. *count* can not be specified without specifying *interval*.

Examples

1. To see the default set of utilization metrics, enter the following command:

   ```
   mpstat 1 1
   ```

2. To see the default set of utilization metrics in wide display mode, enter the following command:

   ```
   mpstat –w 1 1
   ```

3. To see the detailed dispatch & affinity metrics, enter the following command:

   ```
    mpstat –d 1 1
   ```

4. To see the detailed interrupts report, enter the following command:

```
mpstat –i 1 1
```

5. To see all the statistics, enter the following command:

```
mpstat –a 1 1
```

6. To see simultaneous multithreading utilization, enter the following command:

```
mpstat –s 1 1
```

7. To see all the processor metrics of a WPAR, enter the following command:

```
mpstat -@ wparname
```

Note: To see all the processor metrics of a WPAR inside the WPAR, enter the following command:

```
mpstat -@
```

8. To see the sorted output for the column **cs**, enter the following command:

```
mpstat -d -O sortcolumn=cs
```

9. To see the list of the top 10 CPUs, enter the following command:

```
mpstat -a -O sortcolumn=min,sortorder=desc,topcount=10
```

10. To see metrics based on the virtual processor, enter the following command:

```
mpstat –v
```

Files

Item	Description
`/usr/bin/mpstat`	Contains the `mpstat` command.

mrouted Daemon

Purpose

Forwards a multicast datagram.

Syntax

/usr/sbin/mrouted [**-p**] [**-c** *Config_File*] [**-d** [*Debug_Level*]]

Description

The **mrouted** daemon is an implementation of the Distance Vector Multicast Routing Protocol (DVMRP), an earlier version of which is specified in RFC 1075. It maintains topological knowledge using a distance vector routing protocol (like RIP, described in RFC 1058), on which it implements a multicast datagram forwarding algorithm called Reverse Path Multicasting.

The **mrouted** daemon forwards a multicast datagram along a shortest (reverse) path tree rooted at the subnet on which the datagram originates. The multicast delivery tree may be thought of as a broadcast delivery tree that has been pruned back so that it does not extend beyond those subnetworks that have members of the destination group. Hence, datagrams are not forwarded along those branches that have no listeners of the multicast group. The IP time-to-live of a multicast datagram can be used to limit the range of multicast datagrams.

To support multicasting among subnets that are separated by (unicast) routers that do not support IP multicasting, the **mrouted** daemon includes support for tunnels, which are virtual point-to-point links between pairs of the **mrouted** daemons located anywhere in an Internet. IP multicast packets are encapsulated for transmission through tunnels, so that they look like typical unicast datagrams to intervening routers and subnets. The encapsulation is added on entry to a tunnel, and stripped off on exit from a tunnel. By default, the packets are encapsulated using the IP-in-IP protocol (IP protocol number 4). Older versions of the **mrouted** tunnel use IP source routing, which puts a heavy load on some types of routers. This version does not support IP source-route tunneling.

The tunneling mechanism allows the **mrouted** daemon to establish a virtual Internet, for the purpose of multicasting only, which is independent of the physical Internet and which may span multiple Autonomous Systems. This capability is intended for experimental support of Internet multicasting only, pending widespread support for multicast routing by the regular (unicast) routers. The **mrouted** daemon suffers from the well-known scaling problems of any distance-vector routing protocol and does not support hierarchical multicast routing.

The **mrouted** daemon automatically configures itself to forward on all multicast-capable interfaces (that is, interfaces that have the IFF_MULTICAST flag set, excluding the loopback interface), and it finds other **mrouted** daemons directly reachable using those interfaces.

The **mrouted** daemon does not initiate execution if it has fewer than two enabled virtual interfaces, where a virtual interface (Vif) is either a physical multicast-capable interface or a tunnel. It logs a warning if all of its virtual interfaces are tunnels; such an **mrouted** daemon's configuration would be better replaced by more direct tunnels.

The **mrouted** daemon handles multicast routing only; there might be unicast-routing software running on the same machine as the **mrouted** daemon. With the use of tunnels, it is unnecessary for the **mrouted** daemon to have access to more than one physical subnet to perform multicast forwarding.

Flags

Item	Description
-c *Config_File*	Starts the **mrouted** command using an alternate configuration file specified by the *Config_File* variable. There are five types of configuration entries: **phyint** local-addr **[disable]** [metric m] **[threshold** t] **[rate_limit** b] **[boundary (**boundary-name**|**scoped-addr**/**mask-len**)] [altnet** network**/**mask-len**]** **tunnel** *local-addr remote-addr* [**metric** *m*] [**threshold** *t*] [**rate_limit** *b*] [**boundary** (*boundary-name* | *scoped-addr* / *mask-len*)] **cache_lifetime** *ct* **pruning** *off* | *on* **name** *boundary-name scoped-addr* / *mask-len*
-d	Sets the debug level. If no **-d** option is given, or if the debug level is specified as 0, the **mrouted** daemon detaches from the invoking terminal. Otherwise, it remains attached to the invoking terminal and responsive to signals from that terminal. If **-d** is given with no argument, the debug level defaults to 2. Regardless of the debug level, the **mrouted** daemon always writes warning and error messages to the system log demon. Non-zero debug levels have the following effects: **level 1** All syslog'ed messages are also printed to **stderr**. **level 2** All level 1 messages plus notifications of significant events are printed to **stderr**. **level 3** All level 2 messages plus notifications of all packet arrivals and departures are printed to **stderr**. Upon startup, the **mrouted** daemon writes its pid to the file **/etc/mrouted.pid**.
-p	Turns off pruning. Default is pruning enabled.

Signals

The following signals can be sent to the **mrouted** daemon:

Item	Description
HUP	Restarts the **mrouted** daemon. The configuration file is reread every time this signal is evoked.
INT	Terminates execution gracefully (that is, by sending good-bye messages to all neighboring routers).
TERM	Same as **INT**.
USR1	Dumps the internal routing tables to **/usr/tmp/mrouted.dump**.
USR2	Dumps the internal cache tables to **/usr/tmp/mrouted.cache**.
QUIT	Dumps the internal routing tables to **stderr** (if the **mrouted** daemon was invoked with a nonzero debug level).

For convenience in sending signals, the **mrouted** daemon writes its pid to **/etc/mrouted.pid** on startup.

Examples

1. To display routing table information, type:

```
kill -USR1 *cat /etc/mrouted.pid*
```

This produces the following output:

```
Virtual Interface Table
 Vif Local-Address                    Metric   Thresh   Flags
  0  36.2.0.8     subnet: 36.2           1        1     querier
                  groups: 224.0.2.1
                          224.0.0.4
                pkts in: 3456
               pkts out: 2322323

  1  36.11.0.1    subnet: 36.11          1        1     querier
                  groups: 224.0.2.1
                          224.0.1.0
                          224.0.0.4
                pkts in: 345
               pkts out: 3456

  2  36.2.0.8      tunnel: 36.8.0.77       3        1
                  peers: 36.8.0.77 (2.2)
             boundaries: 239.0.1
                       : 239.1.2
                pkts in: 34545433
               pkts out: 234342

  3  36.2.0.8     tunnel: 36.6.8.23       3        16

Multicast Routing Table (1136 entries)
 Origin-Subnet    From-Gateway         Metric Tmr In-Vif Out-Vifs
 36.2                                 1    45  0     1* 2 3*
 36.8             36.8.0.77            4    15  2     0* 1* 3*
 36.11                                1    20  1     0* 2 3*
 .
 .
 .
```

In this example, there are four virtual interfaces connecting to two subnets and two tunnels. The Vif 3 tunnel is not in use (no peer address). The Vif 0 and Vif 1 subnets have some groups present; tunnels never have any groups. This instance of the **mrouted** daemon is the one responsible for sending periodic group membership queries on the Vif 0 and Vif 1 subnets, as indicated by the **querier** flags. The list of boundaries indicate the scoped addresses on that interface. A count of the no. of incoming and outgoing packets is also shown at each interface.

Associated with each subnet from which a multicast datagram can originate is the address of the previous hop router (unless the subnet is directly connected), the metric of the path back to the origin,

the amount of time since an update for this subnet was last received, the incoming virtual interface for multicasts from that origin, and a list of outgoing virtual interfaces. The * (asterisk) means that the outgoing virtual interface is connected to a leaf of the broadcast tree rooted at the origin, and a multicast datagram from that origin will be forwarded on that outgoing virtual interface only if there are members of the destination group on that leaf.

The **mrouted** daemon also maintains a copy of the kernel forwarding cache table. Entries are created and deleted by the **mrouted** daemon.

2. To display cache table information, type:

```
kill -USR2 *cat /etc/mrouted.pid*
```

This produces the following output:

```
Multicast Routing Cache Table (147 entries)
 Origin          Mcast-group      CTmr   Age  Ptmr  IVif  Forwvifs
 13.2.116/22     224.2.127.255      3m    2m    -    0     1
 >13.2.116.19
 >13.2.116.196
 138.96.48/21    224.2.127.255      5m    2m    -    0     1
 >138.96.48.108
 128.9.160/20    224.2.127.255      3m    2m    -    0     1
 >128.9.160.45
 198.106.194/24 224.2.135.190       9m    28s   9m    0P
 >198.106.194.22
```

Each entry is characterized by the origin subnet number and mask and the destination multicast group. The **CTmr** field indicates the lifetime of the entry. The entry is deleted from the cache table when the timer decrements to zero. The Age field is the time since this cache entry was originally created. Because cache entries get refreshed if traffic is flowing, routing entries can grow very old. The **Ptmr** field is a hyphen if no prune was sent upstream or the amount of time until the upstream prune will time out. The **Ivif** field indicates the incoming virtual interface for multicast packets from that origin. Each router also maintains a record of the number of prunes received from neighboring routers for a particular source and group. If there are no members of a multicast group on any downward link of the multicast tree for a subnet, a prune message is sent to the upstream router. They are indicated by a P after the virtual interface number. The **Forwvifs** field shows the interfaces along which datagrams belonging to the source group are forwarded. A p indicates that no datagrams are being forwarded along that interface. An unlisted interface is a leaf subnet with are no members of the particular group on that subnet. A b on an interface indicates that it is a boundary interface, that is, traffic will not be forwarded on the scoped address on that interface. An additional line with a > (greater-than sign) as the first character is printed for each source on the subnet. There can be many sources in one subnet.

Files

Item	Description
/etc/mrouted.conf	Contains the configuration information for the **mrouted** daemon.
/usr/tmp/mrouted.dump	Contains the internal routing tables for the **mrouted** daemon.
/etc/mrouted.pid	Contains the process ID for the **mrouted** daemon.
/usr/tmp/mrouted.cache	Contains the internal cache tables for the **mrouted** daemon.

msgchk Command

Purpose

Checks for messages.

Syntax

msgchk [*User* ...]

Description

The **msgchk** command checks mail drops for messages. The **msgchk** command reports whether the mail drop for the specified user contains messages and indicates if the user has already seen these messages. By default, the **msgchk** command checks the mail drop for the current user.

Flags

Item	Description
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For the Message Handler (MH), the name of this flag must be fully spelled out.

Examples

1. To check to see if you have any new messages, enter:

   ```
   msgchk
   ```

   If you have new messages, the system responds with a message similar to the following:

   ```
   You have new Internet mail waiting
   ```

   If you have no messages, the system responds with a message similar to the following:

   ```
   You don't have any mail waiting
   ```

2. To check to see if user `karen` on your local system has any new messages, enter:

   ```
   msgchk karen
   ```

   In this example, if user `karen` on your local system has new messages, the system responds with a message similar to the following:

   ```
   karen has new Internet mail waiting
   ```

   If user `karen` on your local system has no messages, the system responds with a message similar to the following:

   ```
   karen doesn't have any mail waiting
   ```

Files

Item	Description
$HOME/.mh_profile	Contains the user's MH profile.
/etc/mh/mtstailor	Contains the MH tailor file.
/var/spool/Mail/$USER	Defines the location of the mail drop.
/usr/bin/msgchk	Contains the **msgchk** command.

msh Command

Purpose

Creates a Message Handler (MH) shell.

Syntax

msh [*File*] [**-prompt** *String*] [**-notopcur** | **-topcur**]

Description

The **msh** command creates an MH shell for use with messages that are packed in a file. By default, this command looks for the **msgbox** file in the current directory. Within the MH shell, you can use the following MH commands:

```
ali             burst           comp        dist
folder          forw            inc         mark
mhmail          msgchk          next        packf
pick            prev            refile      repl
rmm             scan            send        show
sortm           whatnow         whom
```

These commands operate with limited functionality in the MH shell. To see how a command operates in the MH shell, enter the command name followed by the **-help** flag. Entering `help` or a `?` (question mark) displays a list of the MH commands you can use.

To leave the **msh** shell, press the Ctrl-D key sequence or enter `quit`.

Flags

Item	Description
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-notopcur	Makes the current message track the center line of the **vmh** scan window when the **msh** command is started using the **vmh** command. This flag is the default.
-prompt *String*	Prompts for the **msh** commands with the specified string.
-topcur	Makes the current message track the top line of the **vmh** scan window when the **msh** command is started using the **vmh** command.

Profile Entries

The following entries are found in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`fileproc:`	Specifies the program used to refile messages.
`Msg-Protect:`	Sets the protection level for your new message files.
`Path:`	Specifies the user's MH directory.
`showproc:`	Specifies the program used to show messages.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To start an **msh** shell, enter:

   ```
   msh
   ```

   If the **msgbox** file exists in the current directory, the system responds with a message similar to the following:

   ```
   Reading ./msgbox, currently at message 1 of 10
   ```

   Then, the system prompt appears as follows:

   ```
   (msh)
   ```

   In this example, the current message is message 1 in the `msgbox` file. You can now enter a modified subset of MH commands.
2. To start an **msh** shell to manipulate the messages stored in the `meetings` file, enter:

   ```
   msh meetings
   ```

Files

Item	Description
$HOME/.mh_profile	Specifies the user's MH profile.
/etc/mh/mtstailor	Contains the MH tailor file.
/usr/bin/msh	Contains the **msh** command.

mt Command (BSD)

Purpose

Gives subcommands to streaming tape device.

Syntax

mt [**-f** *TapeName*] *Subcommand* [*Count*]

Description

The **mt** command gives subcommands to a streaming tape device. If you do not specify the **-f** flag with the *TapeName* parameter, the **TAPE** environment variable is used. If the environment variable does not exist, the **mt** command uses the **/dev/rmt0.1** device. The *TapeName* parameter must be a raw (not block) tape device. You can specify more than one operation with the *Count* parameter.

Subcommands

Item	Description
eof, weof	Writes the number of end-of-file markers specified by the *Count* parameter at the current position on the tape.
fsf	Moves the tape forward the number of files specified by the *Count* parameter and positions it to the beginning of the next file.

Item	Description
bsf	Moves the tape backwards the number of files specified by the *Count* parameter and positions it to the beginning of the last file skipped. If using the bsf subcommand would cause the tape head to move back past the beginning of the tape, then the tape will be rewound, and the **mt** command will return **EIO**.
fsr	Moves the tape forward the number of records specified by the *Count* parameter.
bsr	Moves the tape backwards the number of records specified by the *Count* parameter.
rewoff1, rewind	Rewinds the tape. The *Count* parameter is ignored.
status	Prints status information about the specified tape device. The output of the status command may change in future implementations.

Flag

Item	Description
-f *TapeName*	Specifies the *TapeName* parameter.

Examples

1. To rewind the `rmt1` tape device, enter:

```
mt -f /dev/rmt1 rewind
```

2. To move forward two files on the default tape device, enter:

```
mt fsf 2
```

3. To write two end-of-file markers on the tape in the `/dev/rmt0.6` file, enter:

```
mt -f /dev/rmt0.6 weof 2
```

Exit Status

Item	Description
0	Indicates a successful completion.
>0	Indicates an error occurred.

Files

Item	Description
/dev/rmt/*n***.***n*	Specifies the raw streaming tape interface.
/usr/bin/mt	Contains the **mt** command file.

mtrace Command

Purpose

Prints a multicast path from a source to a receiver.

Syntax

mtrace [**-l**] [**-M**] [**-n**] [**-p**] [**-s**] [**-U**] [**-g** *gateway*] [**-i** *if_addr*] [**-m** *max_hops*] [**-q** *nqueries*] [**-r** *resp_dest*] [**-S** *statint*] [**-t** *ttl*] [**-w** *wait*] *source* [*receiver*] [*group*]

Description

A trace query is passed hop-by-hop along the path from the receiver to the source, collecting hop addresses, packet counts, and routing error conditions along the path, and then the response is returned to the requestor. The default receiver is the host running the **mtrace** command, and the default group is 0.0.0.0.

Note: The **mtrace** command is intended for use in network testing, measurement, and management. Because the **mtrace** command heavily loads on the network, avoid using the **mtrace** command during typical operations or from automated scripts. It should be used primarily or with manual fault isolation. If the **-g** flag is specified, the source defaults to the host running **mtrace** and the receiver defaults to the router being addressed.

By default, the **mtrace** command first attempts to trace the full reverse path, unless the number of hops to trace is explicitly set with the **-m** flag. If there is no response within a 3-second timeout interval (changed with the **-w** flag), an * (asterisk) is printed and the probing switches to hop-by-hop mode. Trace queries are issued starting with a maximum hop count of one and increasing by one until the full path is traced or no response is received. At each hop, multiple probes are sent (default is three, changed with **-q** flag). The first half of the attempts (default is two) are made with the reply address set to standard multicast address, mtrace.mcast.net (224.0.1.32) with the *ttl* set to 32 more than what is needed to pass the thresholds seen so far along the path to the receiver. For each additional attempt, the *ttl* is increased by another 32 each time up to a maximum of 192. Because the desired router may not be able to send a multicast reply, the remainder of the attempts request that the response be sent via unicast to the host running the **mtrace** command.

Alternatively, the multicast *ttl* can be set explicitly with the **-t** flag, the initial multicast attempts can be forced to use unicast instead with the **-U** flag, the final unicast attempts can be forced to use multicast instead with the **-M** flag, or if you specify **-UM**, the **mtrace** command will first attempt using unicast and then multicast. For each attempt, if no response is received within the timeout, an * (asterisk) is printed. After the specified number of attempts have failed, the **mtrace** command will try to query the next hop router with a **DVMRP_ASK_NEIGHBORS2** request to see what kind of router it is. The **mtrace** command will try to query three (changed with the **-e** flag) hops past a non-responding router. Even though the **mtrace** command is incapable of sending a response, it might be capable of forwarding the request.

Flags

Item	Description
-g *gateway*	Sends the trace query via unicast directly to the multicast router *gateway* rather than multicasting the query. This must be the last-hop router on the path from the intended source to the receiver.
-i *if_addr*	Uses *if_addr* as the local interface address (on a multi-homed host) for sending the trace query and as the default for the receiver and the response destination.
-l	Loops indefinitely printing packet rate and loss statistics for the multicast path every 10 seconds (see -S *stat_int*).
-m *max_hops*	Sets *max_hops* to the maximum number of hops that will be traced from the receiver to the source. The default is 32 hops and infinity for the DVMRP routing protocol).

Item	Description
-M	Always requests the response using multicast rather than attempting unicast for the last half of the tries.
-n	Prints hop addresses numerically rather than symbolically and numerically (saves a name server address-to-name lookup for each router found on the path).
-p	Listens passively for multicast responses from traces initiated by others. This works best when run on a multicast router.
-q *nqueries*	Sets the maximum number of query attempts for any hop to *nqueries*. The default is 3.
-r *resp_dest*	Sends the trace response to dhost rather than to the host on which the **mtrace** command is being run, or to a multicast address other than the one registered for this purpose (224.0.1.32).
-s	Prints a short form output including only the multicast path and not the packet rate and loss statistics.
-S *statint*	Changes the interval between statistics gathering traces to *statint* seconds (default 10 seconds).
-t *ttl*	Sets the *ttl* (time-to-live, or number of hops) for multicast trace queries and responses. The default is 127, except for local queries to the `all routers` multicast group that use the *ttl* of 1.
-U	Forces initial multicast attempts to use unicast instead.
-w *wait*	Sets the time to wait for a trace response to*wait* seconds (default 3 seconds).

Parameters

Item	Description
source	Specifies the host for which the multicast path from a particular receiver is sought. This is a required parameter.
receiver	Specifies the host from which the multicast path is sought for a particular source. Default is the host in which the **mtrace** command is running. This is an optional parameter.
group	Specifies the multicast group. This is an optional parameter.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

In the following example, the two machines, 10.27.41.57 and 10.27.40.20, are on two different subnets separated by a router having two interfaces, 10.27.40.11 and 10.27.41.11. To find the multicast path from 10.27.40.20 to 10.27.41.57, type the following:

```
# mtrace 10.27.41.57 224.2.0.1
```

The following is displayed:

```
Mtrace from 10.27.41.57 to 10.27.40.20 via group 224.2.0.1
Querying full reverse path...
  0  ? (10.27.40.20)
 -1  ? (10.27.40.11)  DVMRP  thresh^ 1
```

```
 -2  ? (10.27.41.57)
Round trip time 1 ms; total ttl of 2 required.

Waiting to accumulate statistics... Results after 10 seconds:

  Source        Response Dest    Overall     Packet Statistics For Traffic From
10.27.41.57     224.0.1.32       Packet      10.27.41.57 To 224.2.0.1
     v       __/  rtt    1 ms     Rate       Lost/Sent = Pct  Rate
10.27.41.11
10.27.40.11     ?
     v       \__   ttl    2         0 pps
10.27.40.20     10.27.40.20
  Receiver      Query Source
```

multibos Command

Purpose

Creates, updates, and manages multiple versions of the Base Operating System (BOS) on a `rootvg`.

Syntax

`multibos -s` [`-l` *device* {`-a` | `-f` *file* | `-b` *file* | **-x** *file*}] [`-e` *file*] [`-i` *file*] [`-L` *file*] [`-pntNX`]

`multibos -c -l` *device* {`-a` | `-f` *file* | `-b` *file* | **-x** *file*} [`-pnNX`]

`multibos -m` [`-pnX`]

`multibos -u` [`-pnX`]

`multibos -B` [`-ntX`]

`multibos -S` [`-nX`]

`multibos -R` [`-ptX`]

`multibos -C` [*VG name*]

`multibos` **-s** `-M` *file* [`-pntNX`]

Description

The `multibos` command allows the root user to create an instance of the Base Operating System (BOS) in the `rootvg` volume group.

Note: You cannot use the `multibos` command with a mirrored `rootvg` volume group.

The `multibos` setup operation creates a standby Base Operating System (BOS) that boots from a distinct boot logical volume (BLV). This creates two bootable sets of BOS on a given `rootvg`. The administrator can boot from either instance of BOS by specifying the respective BLV as an argument to the `bootlist` command or using system firmware boot operations. Two bootable instances of BOS can be simultaneously maintained. The instance of BOS associated with the booted BLV is referred to as the *active* BOS. The instance of BOS associated with the BLV that has not been booted is referred to as the *standby* BOS. Currently, only two instances of BOS are supported per `rootvg`.

The `multibos` command allows the administrator to access, install maintenance and technology levels for, update, and customize the standby BOS either during setup or in subsequent customization operations. Installing maintenance and technology updates to the standby BOS does not change system files on the active BOS. This allows for concurrent update of the standby BOS, while the active BOS remains in production.

In addition, the `multibos` command copies or shares logical volumes and file systems. By default, the BOS file systems (currently `/`, `/usr`, `/var`, and `/opt`,) and the boot logical volume are copied. The administrator can make copies of additional BOS objects (using the `-L` flag).

All other file systems and logical volumes are shared between instances of BOS. Separate log device logical volumes (for example, those that are not contained within the file system) are not supported for copy and are shared.

In AIX 5L Version 5.3 with the 5300-09 Technology Level, you can populate the standby instance with a later version such as AIX Version 6.1 with the 6100-02 Technology Level. The function is done by creating a **mksysb** backup of a system at a later version and then using the backup to populate the standby instance. For example, system A is at a level of 5.3.9.0 and system B is at a level of 6.1.2.0. You can create a backup of system B using the `mksysb -M` command and use the **mksysb** backup to populate a standby instance of the operating system on system A.

Notes:

1. While the **multibos** command is running, ensure that system activity is minimal.
2. Any logical volume or file system attributes that are new to the higher level are not implemented when the standby instance is created because the operating system at a lower level has no knowledge of the attributes.
3. Do not keep both an AIX 6.1 instance and an AIX 5.3 instance for an extended period. You might not be able to switch between the instances because of incompatibilities. Commit to one of the instances and remove the other.
4. If your operating system is running with the logical volumes in the active BOS that have the multibos-created `bos_hd*` names, and no standby BOS in the `rootvg` directory, then a preservation or migration type of installation can occur starting with AIX 7200-00. The logical volumes that have the `bos_hd*` names on the system are `bos_hd5`, `bos_hd4`, `bos_hd2`, `bos_hd9var`, and `bos_hd10opt`. The operating system must not have `hd5`, `hd4`, `hd2`, `hd9var`, or `hd10opt` logical volumes. If you created the multibos instance from an `mksysb` image, which was created by using the **mksysb** command with the **-M** flag, the `hd8` logical volume might also have been renamed to `bos_hd8`. You can check this prerequisite with the **lsvg -l rootvg** command. Always back up your system before migrating. Also, copy the `/usr/lpp/bos/pre_migration` file from the media or your network installation manager (NIM) spot of the level to which you are migrating, to the target system and execute the file on the target system to check for any migration warnings.

   Before you perform a migration or a preservation type of operating system installation in this environment, verify that the disk control block has a valid level for your `rootvg`. You can run the **/usr/lpp/bosinst/blvset -d /dev/hdiskN -g level** command, where `hdiskN` is the disk that contains the `bos_hd5` logical volume. If this command returns `0.0` or an unexpected level, run the **bosboot -ad /dev/ipldevice** command to correct it, and rerun the **blvset** command to verify. It must return `6.1` or `7.1`.
5. In addition to the flags mentioned in the syntax section, the **-V** flag performs the verify operation from the inittab during boot. It is important that you do not modify this entry. The verify operation enables the **multibos** utility to synchronize changes in logical volumes and file systems between the active and standby instances. This entry also synchronizes the ODM and devices on initial boot after a **mksysb** restore. Without this operation, both the active and standby instances could become inconsistent with normal file system and logical volume operations.

The file system types (JFS or JFS2) of the **mksysb** backup need to be the same as that of the system where the **multibos** command is to be run. For example, if the **/usr** file system is a JFS2 file system, the **/usr** file system on the **mksysb** backup needs to be a JFS2 file system.

A log is stored in the **/etc/multibos/logs/op.alog** file after you run the **multibos** command. You can view the log file using the **alog -f /etc/multibos/logs/op.alog -o** command.

Note: You can create a backup that contains both instances by first mounting the standby instance (using the `-m` flag), and then creating the backup. However, you can restore the backup onto a disk only by using the **alt_disk_mksysb** command.

Restrictions

- The `multibos` command is supported on AIX 5L Version 5.3 with the 5300-03 Recommended Maintenance package and later.

- The current `rootvg` must have enough space for each BOS object copy. BOS object copies are placed on the same disk or disks as the original.
- The total number of copied logical volumes cannot exceed 128. The total number of copied logical volumes and shared logical volumes are subject to volume group limits.
- Using the **multibos** command in a mirrored `rootvg` volume group is not supported. You can remove the mirror of the `rootvg` volume group by running the **unmirrorvg** command. You can run the **alt_disk_copy** command to have a copy of the `rootvg` volume group that can be upgraded as needed. When you have the required instance of AIX, remove the volume group no longer needed by using the `alt_rootvg_op` operation, and re-mirror the `rootvg` volume group.

Flags

Item	Description
`-a`	Specifies the `update_all` install option. Valid with setup and customization operation.
`-B`	Build boot image operation. The standby boot image is created and written to the standby BLV using the AIX `bosboot` command.
`-b` *file*	Specifies the install bundle to be installed during the setup or customization operation. The install bundle syntax should follow `geninstall` conventions.
`-c`	Performs a customized update of the software in standby BOS.
`-C` *VG name*	Allows you to vary on volume groups for syncing when it is appropriate, as the auto varied off volume groups are not varied on during the reboot, like the rootvg volume group, in order to be synced. The `multibos -C` *VG name* command should be used on auto varied off volume groups after the `multibos` command has created an alternative root volume group (operating system) on the disk and has been booted.
`-e` *file*	Lists active BOS files to be excluded during the setup operation in regular expression syntax.
`-f` *file*	Lists fixes (such as APARs) that are to be installed during the setup or customization operation. The syntax of the list follows `instfix` conventions.
`-i` *file*	Specifies optional `image.data` file to use instead of the default `image.data` file created from the current `rootvg`.
`-L` *file*	Specifies a file that contains a list of additional logical volumes to include in standby BOS.
`-l` *device*	Installs device or directory for software update during the setup or customization operation.
`-m`	Mounts standby BOS.
`-M` *file*	Specifies a file that contains a **mksysb** image. The **mksysb** image must have been created using the `mksysb -M` command beginning with AIX 6.1 with 6100-02.
`-N`	Skips boot image processing. This flag should only be used by experienced administrators that have a good understanding of the AIX boot process.

Item	Description
`-n`	Does not perform cleanup upon failure. This option is useful to retain `multibos` data after a failed operation.
`-p`	Performs a preview of the given operation. Valid with setup, remove, mount, unmount, and customization operations.
`-R`	Removes all standby BOS objects.
`-S`	Initiates an interactive shell with `chroot` access to the standby BOS file systems.
`-s`	Creates an instance of standby BOS.
`-t`	Prevents `multibos` from changing the bootlist.
`-u`	Unmounts standby BOS.
-x *file*	Runs the optional customization script before any other customized parameters, such as, update_all (**-a**), install bundle file (**-b**), and fix list file (**-f**). You must use a full pathname for the script.
`-X`	Allows for automatic file system expansion if space is needed to perform tasks related to `multibos`. It is recommended that all `multibos` operations are executed with this flag.

Exit Status

Item	Description
`0`	All the `multibos` command operations completed successfully.
`>0`	An error occurred.

Security

Only the root user can run the `multibos` command.

Examples

1. To perform a standby BOS setup operation preview, enter the following command:

   ```
   multibos -Xsp
   ```

2. To set up standby BOS, enter the following command:

   ```
   multibos -Xs
   ```

3. To set up standby BOS with optional `image.data` file `/tmp/image.data` and exclude list `/tmp/exclude.list`, enter the following command:

   ```
   multibos -Xs -i /tmp/image.data -e /tmp/exclude.list
   ```

4. To set up standby BOS and install additional software listed as bundle file `/tmp/bundle` and located in the images source `/images`, enter the following command:

   ```
   multibos -Xs -b /tmp/bundle -l /images
   ```

5. To execute a customization operation on standby BOS with the `update_all` install option, enter the following command:

```
multibos -Xac -l /images
```

6. To mount all standby BOS file systems, enter the following command:

```
multibos -Xm
```

7. To perform a standby BOS remove operation preview, enter the following command:

```
multibos -RXp
```

8. To remove standby BOS, enter the following command:

```
multibos -RX
```

9. To use an existing **mksysb** file **/backups/mksysb1** to populate the standby instance of rootvg, enter the following command:

```
multibos -M /backups/mksysb1 -sX
```

Files

Item	Description
`/usr/sbin/multibos`	Contains the `multibos` command.
`/etc/multibos`	Contains `multibos` data and logs.

mv Command

Purpose

Moves files.

Syntax

To Move and Rename a File

mv [**-d**] [**-e**] [**-E**{**force**|**ignore**|**warn**}] [**-i** | **-f**] [**-I**] *SourceFile ... TargetFile*

To Move and Rename a Directory

mv [**-d**] [**-e**] **-E**{**force**|**ignore**|**warn**}] [**-i** | **-f**] [**-I**] *SourceDirectory ... TargetDirectory*

To Move Files or Directories to a Directory Maintaining Original File Names

mv [**-d**] [**-e**] **-E**{**force**|**ignore**|**warn**}] [**-i** | **-f**] [**-I**] *SourceFile/SourceDirectory TargetDirectory*

Description

Attention: The **mv** command can overwrite many existing files unless you specify the **-i** flag. The **-i** flag prompts you to confirm before it overwrites a file. If both the **-f** and **-i** flags are specified in combination, the last flag specified takes precedence.

The **mv** command moves files and directories from one directory to another or renames a file or directory. If you move a file or directory to a new directory, it retains the base file name. When you move a file, all links to other files remain intact, except when you move it to a different file system. When you move a directory into an existing directory, the directory and its contents are added under the existing directory.

When you use the **mv** command to rename a file or directory, the *TargetDirectory* parameter can specify either a new file name or a new directory path name.

If moving the file would overwrite an existing file that does not have write-permission set and if standard input is a workstation, the **mv** command displays the file-permission code and reads a line from standard input. If that line begins with a y or the locale's equivalent of a y, the **mv** command moves the file. If the response is anything other than a y, the **mv** command does nothing to that file and continues with the next specified file. The file-permission code displayed may not fully represent the access permission if the *TargetFile* is associated with an ACL. When the parent directory of the *SourceFile* is writable and has the sticky bit set, one or more of the following conditions are true:

- The user must own the file.
- The user must own the directory
- The user must be a privileged user.
- The file must be writable by the user.

This warning message and prompt for input can be overridden by using the `-f` option.

You can use the **mv** command to move files within the same file system or between file systems. Whether you are working in one file system or across file systems, the **mv** command copies the file to the target and deletes the original file. The **mv** command preserves in the new file the time of the most recent data modification, the time of the most recent access, the user ID, the group ID, the file mode, the extended attributes, and ACLs of the original file. For symbolic links, the **mv** command preserves only the owner and group of the link itself.

If it is unable to preserve the owner and group ID, the **mv** command clears S_ISUID and S_ISGID bits in the target. The **mv** command prints a diagnostic message to stderr if it is unable to clear these bits, though the exit code is not affected.

The **mv** command modifies either the source file or the destination path if the command is prematurely terminated.

Note: The **mv** command supports the **--** (dash, dash) parameter as a delimiter that indicates the end of the flags.

The `mv` command will not move an object if the object is exported as an NFS version 4 referral. The referral object is marked as busy and remains so until it is unexported.

Note: The I/O buffer size for the read and write system calls generated by this command can be configured by using the *AIX_STDBUFSZ* environment variable.

Flags

> **Attention:** The **mv** command can overwrite many existing files unless you specify the **-i** flag. The **-i** flag prompts you to confirm before it overwrites a file. If both the **-f** and **-i** flags are specified in combination, the last flag specified takes precedence.

Item	Description
-d	The source file is stored in decrypted (clear-text) format on target.
-e	The source file is stored in encrypted form, if the target file system is an Encrypted File System (EFS).

Item	Description
-E	The **-E** option requires one of the following arguments. If you omit the **-E** option, **warn** is the default behavior. **force** Fails the **mv** operation on a file if the fixed extent size or space reservation of the file cannot be preserved. **ignore** Ignores any errors in preserving extent attributes. **warn** Issues a warning if the space reservation or the fixed extent size of the file cannot be preserved.
-f	Does not prompt you before overwriting an existing file.
-i	Prompts you before moving a file or directory to an existing path name by displaying the name of the file followed by a question mark. If you answer with a line starting with y or the locale's equivalent of a y, the move continues. Any other reply prevents the move from occurring.
-I	Suppresses the warning message during ACL conversion.

The following table shows the encryption or decryption status of the target file under different conditions:

Explicit flag for the **mv** command	Source file	Target file system	Result
-e (encrypted)	Non-EFS	Non-EFS	Error
-e	Non-EFS	EFS	Encrypted file
-e	EFS	EFS	Encrypted file
-e	EFS	Non-EFS	Error
-d (decrypted)	Non-EFS	Non-EFS	Clear-text file
-d	Non-EFS	EFS	Clear-text file
-d	EFS	Non-EFS	Clear-text file
-d	EFS	EFS	Clear-text file
No explicit flag	Non-EFS	Non-EFS	Clear-text file
No explicit flag	Non-EFS	EFS	If the target file system is EFS activated, the target file is an encrypted file. Else, the target file is a clear-text file.
No explicit flag	EFS	EFS	Encrypted file
No explicit flag	EFS	Non-EFS	Error

Note: It is not permitted to overwrite an encrypted file with a plain-text file and vice versa unless you specify the **-f** flag. The encryption status of the target depends on the **-e** or **-d** flag, the encryption inheritance if you do not specify the **-e** or **-d** flag with the **-f** flag, and the encryption status of the source file if the encryption inheritance is not active.

Examples

1. To rename a file, enter:

   ```
   mv appendix apndx.a
   ```

   This command renames `appendix` to `apndx.a`. If a file named `apndx.a` already exists, its old contents are replaced with those of `appendix`.

2. To move a directory, enter:

   ```
   mv book manual
   ```

   This command moves all files and directories under `book` to the directory named `manual`, if `manual` exists. Otherwise, the directory `book` is renamed `manual`.

3. To move a file to another directory and give it a new name, enter:

   ```
   mv intro manual/chap1
   ```

   This command moves `intro` to `manual/chap1`. The name `intro` is removed from the current directory, and the same file appears as `chap1` in the directory `manual`.

4. To move a file to another directory, keeping the same name, enter:

   ```
   mv chap3 manual
   ```

   This command moves `chap3` to `manual/chap3`

   > **Note:** Examples 1 and 3 name two files, example 2 names two existing directories, and example 4 names a file and a directory.

5. To move several files into another directory, enter:

   ```
   mv chap4 jim/chap5 /home/manual
   ```

   This command moves the `chap4` file to the `/home/manual/chap4` file directory and the `jim/chap5` file to the `/home/manual/chap5` file.

6. To use the **mv** command with pattern-matching characters, enter:

   ```
   mv manual/* .
   ```

   This command moves all files in the `manual` directory into the current directory . (period), retaining the names they had in `manual`. This move also empties `manual`. You must type a space between the asterisk and the period.

   > **Note:** Pattern-matching characters expand names of existing files only. For example, the command `mv intro man*/chap1` does not work if the file `manual/chap1` does not exist.

Exit Status

Item	Description
0	All input files were moved successfully.
>0	An error occurred.

Files

Item	Description
/usr/bin/mv	Contains the **mv** command.

mvdir Command

Purpose

Moves (renames) a directory.

Syntax

mvdir *Directory1 Directory2*

Description

The **mvdir** command renames directories within a file system. To use the **mvdir** command, you must have write permission to *Directory1* and *Directory2* as well as in the parent directories.

The *Directory1* parameter must name an existing directory. If *Directory2* does not exist, *Directory1* is moved to *Directory2*. If *Directory2* exists, *Directory1* becomes a subdirectory of *Directory2*. Neither directory can be a subset of the other.

The **mvdir** Command can also be used to move or rename files. If the *Directory1* parameter is an existing file name and the *Directory2* parameter is an existing directory name, the file specified by *Directory1* is moved to the directory specified by *Directory2*. If the *Directory1* parameter is an existing file name and the *Directory2* parameter does not yet exist, *Directory2* replaces the file name *Directory1*. If both are existing file names, the file specified by *Directory1* is renamed *Directory2*, and the existing *Directory2* is removed.

The **mv** command provides the same functionality as the **mvdir** command.

The `mvdir` command will not rename a directory if the directory is exported for use by NFS version 4, or if the directory leads to a directory exported for use by NFS version 4. NFS version 4-exported directories and directories leading to NFS version 4-exported directories are marked as busy and remain so until unexported.

Example

To rename or move a directory to another location, enter:

```
mvdir appendixes manual
```

If `manual` does not exist, this renames the `appendixes` directory to `manual`.

If a directory named `manual` already exists, this moves `appendixes` and its contents to `manual/appendixes`. In other words, `appendixes` becomes a subdirectory of `manual`.

Files

Item	Description
/usr/sbin/mvdir	Contains the **mvdir** command.

mvfilt Command

Purpose

Moves a filter rule.

Syntax

mvfilt -v 4|6 -p *p_fid* **-n** *n_fid*

Description

Use the **mvfilt** command to change the position of a filter rule in the filter rule table. IPsec filter rules for this command can be configured using the `genfilt` command or IPsec smit (IP version 4 or IP version 6).

Flags

Item	Description
-v	IP version of the filter rule. The value **4** specifies IP version 4 and the value **6** specifies IP version 6.
-p	Filter rule ID. It specifies the previous position of the rule in the filter rule table. For IP version 4, the value of **1** is invalid since the first filter rule is unmoveable.
-n	Filter rule ID. It specifies the new position of the rule in the filter rule table after the move. For IP version 4, the value of **1** is invalid since the first filter rule is reserved and thus is unmoveable.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

mvt Command

Purpose

Typesets English-language view graphs and slides.

Syntax

mvt [**-a**] [**-c**] [**-e**] [**-g**] [**-p**] [**-t**] [**-z**] [**-T***Name* | **-D***Destination*] [*File* ... | **-**]

Description

The **mvt** command typesets its input with the **mv** macro package for view graphs and slides in a manner similar to the **mmt** command. The **mvt** command has flags to specify preprocessing by the **tbl**, **eqn**, **pic**, **cw**, and **grap** commands. The flags you select determine which pipelines, flags, and parameters are generated for the **troff** command and the macro package.

The **mvt** command, unlike the **troff** command, automatically pipes its output to a postprocessor, unless specifically requested not to do so. The user should not specify a postprocessor when using the **mvt** command. The path that the **mvt** command takes is as follows:

1. The **-z** flag (no postprocessor is used).
2. The **-T***Name* flag.
3. The **TYPESETTER** environment variable is read.
4. The default is set to **ibm3816**.

File specifies the file that the **mvt** command formats.

Flags

Flags can occur in any order, but they must be displayed before the *File* parameter. If no file is specified, the **mvt** command prints a list of its flags.

Item	Description
-a	Displays readable **troff** output to the terminal.
-c	Calls the **cw** command.
-e	Calls the **eqn** command; also causes the **eqn** command to read the **/usr/share/lib/pub/eqnchar** file.
-g	Calls the **grap** command, which in turn calls the **pic** command.
-p	Calls the **pic** command.
-t	Calls the **tbl** command.
-z	Calls no output filter (or postprocessor) to process or redirect the output of the **troff** command.
-D*Destination*	Directs the output to the specified device destination. Supported value for the *Destination* variable is **4014**, which is the Tektronix 4014 terminal by way of the **tc** command.
-T*Name*	Creates output for the **troff** device as specified by the *Name* variable. The output is sent through the appropriate postprocessor. The default is **ibm3816**.
-	Forces input to be read from standard input.

Any other parameters or flags that you give the **mvt** command (such as the **-a** flag) are passed to the **troff** command.

The **mvt** command reads standard input when you specify the **-** (minus) flag instead of the *File* parameter.

Use the **-o***List* flag of the **troff** command to specify ranges of pages to be output.

> **Note:** If you call the **mvt** command with one or more of the **-e**, **-c**, **-t**, **-p**, **-g**, or **-** flags, together with the **-o***List* flag of the **troff** command, you may receive a `broken pipe` message. This occurs if you do not specify the last page of the document in the *List* variable. This broken pipe message is not an indication of any problem and can be ignored.

Environment Variables

Item	Description
TYPESETTER	Contains information about a particular printing device.

Files

Item	Description
/usr/share/lib/pub/eqnchar	Contains special character definitions.

mwm Command

Purpose

Runs the AIXwindows Window Manager (MWM).

Syntax

mwm **-display** *Host:Display:ScreenID* **-xrm** *ResourceString* **-multiscreen** **-name** *Name* **-screens** *Name* [*Name* ...]

Description

The **mwm** command runs the AIXwindows Window Manager (MWM) and is often started by a display or session manager. The AIXwindows Window Manager (MWM) is an X Window System client that provides window management functionality and some session management functionality. It provides functions that facilitate control (by the user and the programmer) of elements of window states such as placement, size, icon or normal display, and input-focus ownership. It also provides session management functions such as stopping a client.

The appearance and behavior of the window manager can be altered by changing the configuration of specific resources. Resources are defined under X Defaults .

By default, the **mwm** command manages only the single screen specified by the **-display** option or the **DISPLAY** environment variable (by default, screen 0). If the **-multiscreen** option is specified or if the **multiScreen** resource is True, the **mwm** command tries to manage all the screens on the display.

When the **mwm** command is managing multiple screens, the **-screens** option can be used to give each screen a unique resource name. The names are separated by blanks, for example, **-screens mwm0 mwm1**. If there are more screens than names, resources for the remaining screens are retrieved using the first name. By default, the screen number is used for the screen name.

For information on windows, icons, resources, events, button and key bindings, menus, and variables, see the following sections:

- Windows
- Icons
- Icon Box
- Component Appearance Resources
- General Appearance and Behavior Resources
- Client-Specific Resources
- Window Manager Event Specification
- Button Bindings
- Key Bindings
- Menu Panes
- Environment

Flags

Flag Description

Item	Description
-display *Host*:*Display*:*ScreenID*	Specifies the display to use. The **-display** option has the following parameters: ***Host*** Specifies the host name of a valid system on the network. Depending on the situation, this could be the host name of the user or the host name of a remote system. ***Display*** Specifies the number (usually 0) of the display on the system on which the output is to be displayed. ***ScreenID*** Specifies the number of the screen where the output is to be displayed. This number is 0 for single-screen systems.
-xrm *ResourceString*	Enables the named resources when starting the **mwm** command.
-multiscreen	Causes the **mwm** command to manage all screens on the display. The default is to manage only a single screen.
-name *Name*	Causes the **mwm** command to retrieve its resources using the specified name, as in *Name*Resource*.
-screens *Name* [*Name* [...]]	Specifies the resource names to use for the screens managed by MWM. If MWM is managing a single screen, only the first name in the list is used. If multiple screens are being managed, the names are assigned to the screens in order, starting with screen 0. For example, screen 0 gets the first name and screen 1 gets the second name.

Windows

Default window manager window frames have the following distinct components with associated functions:

Windows Description

Item	Description
title area	In addition to displaying the client's title, the title area is used to move the window. To move the window, place the pointer over the title area, press button 1 and drag the window to a new location. A wire frame is moved during the drag to indicate the new location. When the button is released, the window is moved to the new location.
title bar	The title bar includes the title area, the **Minimize** button, the **Maximize** button, and the **Window Menu** button. In shaped windows, such as round windows, the title bar floats above the window.
Minimize button	To turn the window into an icon, click button 1 on the **Minimize** button (the frame box with a small square in it).
Maximize button	To make the window fill the screen (or enlarge to the largest size allowed by the configuration files), click button 1 on the **Maximize** button (the frame box with a large square in it).

Windows Description *(continued)*

Item	Description
Window Menu button	The **Window Menu** button is the frame box with a horizontal bar in it. To pull down the window menu, press button 1. While pressing the button, drag the pointer on the menu to your selection and release the button when your selection is highlighted. Pressing button 3 in the title bar or resize border handles also posts the window menu. Alternately, you can click button 1 to pull down the menu and keep it posted; then position the pointer and select. You can also post the window menu by pressing the Shift+Esc or Alt+Space key sequence. Double-clicking button 1 with the pointer on the **Window Menu** button closes the window. The following table lists the contents of the window menu:

Default Window Menu

Selection	Accelerator	Description
Restore	Alt+F5	Restores the window to its size before minimizing or maximizing.
Move	Alt+F7	Allows the window to be moved with keys or mouse.
Size	Alt+F8	Allows the window to be resized.
Minimize	Alt+F9	Turns the window into an icon.
Maximize	Alt+F10	Makes the window fill the screen.
Lower	Alt+F3	Moves window to bottom of window stack.
Close	Alt+F4	Causes client to stop.

resize border handles	To change the size of a window, move the pointer over a resize border handle (the cursor changes), press button 1, and drag the window to a new size. When the button is released, the window is resized. While dragging is being done, a rubber-band outline is displayed to indicate the new window size.
matte	An optional matte decoration can be added between the client area and the window frame. A matte is not actually part of the window frame. There is no functionality associated with a matte.

Icons

Icons are small graphic representations of windows. A window can be iconified (minimized) using the **Minimize** button on the window frame. Icons provide a way to reduce clutter on the screen.

Pressing the left mouse button when the pointer is over an icon causes the icon's window menu to open. Releasing the button (press + release without moving mouse = click) causes the menu to stay posted. The menu contains the following selections:

Icon Window Menu		
Selection	**Accelerator**	**Description**
Restore	Alt+F5	Opens the associated window.
Move	Alt+F7	Allows the icon to be moved with keys.
Size	Alt+F8	Inactive (not an option for icons).
Minimize	Alt+F9	Inactive (not an option for icons).
Maximize	Alt+F10	Opens the associated window and makes it fill the screen.
Lower	Alt+F3	Moves icon to bottom of icon stack.
Close	Alt+F4	Removes client from window manager management.

Pressing button 3 over an icon also causes the icon's window menu to open. To make a menu selection, drag the pointer over the menu and release button 3 when the desired item is highlighted.

Double-clicking button 1 on an icon calls the **f.restore_and_raise** function and restores the icon's associated window to its previous state. For example, if a maximized window is iconified, double-clicking button 1 restores it to its maximized state. Double-clicking button 1 on the icon box's icon opens the icon box and allow access to the contained icons. (Double-clicking a mouse button is a quick way to perform a function.) Pressing the Shift+Esc key sequence or the pop-up Menu key causes the icon window menu of the currently selected icon to open.

Icon Box

When icons begin to clutter the screen, they can be packed into an icon box. (To use an icon box, the window manager must be started with the icon box configuration already set.) The icon box is a window manager window that holds client icons. It includes one or more scroll bars when there are more window icons than the icon box can show at the same time.

Icons in the icon box can be manipulated with the mouse. The following button action descriptions summarize the behavior of this interface. Button actions apply whenever the pointer is on any part of the icon. Double-clicking an icon in the icon box calls the **f.restore_and_raise** function.

Icon Box

Button Action	**Description**
Button 1 click	Selects the icon.
Button 1 double-click	Normalizes (opens) the associated window.
Button 1 double-click	Raises an already *open* window to the top of the stack.
Button 1 drag	Moves the icon.
Button 3 press	Causes the menu for that icon to open.
Button 3 drag	Highlights items as the pointer moves across the menu.

Pressing mouse button 3 when the pointer is over an icon causes the menu for that icon to open.

Icon Menu for Icon Box		
Selection	**Accelerator**	**Description**
Restore	Alt+F5	Opens the associated window (if not already open).
Move	Alt+F7	Allows the icon to be moved with keys.
Size	Alt+F8	Inactive.
Minimize	Alt+F9	Inactive.
Maximize	Alt+F10	Opens the associated window (if not already open) and maximizes its size.
Lower	Alt+F3	Inactive.
Close	Alt+F4	Removes client from window manager management.

To pull down the window menu for the icon box itself, press button 1 with the pointer over the menu button for the icon box. The window menu of the icon box differs from the window menu of a client window: The **Close** selection is replaced with the **PackIcons** (Shift+Alt+F7) selection. When selected, the **PackIcons** option packs the icons in the box to achieve neat rows with no empty slots.

You can also post the window menu by pressing the Shift+Esc or Alt+Space key sequence. Pressing the pop-up Menu key causes the icon window menu of the currently selected icon to open.

Input Focus

The **mwm** command supports (by default) a keyboard input focus policy of *explicit selection*. This means when a window is selected to get keyboard input, it continues to get keyboard input until the window is withdrawn from window management, another window is explicitly selected to get keyboard input, or the window is iconified. Several resources control the input focus. The client window with the keyboard input focus has the active window appearance with a visually distinct window frame.

The following table and key action descriptions summarize the keyboard input focus selection behavior:

Input focus		
Button Action	**Object**	**Function Description**
Button 1 press	Window or window frame	Keyboard focus selection
Button 1 press	Icon	Keyboard focus selection

Function Description

Key Action	**Function Description**
Alt+Tab	Moves the input focus to next window in the window stack.
Alt+Shift+Tab	Moves the input focus to the previous window in the window stack (available only in explicit focus mode).

Window Stacking

There are two types of window stacks: global window stacks and an application's local family window stack.

The global stacking order of windows can be changed as a result of setting the keyboard input focus, iconifying a window, or performing a window manager window stacking function. When keyboard focus policy is explicit the default value of the **focusAutoRaise** resource is True. This causes a window to be raised to the top of the stack when it receives input focus, for example, by pressing button 1 on the title bar. The key actions defined in the preceding list raises the window receiving focus to the top of the stack.

In pointer mode, the default value of the **focusAutoRaise** resource is False; that is, the window stacking order is not changed when a window receives keyboard input focus. The following key actions can be used to cycle through the global window stack:

Windows Stacking function description

Key Action	**Function Description**
Alt+Esc	Places top window on bottom of stack.
Alt+Shift+Esc	Places bottom window on top of stack.

By default, a window's icon is placed on the bottom of the stack when the window is iconified; however, the default can be changed by the **lowerOnIconify** resource.

Transient windows (secondary windows such as dialog boxes) stay above their parent windows by default. However, an application's local family stacking order can changed to allow a transient window to be placed below its parent top-level window. The following parameter values show the modification of the stacking order for the **f.lower** function:

Transient windows

Item	Description
f.lower	Lowers the transient window within the family (staying above the parent) and lowers the family in the global window stack.
f.lower [**within**]	Lowers the transient window within the family (staying above the parent) but does not lower the family in the global window stack.
f.lower [**freeFamily**]	Lowers the window separate from its family stack (below the parent), but does not lower the family in the global window stack.

The **within** and **freeFamily** parameter values can also be used with the **f.raise** and **f.raise_lower** functions.

X Defaults

The **mwm** command is configured from its resource database. This database is built from the following sources. They are listed in order of precedence.

1. **mwm** command line options
2. **XENVIRONMENT** variable or **$HOME/.Xdefaults-host**
3. **RESOURCE_MANAGER** root window property or **$HOME/.Xdefaults**
4. **$HOME/Mwm**
5. **/usr/lib/X11/app-defaults/Mwm**.

The **/usr/lib/X11/app-defaults/Mwm** and **$HOME/Mwm** file names represent customary locations for these files. The actual location of the systemwide class resource file might depend on the **XFILESEARCHPATH** environment variable and the current language environment. The actual location of the user-specific class resource file might depend on the **XUSERFILESEARCHPATH** and **XAPPLRESDIR** environment variables and the current language environment.

Entries in the resource database can refer to other resource files for specific types of resources. These include files that contain bitmaps, fonts, and **mwm**-specific resources such as menus and behavior specifications (for example, button and key bindings).

Mwm is the resource class name of the **mwm** command and **mwm** is the resource name used by the **mwm** command to look up resources. (For looking up resources of multiple screens, the **-screens** command-line option specifies resource names such as **mwm_b+w** and **mwm_color**.) In the following discussion of resource specification, "Mwm" and "mwm" (and the aliased **mwm** resource names) can be used interchangeably, but "mwm" takes precedence over "Mwm". The **mwm** command uses the following types of resources:

X Defaults description

Item	Description
component appearance resource set	These resources specify appearance attributes of window manager user-interface components. They can be applied to the appearance of window manager menus, feedback windows (for example, the window reconfiguration feedback window), client window frames, and icons.
frame and icon component resource set	This subset of component appearance resources specifies attributes that are specific to frame and icon components.
general appearance and behavior resource set	These resources specify the **mwm** command appearance and behavior (for example, window management policies). They are not set separately for different **mwm** command user-interface components.

X Defaults description *(continued)*

Item	Description
client-specific resource set	These **mwm** resources can be set for a particular client window or class of client windows. They specify client-specific icon and client window frame appearance and behavior.

Resource identifiers can be either a resource name (for example, **foreground**) or a resource class (for example, **Foreground**). If the value of a resource is a file name and if the file name is prefixed by the ~/ (tilde followed by a slash) characters, it is relative to the path contained in the **HOME** environment variable (generally the user's home directory).

Component Appearance Resources

The syntax for specifying component appearance resources that apply to window manager icons, menus, and client window frames is as follows:

Mwm*ResourceID*

For example, **Mwm*foreground** is used to specify the foreground color for the **mwm** command menus, icons, client window frames, and feedback dialogs.

The syntax for specifying component appearance resources that apply to a particular **mwm** component is as follows:

Mwm*[*Menu*|*Icon*|*Client*|*Feedback*]***ResourceID*

If *Menu* is specified, the resource is applied only to Mwm menus; if *Icon* is specified, the resource is applied to icons; and if *Client* is specified, the resource is applied to client window frames. For example, **Mwm***Icon***foreground** is used to specify the foreground color for the **mwm** command icons, **Mwm***Menu***foreground** specifies the foreground color for the **mwm** command menus, and **Mwm***Client***foreground** is used to specify the foreground color for the **mwm** command client window frames.

The appearance of the title area of a client window frame (including window management buttons) can be separately configured. The syntax for configuring the title area of a client window frame is as follows:

Mwm*Client***Title***ResourceID*

For example, **Mwm***Client***Title*****foreground** specifies the foreground color for the title area. Defaults for title area resources are based on the values of the corresponding client window frame resources.

The appearance of menus can be configured based on the name of the menu. The syntax for specifying menu appearance by name is as follows:

Mwm*Menu***MenuName*****ResourceID*

For example, **Mwm***Menu***MyMenu*****foreground** specifies the foreground color for the menu named MyMenu.

The user can also specify resources for window manager menu components (the gadgets that comprise the menu). These may include, for example, a menu title, a title separator, one or more buttons, and separators. If a menu contains more than one instance of a class, such as multiple **PushButtonGadget** gadgets, the name of the first instance is **PushButtonGadget1**, the second is **PushButtonGadget2**, and so on. The following list identifies the naming conventions used for window manager menu components:

Component Appearance Resources

Item	Description
TitleName	Menu title LabelGadget
TitleSeparator	Menu title SeparatorGadget

Component Appearance Resources *(continued)*

Item	Description
CascadeButtonGadget*n*	CascadeButtonGadget
PushButtonGadget*n*	PushButtonGadget
SeparatorGadget*n*	SeparatorGadget

The following component appearance resources that apply to all window manager parts can be specified.

Component Appearance Resource Set

Component Appearance Resource Set	
Name	**Properties**
background	**Class** Background **Value type** color **Default** varies[1]
backgroundPixmap	**Class** BackgroundPixmap **Value type** string[2] **Default** varies[1]
bottomShadowColor	**Class** Foreground **Value type** color **Default** varies[1]
bottomShadowPixmap	**Class** BottomShadowPixmap **Value type** string[2] **Default** varies[1]
fontList	**Class** FontList **Value type** string[3] **Default** "fixed"

Component Appearance Resource Set *(continued)*	
Name	**Properties**
foreground	**Class** Foreground **Value type** color **Default** varies[1]
saveUnder	**Class** SaveUnder **Value type** True of False **Default** False
topShadowColor	**Class** Background **Value type** color **Default** varies[1]
topShadowPixmap	**Class** TopShadowPixmap **Value type** string[2] **Default** varies[1]
background	**Class** Background **Value type** color **Default** varies[1]
backgroundPixmap	**Class** BackgroundPixmap **Value type** string[2] **Default** varies[1]
bottomShadowColor	**Class** Foreground **Value type** color **Default** varies[1]

Component Appearance Resource Set *(continued)*	
Name	**Properties**
bottomShadowPixmap	**Class** BottomShadowPixmap **Value type** string[2] **Default** varies[1]
fontList	**Class** FontList **Value type** string[3] **Default** "fixed"
foreground	**Class** Foreground **Value type** color **Default** varies[1]
saveUnder	**Class** SaveUnder **Value type** True of False **Default** False
topShadowColor	**Class** Background **Value type** color **Default** varies[1]
topShadowPixmap	**Class** TopShadowPixmap **Value type** string[2] **Default** varies[1]

Note:

1. The default is chosen based on the visual type of the screen.
2. Image name.
3. X Version 11 Release 4 (X11R4) font description.

Background

Item	Description
background (class **Background**)	Specifies the background color. Any legal X color can be specified. The default value is chosen based on the visual type of the screen.
backgroundPixmap (class **BackgroundPixmap**)	Specifies the background pixmap of the **mwm** decoration when the window is inactive (does not have the keyboard focus). The default value is chosen based on the visual type of the screen.
bottomShadowColor (class **Foreground**)	Specifies the bottom shadow color. This color is used for the lower and right bevels of the window manager decoration. Any legal X color can be specified. The default value is chosen based on the visual type of the screen.
bottomShadowPixmap (class **BottomShadowPixmap**)	Specifies the bottom shadow pixmap. This pixmap is used for the lower and right bevels of the window manager decoration. The default is chosen based on the visual type of the screen.
fontList (class **FontList**)	Specifies the font used in the window manager decoration. The character encoding of the font needs to match the character encoding of the strings that are used. The default is the fixed value.
foreground (class **Foreground**)	Specifies the foreground color. The default is chosen based on the visual type of the screen.
saveUnder (class **SaveUnder**)	Controls the repainting of windows that are uncovered after being obscured. This resource indicates whether *save unders* are used for **mwm** components. For this to have any effect, save unders must be implemented by the X server. If save unders are implemented, the X server saves the contents of windows obscured by windows that have the save under attribute set. If the **saveUnder** resource is True, the **mwm** command sets the save under attribute on the window manager frame of any client that has it set. If the **saveUnder** resource is False, save unders is not used on any window manager frames. The default value is False.
topShadowColor (class **Background**)	Specifies the top shadow color. This color is used for the upper and left bevels of the window manager decoration. The default is chosen based on the visual type of the screen.
topShadowPixmap (class **TopShadowPixmap**)	Specifies the top shadow pixmap. This pixmap is used for the upper and left bevels of the window manager decoration. The default is chosen based on the visual type of the screen.

Frame and Icon Component Resource Set

Note: Hyphens in the following table are for readability purposes only. Do not include hyphens within names in programs.

Frame and Icon Component Resource Set	
Name	**Properties**
activeBackground	**Class** Background **Value type** color **Default** varies[1]
activeBackground-Pixmap	**Class** BackgroundPixmap **Value type** string[2] **Default** varies[1]
activeBottomShadow-Color	**Class** Foreground **Value type** color **Default** varies[1]
activeBottomShadow-Pixmap	**Class** BottomShadow-Pixmap **Value type** string[2] **Default** varies[1]
activeForeground	**Class** Foreground **Value type** color **Default** varies[1]
activeTopShadowColor	**Class** Background **Value type** color **Default** varies[1]
activeTopShadowPixmap	**Class** TopShadowPixmap **Value type** string[2] **Default** varies[1]

Frame and Icon Component Resource Set *(continued)*	
Name	**Properties**
activeBackground	**Class** Background **Value type** color **Default** varies[1]
activeBackgroundPixmap	**Class** BackgroundPixmap **Value type** string[2] **Default** varies[1]
activeBottomShadowColor	**Class** Foreground **Value type** color **Default** varies[1]
activeBottomShadowPixmap	**Class** BottomShadowPixmap **Value type** string[2] **Default** varies[1]
activeForeground	**Class** Foreground **Value type** color **Default** varies[1]
activeTopShadowColor	**Class** Background **Value type** color **Default** varies[1]
activeTopShadowPixmap	**Class** TopShadowPixmap **Value type** string[2] **Default** varies[1]

Note:

1. The default is chosen based on the visual type of the screen.
2. Image name.

Background

Item	Description
activeBackground (class **Background**)	Specifies the background color of the **mwm** decoration when the window is active (has the keyboard focus). The default is chosen based on the visual type of the screen.
activeBackgroundPixmap (class **BackgroundPixmap**)	Specifies the background pixmap of the **mwm** decoration when the window is active (has the keyboard focus). The default is chosen based on the visual type of the screen.
activeBottomShadowColor (class **Foreground**)	Specifies the bottom shadow color of the **mwm** decoration when the window is active (has the keyboard focus). The default is chosen based on the visual type of the screen.
activeBottomShadowPixmap (class **BottomShadowPixmap**)	Specifies the bottom shadow pixmap of the **mwm** decoration when the window is active (has the keyboard focus). The default is chosen based on the visual type of the screen.
activeForeground (class **Foreground**)	Specifies the foreground color of the **mwm** decoration when the window is active (has the keyboard focus). The default is chosen based on the visual type of the screen.
activeTopShadowColor (class **Background**)	Specifies the top shadow color of the **mwm** decoration when the window is active (has the keyboard focus). The default is chosen based on the visual type of the screen.
activeTopShadowPixmap (class **TopShadowPixmap**)	Specifies the top shadow pixmap of the **mwm** decoration when the window is active (has the keyboard focus). The default is chosen based on the visual type of the screen.

General Appearance and Behavior Resources

The syntax for specifying general appearance and behavior resources is as follows:

Mwm*ResourceID*

For example, **Mwm*keyboardFocusPolicy** specifies the window manager policy for setting the keyboard focus to a particular client window.

General Appearance and Behavior Resource Set

Note: Hyphens in the following table are for readability purposes only. Do not include hyphens within names in programs.

General Appearance and Behavior Resource Set	
Name	**Properties**
autoKeyFocus	**Class** AutoKeyFocus **Value type** True or False **Default** True
autoRaiseDelay	**Class** AutoRaiseDelay **Value type** millisec. **Default** 500
bitmapDirectory	**Class** BitmapDirectory **Value type** directory **Default** /usr/include/X11/bitmaps
buttonBindings	**Class** ButtonBindings **Value type** string **Default** "DefaultButton-Bindings"
cleanText	**Class** CleanText **Value type** True or False **Default** True
clientAutoPlace	**Class** ClientAutoPlace **Value type** True or False **Default** True
colormapFocusPolicy	**Class** ColormapFocus-Policy **Value type** string **Default** keyboard

General Appearance and Behavior Resource Set *(continued)*	
Name	**Properties**
configFile	**Class** ConfigFile **Value type** file **Default** .mwmrc
defaultSetBehavior	**Class** defaultSetBehavior **Value type** True or False **Default** True
deiconifyKeyFocus	**Class** DeiconifyKeyFocus **Value type** True or False **Default** True
doubleClickTime	**Class** DoubleClickTime **Value type** milliseconds **Default** multiclick time
enableWarp	**Class** enableWarp **Value type** True or False **Default** True
enforceKeyFocus	**Class** EnforceKeyFocus **Value type** True or False **Default** True
fadeNormalIcon	**Class** FadeNormalIcon **Value type** True or False **Default** False

General Appearance and Behavior Resource Set *(continued)*	
Name	**Properties**
feedbackGeometry	**Class** FeedbackGeometry **Value type** string **Default** center on screen
frameBorderWidth	**Class** FrameBorderWidth **Value type** pixels **Default** varies
iconAutoPlace	**Class** IconAutoPlace **Value type** True or False **Default** True
iconBoxGeometry	**Class** IconBoxGeometry **Value type** string **Default** 6x1+0-0
iconBoxName	**Class** IconBoxName **Value type** string **Default** iconbox
iconBoxSBDisplayPolicy	**Class** IconBoxSBDisplayPolicy **Value type** string **Default** all
iconBoxTitle	**Class** IconBoxTitle **Value type** XmString **Default** Icons

General Appearance and Behavior Resource Set *(continued)*	
Name	**Properties**
iconClick	**Class** IconClick **Value type** True or False **Default** True
iconDecoration	**Class** IconDecoration **Value type** string **Default** varies
iconImage-Maximum	**Class** IconImage-Maximum **Value type** width x height **Default** 50x50
iconImageMinimum	**Class** IconImageMinimum **Value type** width x height **Default** 16x16
iconPlacement	**Class** IconPlacement **Value type** string **Default** left bottom
iconPlacementMargin	**Class** IconPlacementMargin **Value type** pixels **Default** varies
interactivePlacement	**Class** InteractivePlacement **Value type** True or False **Default** False

General Appearance and Behavior Resource Set *(continued)*	
Name	**Properties**
keyBindings	**Class** KeyBindings **Value type** string **Default** "DefaultKeyBindings"
keyboardFocusPolicy	**Class** KeyboardFocusPolicy **Value type** string **Default** explicit
limitResize	**Class** LimitResize **Value type** True or False **Default** True
lowerOnIconify	**Class** LowerOnIconify **Value type** True or False **Default** True

Maximum size	
Item	**Description**
maximumMaximuSize	**Class** MaximumMaximuSize **Value type** width x height (pixels) **Default** 2X screen width & height
moveOpaque	**Class** MoveOpaque **Value type** True or False **Default** False

Maximum size *(continued)*	
Item	**Description**
moveThreshold	**Class** MoveThreshold **Value type** pixels **Default** 4
multiScreen	**Class** MultiScreen **Value type** True or False **Default** False
passButtons	**Class** PassButtons **Value type** True or False **Default** False
PassSelectButton	**Class** PassSelectButton **Value type** True or False **Default** True
positionIsFrame	**Class** PositionIsFrame **Value type** True or False **Default** True
positionOnScreen	**Class** PositionOnScreen **Value type** True or False **Default** True
quitTimeout	**Class** QuitTimeout **Value type** milliseconds **Default** 1000

Maximum size *(continued)*	
Item	**Description**
`raiseKeyFocus`	**Class** `RaiseKeyFocus` **Value type** True or False **Default** False
`resizeBorderWidth`	**Class** `ResizeBorderWidth` **Value type** pixels **Default** varies
`resizeCursors`	**Class** `ResizeCursors` **Value type** True or False **Default** True
`screens`	**Class** `Screens` **Value type** string **Default** varies
`showFeedback`	**Class** `ShowFeedback` **Value type** string **Default** all
`startupKeyFocus`	**Class** `StartupKeyFocus` **Value type** True or False **Default** True
`transientDecoration`	**Class** `TransientDecoration` **Value type** string **Default** menu title

Maximum size *(continued)*	
Item	**Description**
`transientFunctions`	**Class** `TransientFunctions` **Value type** string **Default** -minimize -maximize
`useIconBox`	**Class** `UseIconBox` **Value type** True or False **Default** False
`wMenuButtonClick`	**Class** `WMenuButtonClick` **Value type** True or False **Default** True
`wMenuButtonClick2`	**Class** `WMenuButtonClick2` **Value type** True or False **Default** True

AutoKeyFocus

Item	**Description**
autoKeyFocus (class **AutoKeyFocus**)	Controls whether the focus is set to the previous window that had the focus. If the **autoKeyFocus** resource is given a value of True and a window with the keyboard input focus is withdrawn from window management or is iconified, the focus is set to the previous window that had the focus. If the value given is False, there is no automatic setting of the keyboard input focus. It is recommended that both the **autoKeyFocus** resource and the **startupKeyFocus** resource be set to a value of True to work with tear-off menus. The default value is True. This resource is available only when the keyboard input focus policy is set to the explicit value.
autoRaiseDelay (class **AutoRaiseDelay**)	Specifies the amount of time in milliseconds (ms) that the **mwm** command waits before raising a window after it gets the keyboard focus. The default value of this resource is 500 (milliseconds). This resource is available only when the **focusAutoRaise** resource is True and the keyboard focus policy is the pointer value.

Item	Description
bitmapDirectory (class **BitmapDirectory**)	Identifies a directory to be searched for bitmaps referenced by **mwm** resources. This directory is searched if a bitmap is specified without an absolute path name. The default value for this resource is **/usr/include/X11/bitmaps**. The **/usr/include/X11/bitmaps** directory represents the customary locations for this directory. The actual location of this directory may vary on some systems. If the bitmap is not found in the specified directory, the **XBMLANGPATH** environment variable is searched.
buttonBindings (class **ButtonBindings**)	Identifies the set of button bindings for window management functions. The named set of button bindings is specified in the **mwm** resource description file. These button bindings are merged with the built-in default bindings. The default value for this resource is **DefaultButtonBindings**.
cleanText (class **CleanText**)	Controls the display of window manager text in the client title and feedback windows. If the default value of True is used, the text is drawn with a clear (no stipple) background. This makes text easier to read on monochrome systems where a **backgroundPixmap** is specified. Only the stippling in the area immediately around the text is cleared. If False, the text is drawn directly on top of the existing background.
clientAutoPlace (class **ClientAutoPlace**)	Determines the position of a window when the window does not have a user-specified position. With a value of True, windows are positioned with the top left corners of the frames offset horizontally and vertically. A value of False causes the currently configured position of the window to be used. In either case, the **mwm** command attempts to place the windows totally on-screen. The default value is True.
colormapFocusPolicy (class **ColormapFocusPolicy**)	Indicates the colormap focus policy that is to be used. If the resource value is explicit, a colormap selection action is done on a client window to set the colormap focus to that window. If the value is pointer, the client window containing the pointer has the colormap focus. If the value is keyboard, the client window that has the keyboard input focus has the colormap focus. The default value for this resource is keyboard.

AutoKeyFocus *(continued)*

Item	Description
configFile (class **ConfigFile**)	Contains the path name for an **mwm** resource description file. If the path name begins with the ~/ characters, the **mwm** command considers it to be relative to the user's home directory (as specified by the **HOME** environment variable). If the **LANG** environment variable is set, the **mwm** command looks for **$HOME/$LANG/configFile**. If that file does not exist or if **LANG** is not set, **mwm** looks for **$HOME/configFile**. If the **configFile** path name does not begin with the ~/ characters, **mwm** considers it to be relative to the current working directory. If the **configFile** resource is not specified or if that file does not exist, the **mwm** command uses several default paths to find a configuration file. If the **LANG** environment variable is set, the **mwm** command looks for the configuration file first in the **$HOME/$LANG/.mwmrc** file. If that file does not exist or if the **LANG** environment variable is not set, the **mwm** command looks for the **$HOME/.mwmrc** file. If the **$HOME/.mwmrc** file does not exist and if the **LANG** environment variable is set, the **mwm** command next looks for the **/usr/lib/X11/$LANG/system.mwmrc** file. If the **/usr/lib/X11/$LANG/system.mwmrc** file does not exist or if the **LANG** environment variable is not set, the **mwm** command looks for **/usr/lib/X11/system.mwmrc**.
defaultSetBehavior (class **DefaultSetBehavior**)	Determines whether the **mwm** command automatically add key bindings to the **f.set_behavior** function (see .mwmrc file). If the value for the **defaultSetBehavior** resource is **True** (or On), regardless of the key bindings defined in the **.mwmrc** configuration file, `Alt Ctrl<key>1` and `Alt Shift Ctrl<Key>exclam` is bound to the **f.set_behavior** function. If the value for the **defaultSetBehavior** resource is **False** (or Off), the **f.set_behavior** function is bound to a key specified in the **.mwmrc** configuration file. If no key bindings are specified in the configuration file, the **mwm** command uses the default key binding. The default value for the **defaultSetBehavior** resource is **True** (or On).
deiconifyKeyFocus (class **DeiconifyKeyFocus**)	Determines whether a window receives the keyboard input focus when it is deiconified (normalized). The default value is True. This resource applies only when the keyboard input focus policy is set to the explicit value.
doubleClickTime (class **DoubleClickTime**)	Sets the maximum time (in ms) between the clicks (button presses) that make up a double-click. The default value of this resource is the display's multiclick time.

Item	Description
enableWarp (class **EnableWarp**)	Causes the **mwm** command to *warp* the pointer to the center of the selected window during keyboard-controlled resize and move operations. Setting the value to False causes the **mwm** command to leave the pointer at its original place on the screen unless the user explicitly moves it with the cursor keys or pointing device. The default value of this resource is True.
enforceKeyFocus (class **EnforceKeyFocus**)	Determines whether the keyboard input focus is always explicitly set to selected windows even if there is an indication that they are *globally active* input windows. (An example of a globally active window is a scroll bar that can be operated without setting the focus to that client.) If the resource is False, the keyboard input focus is not explicitly set to globally active windows. The default value is True.
fadeNormalIcon (class **FadeNormalIcon**)	Determines whether an icon is unavailable whenever it is normalized (its window is opened). The default value is False.
feedbackGeometry (class **FeedbackGeometry**)	Sets the position of the move and resize feedback window. If this resource is not specified, the default is to place the feedback window at the center of the screen. The value of the resource is a standard window geometry string with the following syntax: [=][{+-}*XOffset*{+-}*YOffset*]
frameBorderWidth (class **FrameBorderWidth**)	Specifies the width in pixels of a client window frame border without resize handles. The border width includes the three-dimensional (3-D) shadows. The default value is based on the size and resolution of the screen.
iconAutoPlace (class **IconAutoPlace**)	Indicates whether the window manager arranges icons in a particular area of the screen or places each icon where the window was when it was iconified. The True value indicates that icons are arranged in a particular area of the screen determined by the **iconPlacement** resource. The False value indicates that an icon is placed at the location of the window when it is iconified. The default is True.
iconBoxGeometry (class **IconBoxGeometry**)	Indicates the initial position and size of the icon box. The value of the resource is a standard window geometry string with the following syntax: [=][*Width*x*Height*][{+-}*XOffset*{+-}*YOffset*] If the offsets are not provided, the **iconPlacement** policy is used to determine the initial placement. The units for width and height are columns and rows. The actual screen size of the icon box window depends on the **iconImageMaximum** (size) and **iconDecoration** resources. The default value for size is (6 times **iconWidth** + padding) wide by (1 times **iconHeight** + padding) high. The default value of the location is +0 -0.
iconBoxName (class **IconBoxName**)	Specifies the name that is used to look up icon box resources. The default name is **iconbox**.

Item	Description
iconBoxSBDisplayPolicy (class **IconBoxSBDisplayPolicy**)	Specifies the scroll bar display policy of the window manager in the icon box. The resource has the following three possible values: all, vertical, and horizontal. The default value, all, causes both vertical and horizontal scroll bars always to be displayed. The vertical value causes a single vertical scroll bar to be displayed in the icon box and sets the orientation of the icon box to horizontal (regardless of the **iconBoxGeometry** specification). The horizontal value causes a single horizontal scroll bar to be displayed in the icon box and sets the orientation of the icon box to vertical (regardless of the **iconBoxGeometry** specification).
iconBoxTitle (class **IconBoxTitle**)	Specifies the name that is used in the title area of the icon box frame. The default value is Icons.
iconClick (class **IconClick**)	Specifies whether the system menu is posted and remains posted when an icon is clicked. The default value is True.
iconDecoration (class **IconDecoration**)	Specifies the general icon decoration. The resource value is label (only the label part is displayed) or image (only the image part is displayed) or label image (both the label and image parts are displayed). A value of activelabel can also be specified to get a label (not truncated to the width of the icon) when the icon is selected. The default icon decoration for icon box icons is that each icon has a label part and an image part (label image). The default icon decoration for standalone icons is that each icon has an active label part, a label part, and an image part (activelabel, label, and image).
iconImageMaximum (class **IconImageMaximum**)	Specifies the maximum size of the icon image. The resource value is *Width* x *Height* (for example, 64x64). The maximum supported size is 128x128. The default value of this resource is 50x50.
iconImageMinimum (class **IconImageMinimum**)	Specifies the minimum size of the icon image. The resource value is *Width* x *Height* (for example, 32x50). The minimum supported size is 16x16. The default value of this resource is 16x16.
iconPlacement (class **IconPlacement**)	Specifies the icon placement scheme to be used. The resource value has the following syntax: *PrimaryLayout SecondaryLayout [Tight]* The layout values are described as one of the following: **top** Lays out the icons from top to bottom. **bottom** Lays out the icons from bottom to top. **left** Lays out the icons from left to right. **right** Lays out the icons from right to left.

Item	Description
	A horizontal (vertical) layout value must not be used for both the *PrimaryLayout* and the *SecondaryLayout* (for example, do not use top for the *PrimaryLayout* and bottom for the *SecondaryLayout*). The *PrimaryLayout* indicates at the time an icon placement is done whether the icon is placed in a row or a column and the direction of placement. The *SecondaryLayout* indicates where to place new rows or columns. For example, the top right value indicates that icons should be placed top to bottom on the screen and that columns should be added from right to left on the screen. The default placement is the left bottom value (icons are placed from left to right on the screen, with the first row on the bottom of the screen, and new rows added from the bottom of the screen to the top of the screen). A tight value places icons with zero spacing between icons. This value is useful for aesthetic reasons, as well as for terminals with small screens. The following is a list of options for **iconPlacement** values: **Icon Placement** **Appropriate Scheme** **From left to right across the top of the screen, new rows below** Left top **From right to left across the top of the screen, new rows below** Right top **From left to right across the bottom of the screen, new rows above** Left bottom **From right to left across the bottom of the screen, new rows above** Right bottom **From bottom to top along the left of the screen, new columns to right** Bottom left **From bottom to top along the right of the screen, new columns to left** Bottom right **From top to bottom along the left of the screen, new columns to right** Top left **From top to bottom along the right of the screen, new columns to left** Top right

AutoKeyFocus *(continued)*

Item	Description
iconPlacementMargin (class **IconPlacementMargin**)	Sets the distance between the edge of the screen and the icons that are placed along the edge of the screen. The value should be greater than or equal to 0. A default value is used if the value specified is invalid. The default value for this resource is equal to the space between icons as they are placed on the screen (this space is based on maximizing the number of icons in each row and column).
interactivePlacement (class **InteractivePlacement**)	Controls the initial placement of new windows on the screen. If the value is True, the pointer shape changes before a new window is placed on the screen to indicate to the user that a position needs to be selected for the upper-left corner of the window. If the value is False, windows are placed according to the initial window configuration attributes. The default value of this resource is False.

Key Bindings

Item	Description
keyBindings (class **KeyBindings**)	Identifies the set of key bindings for window management functions. If specified, these key bindings replace the built-in default bindings. The named set of key bindings is specified in **mwm** resource description file. The default value for this resource is DefaultKeyBindings.
keyboardFocusPolicy (class **KeyboardFocusPolicy**)	Determines the keyboard focus policy. If set to the pointer value, the keyboard focus policy has the keyboard focus set to the client window that contains the pointer (the pointer could also be in the client window decoration that the **mwm** command adds). If set to the explicit value, the policy is to have the keyboard focus set to a client window when the user presses the left mouse button with the pointer on the client window or any part of the associated **mwm** decoration. The default value for this resource is explicit.
limitResize (class **LimitResize**)	Determines whether the user is allowed to resize a window to greater than the maximum size. If this resource is True, the user is not allowed to resize a window to greater than the maximum size. The default value for this resource is True.
lowerOnIconify (class **LowerOnIconify**)	Determines whether a window icon is displayed on the bottom of the window stack when the window is iconified (minimized). A value of False places the icon in the stacking order at the same place as its associated window. The default value of this resource is True.
maximumMaximumSize (class **MaximumMaximumSize**)	Limits the maximum size of a client window as set by the user or client. The resource value is *Width* x *Height* (for example, 1024x1024) where the width and height are in pixels. The default value of this resource is twice the screen width and height.
moveOpaque (class **MoveOpaque**)	Controls whether the actual window is moved or a rectangular outline of the window is moved. A default value of False displays a rectangular outline on move operations.

Key Bindings *(continued)*

Item	Description
moveThreshold (class **MoveThreshold**)	Controls the sensitivity of dragging operations that move windows and icons. The value of this resource is the number of pixels that the locator is moved with a button down before the move operation is initiated. This is used to prevent window and icon movement when you click or double-click and there is unintentional pointer movement with the button down. The default value of this resource is 4 (pixels).
multiScreen (class **MultiScreen**)	Determines whether the **mwm** command manages all the screens on the display. If False, the **mwm** command manages only a single screen. The default value is False.
passButtons (class **PassButtons**)	Indicates whether button press events are passed to clients after they are used to do a window manager function in the client context. If the resource value is False, the button press is not passed to the client. If the value is True, the button press is passed to the client window. The window manager function is done in either case. The default value for this resource is False.
passSelectButton (class **PassSelectButton**)	Indicates whether to pass the select button press events to clients after they are used to do a window manager function in the client context. If the resource value is False, the button press is not passed to the client. If the value is True, the button press is passed to the client window. The window manager function is done in either case. The default value for this resource is True.
positionIsFrame (class **PositionIsFrame**)	Indicates how client window position information (from the **WM_NORMAL_HINTS** property and from configuration requests) is to be interpreted. If the resource value is True, the information is interpreted as the position of the MWM client window frame. If the value is False, it is interpreted as being the position of the client area of the window. The default value of this resource is True.
positionOnScreen (class **PositionOnScreen**)	Indicates that windows should initially be placed (if possible) so that they are not clipped by the edge of the screen (if the resource value is True). If a window is larger than the size of the screen, at least the upper-left corner of the window is on-screen. If the resource value is False, windows are placed in the requested position even if totally off-screen. The default value of this resource is True.
quitTimeout (class **QuitTimeout**)	Specifies the amount of time in milliseconds that the **mwm** command waits for a client to update the **WM_COMMAND** property after the **mwm** command has sent the **WM_SAVE_YOURSELF** message. This protocol is used only for those clients that have a **WM_SAVE_YOURSELF** atom and no **WM_DELETE_WINDOW** atom in the **WM_PROTOCOLS** client window property. The default value of this resource is 1000 (milliseconds). See the **f.kill** function for additional information.

Key Bindings *(continued)*

Item	Description
raiseKeyFocus (class **RaiseKeyFocus**)	Specifies whether a window raised by means of the **f.normalize_and_raise** function also receives the input focus. The default value of this resource is False. This resource is available only when the keyboard input focus policy is set to the explicit value.
resizeBorderWidth (class **ResizeBorderWidth**)	Specifies the width (in pixels) of a client window frame border with resize handles. The specified border width includes the 3-D shadows. The default value is based on the size and resolution of the screen.
resizeCursors (class **ResizeCursors**)	Indicates whether the resize cursors are always displayed when the pointer is in the window size border. If True, the cursors are shown; otherwise, the window manager cursor is shown. The default value is True.
screens (class **Screens**)	Specifies the resource names to use for the screens managed by the **mwm** command. If the **mwm** command is managing a single screen, only the first name in the list is used. If the **mwm** command is managing multiple screens, the names are assigned to the screens in order, starting with screen 0. For example, screen 0 gets the first name and screen 1 gets the second name. Examples of default screen names are 0 and 1.

Item	Description
showFeedback (class **ShowFeedback**)	Controls when feedback information is displayed. It controls both window position and size feedback during move or resize operations and initial client placement. It also controls window manager message and dialog boxes. The value for this resource is a list of names of the feedback options to be enabled or disabled; the names must be separated by a space. If an option is preceded by a minus sign, that option is excluded from the list. The sign of the first item in the list determines the initial set of options. If the sign of the first option is - (a minus sign), the **mwm** command assumes all options are present and starts subtracting from that set. If the sign of the first decoration is + (a plus sign) or is not specified, the **mwm** command starts with no options and builds up a list from the resource. The names of the feedback options are as follows: **all** Shows all feedback (default value). **behavior** Confirms the behavior switch. **kill** Confirms on receipt of the **KILL** signal. **move** Shows position during the move. **none** Shows no feedback. **placement** Shows position and size during initial placement. **quit** Confirms quitting MWM. **resize** Shows size during resize. **restart** Confirms MWM restart. The following command line illustrates the syntax for the **showFeedback** resource: **Mwm*showFeedback: placement resize behavior restart** This resource specification provides feedback for initial client placement and resize, and it enables the dialog boxes to confirm the restart and set behavior functions. It disables feedback for the move function. The default value for this resource is the all value.

Key Bindings *(continued)*

Item	Description
startupKeyFocus (class **StartupKeyFocus**)	Determines whether a window gets the keyboard input focus when the window is mapped (that is, initially managed by the window manager). It is recommended that both the **autoKeyFocus** resource and the **startupKeyFocus** resource be set to a value of True to work with tear-off menus. The default value is True. This resource is available only when the keyboard input focus policy is set to the explicit value.
transientDecoration (class **TransientDecoration**)	Controls the amount of decoration that Mwm puts on transient windows. The decoration specification is exactly the same as for the **clientDecoration** (client-specific) resource. Transient windows are identified by the **WM_TRANSIENT_FOR** property, which is added by the client to indicate a relatively temporary window. The default value for this resource is the menu title value (that is, transient windows have resize borders and a title bar with a window menu button). An application can also specify which decorations the window manager should apply to its windows. If it does so, the window manager applies only those decorations indicated by both the application and the **transientDecoration** resource. Otherwise, the window manager applies only the decorations indicated by the **transientDecoration** resource.
transientFunctions (class **TransientFunctions**)	Indicates which window management functions are applicable (or not applicable) to transient windows. The function specification is exactly the same as for the **clientFunctions** (client-specific) resource. The default value for this resource is **-minimize -maximize**. An application can also specify which functions the window manager should apply to its windows. If it does so, the window manager applies only those functions indicated by both the application and the **transientFunctions** resource. Otherwise, the window manager applies only the functions indicated by the **transientFunctions** resource.
useIconBox (class **UseIconBox**)	Determines whether icons are placed in an icon box. If this resource is given a value of True, icons are placed in an icon box. When an icon box is not used, the icons are placed on the root window (default value).
wMenuButtonClick (class **WMenuButtonClick**)	Indicates whether the window menu is posted and remains posted after a click of the mouse when the pointer is over the **Window Menu** button. If the value given this resource is True, the menu remains posted. True is the default value for this resource.
wMenuButtonClick2 (class **WMenuButtonClick2**)	Indicates whether a double-click action on the **Window Menu** button performs an **f.kill** function. When this resource is given the default value of True, a double-click action on the **Window Menu** button performs an **f.kill** function.

Client-Specific Resources

The syntax for specifying client-specific resources is as follows:

Mwm*ClientNameOrClass******ResourceID*

For example, `Mwm*mterm*windowMenu` is used to specify the window menu to be used with `mterm` clients.

The syntax for specifying client-specific resources for all classes of clients is as follows:

Mwm*ResourceID*

Specific client specifications take precedence over the specifications for all clients. For example, **Mwm*windowMenu** is used to specify the window menu to be used for all classes of clients that do not have a window menu specified.

The syntax for specifying resource values for windows that have an unknown name and class (that is, windows that do not have a **WM_CLASS** property associated with them) is as follows:

Mwm*defaults*ResourceID*

For example, **Mwm*defaults*iconImage** is used to specify the icon image to be used for windows that have an unknown name and class.

Client-Specific Resource Set

Note: Hyphens in the following table are for readability purposes only. Do not include hyphens within names in programs.

Client-Specific Resource Set	
Name	**Properties**
`clientDecoration`	**Class** `ClientDecoration` **Value type** all **Default**
`clientFunctions`	**Class** `ClientFunctions` **Value type** string **Default** all
`focusAutoRaise`	**Class** `FocusAutoRaise` **Value type** True or False **Default** varies

Client-Specific Resource Set *(continued)*	
Name	**Properties**
iconImage	**Class** IconImage **Value type** pathname **Default** (image)
iconImageBackground	**Class** Background **Value type** color **Default** icon background
iconImageBottomShadowColor	**Class** Foreground **Value type** color **Default** icon bottom shadow
iconImageBottomShadowPixmap	**Class** BottomShadowPixmap **Value type** color **Default** icon bottom shadow pixmap
iconImageForeground	**Class** Foreground **Value type** color **Default** varies
iconImageTopShadowColor	**Class** Background **Value type** color **Default** icon top shadow color
iconImageTopShadoPixmap	**Class** TopShadowPixmap **Value type** color **Default** icon top shadow pixmap

Client-Specific Resource Set *(continued)*	
Name	**Properties**
matteBackground	**Class** Background **Value type** color **Default** background
matteBottomShadowColor	**Class** Foreground **Value type** color **Default** bottom shadow color
matteBottomShadowPixmap	**Class** BottomShadowPixmap **Value type** color **Default** bottom shadow pixmap
matteForeground	**Class** Foreground **Value type** color **Default** foreground
matteTopShadowColor	**Class** Background **Value type** color **Default** top shadow color
matteTopShadowPixmap	**Class** TopShadowPixmap **Value type** color **Default** top shadow pixmap
matteWidth	**Class** MatteWidth **Value type** pixels **Default** 0

Client-Specific Resource Set *(continued)*	
Name	**Properties**
`maximumClientSize`	**Class** `MaximumClientSize` **Value type** width x height, vertical, horizontal **Default** fill the screen
`useClientIcon`	**Class** `UseClientIcon` **Value type** True or False **Default** F
`usePPosition`	**Class** `UsePPosition` **Value type** string **Default** nonzero
`windowMenu`	**Class** `WindowMenu` **Value type** string **Default** DefaultWindowMenu

Item	Description
clientDecoration (class **ClientDecoration**)	Controls the amount of window frame decoration. The resource is specified as a list of decorations to specify their inclusion in the frame. If a decoration is preceded by **-** (a minus sign), that decoration is excluded from the frame. The sign of the first item in the list determines the initial amount of decoration. If the sign of the first decoration is a minus sign, the **mwm** command assumes all decorations are present and starts subtracting from that set. If the sign of the first decoration is plus (or not specified), the **mwm** command starts with no decoration and builds up a list from the resource. An application can also specify which decorations the **mwm** command should apply to its windows. If it does so, the **mwm** command applies only those decorations indicated by both the application and the **clientDecoration** resource. Otherwise, the **mwm** command applies the decorations indicated by the **clientDecoration** resource. Following is a list of window frame decorations: **all** Specifies to include all decorations (default value). **border** Specifies the window border. **maximize** Specifies the Maximize button (includes title bar). **minimize** Specifies the Minimize button (includes title bar). **none** Specifies no decorations. **resizeh** Specifies the border resize handles (includes border). **menu** Specifies the Window Menu button (includes title bar). **title** Specifies the title bar (includes border). Following are examples of window frame decoration commands: `Mwm*XClock.clientDecoration: -resizeh -maximize` This removes the resize handles and Maximize button from XClock windows. `Mwm*XClock.clientDecoration: menu minimize border` This removes the resize handles and Maximize button from XClock windows. Note that either `menu` or `minimize` implies title.

Item	Description
clientFunctions (class **ClientFunctions**)	Indicates which **mwm** functions are applicable (or not applicable) to the client window. The value for the resource is a list of functions. If the first function in the list has - (a minus sign) in front of it, the **mwm** command starts with all functions and subtracts from that set. If the first function in the list has a + (plus sign) in front of it, the **mwm** command starts with no functions and builds up a list. Each function in the list must be preceded by the appropriate + (plus) or - (minus) sign and separated from the next function by a space. An application can also specify which functions the **mwm** command should apply to its windows. If it does so, the **mwm** command applies only those functions indicated by both the application and the **clientFunctions** resource. Otherwise, the **mwm** command applies the functions indicated by the **clientFunctions** resource. Following is a list of functions available for this resource: **all** Specifies to include all functions (default value). **none** Specifies no functions. **resize** Specifies **f.resize**. **move** Specifies **f.move**. **minimize** Specifies **f.minimize**. **maximize** Specifies **f.maximize**. **close** Specifies **f.kill**.
focusAutoRaise (class **FocusAutoRaise**)	Determines whether clients are raised when they get the keyboard input focus. If the value is False, the stacking of windows on the display is not changed when a window gets the keyboard input focus. The default value is True when the **keyboardFocusPolicy** is the explicit value and False when the **keyboardFocusPolicy** is the pointer value.
iconImage (class **IconImage**)	Specifies an icon image for a client (for example, **Mwm*myclock*iconImage**). The resource value is a path name for a bitmap file. The value of the (client-specific) **useClientIcon** resource is used to determine whether user-supplied icon images are used instead of client-supplied icon images. The default value is to display a built-in window manager icon image.
iconImageBackground (class **Background**)	Specifies the background color of the icon image that is displayed in the image part of an icon. The default value of this resource is the icon background color (that is, specified by **Mwm*background** or **Mwm*icon*background**).
iconImageBottomShadowColor (class **Foreground**)	Specifies the bottom shadow color of the icon image that is displayed in the image part of an icon. The default value of this resource is the icon bottom shadow color (that is, specified by **Mwm*icon*bottomShadowColor**).
iconImageBottomShadowPixmap (class **BottomShadowPixmap**)	Specifies the bottom shadow pixmap of the icon image that is displayed in the image part of an icon. The default value of this resource is the icon bottom shadow pixmap (that is, specified by **Mwm*icon*bottomShadowPixmap**).
iconImageForeground (class **Foreground**)	Specifies the foreground color of the icon image that is displayed in the image part of an icon. The default value of this resource varies depending on the icon background.
iconImageTopShadowColor (class **Background**)	Specifies the top shadow color of the icon image that is displayed in the image part of an icon. The default value of this resource is the icon top shadow color (that is, specified by **Mwm*icon*topShadowColor**).
iconImageTopShadowPixmap (class **TopShadowPixmap**)	Specifies the top shadow pixmap of the icon image that is displayed in the image part of an icon. The default value of this resource is the icon top shadow pixmap (that is, specified by **Mwm*icon*topShadowPixmap**).
matteBackground (class **Background**)	Specifies the background color of the matte when the **matteWidth** resource is a positive value. The default value of this resource is the client background color (that is, specified by **Mwm*background** or **Mwm*client*background**).

Client Decoration *(continued)*

Item	Description
matteBottomShadowColor (class **Foreground**)	Specifies the bottom shadow color of the matte when the **matteWidth** resource is a positive value. The default value of this resource is the client bottom shadow color (that is, specified by **Mwm*bottomShadowColor** or **Mwm*client*bottomShadowColor**).
matteBottomShadowPixmap (class **BottomShadowPixmap**)	Specifies the bottom shadow pixmap of the matte when the **matteWidth** resource is a positive value. The default value of this resource is the client bottom shadow pixmap (that is, specified by **Mwm*bottomShadowPixmap** or **Mwm*client*bottomShadowPixmap**).
matteForeground (class **Foreground**)	Specifies the foreground color of the matte when the **matteWidth** resource is a positive value. The default value of this resource is the client foreground color (that is, specified by **Mwm*foreground** or **Mwm*client*foreground**).
matteTopShadowColor (class **Background**)	Specifies the top shadow color of the matte when the **matteWidth** resource is a positive value. The default value of this resource is the client top shadow color (that is, specified by **Mwm*topShadowColor** or **Mwm*client*topShadowColor**).
matteTopShadowPixmap (class **TopShadowPixmap**)	Specifies the top shadow pixmap of the matte when the **matteWidth** resource is a positive value. The default value of this resource is the client top shadow pixmap (that is, specified by **Mwm*topShadowPixmap** or **Mwm*client*topShadowPixmap**).
matteWidth (class **MatteWidth**)	Specifies the width of the optional matte. The default value is 0, which effectively disables the matte.
maximumClientSize (class **MaximumClientSize**)	Indicates the client size to be used when an application is maximized. The resource value is specified *Width*x*Height*. The width and height are interpreted in the units that the client uses (for example, this is generally characters for terminal emulators). Alternately, the vertical or horizontal value can be specified to indicate the direction in which the client maximizes. If this resource is not specified, the maximum size from the **WM_NORMAL_HINTS** property is used if set. Otherwise, the default value is the size where the client window with window management borders fills the screen. When the maximum client size is not determined by the **maximumClientSize** resource, the **maximumMaximumSize** resource value is used as a constraint on the maximum size.
useClientIcon (class **UseClientIcon**)	Determines whether a client-supplied icon image takes precedence over a user-supplied icon image. The default value is False, giving the user-supplied icon image higher precedence than the client-supplied icon image.
usePPosition (class **UsePPosition**)	Specifies whether the window manager honors the program-specified position **PPosition** specified in the **WM_NORMAL_HINTS** property in the absence of a user-specified position. Setting this resource to On causes the **mwm** command to always honor the program-specified position. Setting this resource to Off causes the **mwm** command to always ignore the program-specified position. Setting this resource to the default value of nonzero causes the **mwm** command to honor program-specified positions other than (0,0).
windowMenu (class **WindowMenu**)	Indicates the name of the menu pane that is posted when the window menu is opened (usually by pressing button 1 on the Window Menu button on the client window frame). Menu panes are specified in the **mwm** resource description file. Window menus can be customized on a client class basis by specifying resources of the form **Mwm***_ClientNameOrClass_***windowMenu** (See mwm Resource Description File Syntax for more information.) The default value of this resource is DefaultWindowMenu.

Resource Description File

The **mwm** resource description file is a supplementary resource file that contains resource descriptions that are referred to by entries in the defaults files (**.Xdefaults**, **app-defaults/Mwm**). It contains descriptions of resources that are to be used by the **mwm** command and that cannot be easily encoded in the defaults files (a bitmap file is an analogous type of resource description file). A particular **mwm** resource description file can be selected using the **configFile** resource.

The following types of resources can be described in the **mwm** resource description file:

Resource Description

Item	Description
buttons	Window manager functions can be bound (associated) with button events.
keys	Window manager functions can be bound (associated) with key press events.
menus	Menu panes can be used for the window menu and other menus posted with key bindings and button bindings.

mwm Resource Description File Syntax

The **mwm** resource description file is a standard text file that contains items of information separated by blanks, tabs, and new-line characters. Blank lines are ignored. Items or characters can be quoted to avoid special interpretation (for example, the # (comment character) can be quoted to prevent it from being interpreted as the comment character). A quoted item can be contained in " " (double quotation marks). Single characters can be quoted by preceding them with the \ (backslash). All text from an unquoted # (comment character) to the end of the line is regarded as a comment and is not interpreted as part of a resource description. If an ! (exclamation mark) is the first character in a line, the line is regarded as a comment. If a line ends in a \ (backslash), the next line is considered a continuation of that line.

Window manager functions can be accessed with button and key bindings and with window manager menus. Functions are indicated as part of the specifications for button and key binding sets and for menu panes. The function specification has the following syntax:

```
Function = FunctionName [FunctionArguments]
FunctionName = Window Manager Function
FunctionArguments = {QuotedItem | UnquotedItem}
```

The following functions are supported. If a function is specified that is not one of the supported functions, it is interpreted by the **mwm** command as the **f.nop** function.

Resource Description File Syntax

Item	Description
f.beep	Causes a beep.
f.circle_down [*Icon* | *Window*]	Causes the window or icon that is on the top of the window stack to be put on the bottom of the window stack (so that it no longer obscures any other window or icon). This function affects only those windows and icons that obscure other windows and icons or that are obscured by other windows and icons. Secondary windows (that is, transient windows) are restacked with their associated primary window. Secondary windows always stay on top of the associated primary window and there can be no other primary windows between the secondary windows and their primary window. If an *Icon* function argument is specified, the function applies only to icons. If a *Window* function argument is specified, the function applies only to windows.

Item	Description
f.circle_up [*Icon* | *Window*]	Raises the window or icon on the bottom of the window stack (so that it is not obscured by any other windows). This function affects only those windows and icons that obscure other windows and icons or that are obscured by other windows and icons. Secondary windows (that is, transient windows) are restacked with their associated primary window. If an *Icon* function argument is specified, the function applies only to icons. If a *Window* function argument is specified, the function applies only to windows.
f.exec or **!**	Causes the command to be run (using the value of the **MWMSHELL** environment variable if it is set; otherwise, the value of the **SHELL** environment variable if it is set; otherwise, **/usr/bin/sh** is used). The ! notation can be used in place of the **f.exec** function name.
f.focus_color	Sets the colormap focus to a client window. If this function is done in a root context, the default colormap (set up by the X Window System client for the screen where MWM is running) is installed and there is no specific client window colormap focus. This function is treated as **f.nop** if **colormapFocusPolicy** is not set to the explicit value.
f.focus_key	Sets the keyboard input focus to a client window or icon. This function is treated as **f.nop** if **keyboardFocusPolicy** is not set to the explicit value or the function is run in a root context.
f.kill	Stops a client. If the **WM_DELETE_WINDOW** protocol is set up, the client is sent a client message event indicating that the client window needs to be deleted. If the **WM_SAVE_YOURSELF** protocol is set up and the **WM_DELETE_WINDOW** protocol is not set up, the client is sent a client message event indicating that the client needs to prepare to be stopped. If the client does not have the **WM_DELETE_WINDOW** or **WM_SAVE_YOURSELF** protocol set up, this function causes a client's X connection to be stopped (usually resulting in the end of the client). See the description of the **quitTimeout** resource.
f.lower [-*Client* | **within** | **freeFamily**]	Lowers a client window to the bottom of the window stack (where it obscures no other window). Secondary windows (that is, transient windows) are restacked with their associated primary window. The *Client* argument indicates the name or class of a client to lower. If the *Client* argument is not specified, the context, in which the function was started, indicates the window or icon to lower.

Item	Description
f.maximize	Causes a client window to be displayed with its maximum size.
f.menu	Associates a cascading (pull-right) menu with a menu pane entry or a menu with a button or key binding. The **menu_name** function argument identifies the menu to be used.
f.minimize	Causes a client window to be iconified (minimized). When a window is minimized and no icon box is used, its icon is placed on the bottom of the window stack (so that it obscures no other windows). If an icon box is used, the client's icon changes to its iconified form inside the icon box. Secondary windows (that is, transient windows) are minimized with their associated primary window. There is only one icon for a primary window and all its secondary windows.
f.move	Causes a client window to be interactively moved.
f.next_cmap	Installs the next colormap in the list of colormaps for the window with the colormap focus.
f.next_key [*Icon* | *Window* | *Transient*]	Sets the keyboard input focus to the next window or icon in the set of windows and icons managed by the window manager (the ordering of this set is based on the stacking of windows on the screen). This function is treated as **f.nop** if **keyboardFocusPolicy** is not the explicit value. The keyboard input focus is moved only to windows that do not have an associated secondary window that is application-modal. If the *Transient* argument is specified, transient (secondary) windows are crossed (otherwise, if only the *Window* argument is specified, traversal is done only to the last focused window in a transient group). If an *Icon* function argument is specified, the function applies only to icons. If a *Window* function argument is specified, the function applies only to windows.
f.nop	Does nothing. If a function is specified in a type of resource where it is not supported or is started in a context that does not apply, the function is treated as **f.nop**.
f.normalize	Causes a client window to be displayed with its normal size. Secondary windows (that is, transient windows) are placed in their normal state along with their associated primary window.
f.normalize_and_raise	Causes the corresponding client window to be displayed with its normal size and raised to the top of the window stack. Secondary windows (that is, transient windows) are placed in their normal state along with their associated primary window.

Item	Description
f.pack_icons	Causes icons to be packed into the icon grid. This function is used to relay out icons (based on the layout policy being used) on the root window or in the icon box.
f.pass_keys	Enables or disables (toggles) processing of key bindings for window manager functions. When it disables key binding processing, all keys are passed on to the window with the keyboard input focus and no window manager functions are started. If the **f.pass_keys** function is started with a key binding to disable key-binding processing, the same key binding can be used to enable key-binding processing.
f.post_wmenu	Posts the window menu. If a key is used to post the window menu and the Window Menu button is present, the window menu is automatically placed with its top-left corner at the bottom-left corner of the Window Menu button for the client window. If no Window Menu button is present, the window menu is placed at the top-left corner of the client window.
f.prev_cmap	Installs the previous colormap in the list of colormaps for the window with the colormap focus.
f.prev_key [*Icon* | *Window* | *Transient*]	Sets the keyboard input focus to the previous window or icon in the set of windows and icons managed by the window manager (the ordering of this set is based on the stacking of windows on the screen). This function is treated as **f.nop** if **keyboardFocusPolicy** is not the explicit value. The keyboard input focus is moved only to windows that do not have an associated secondary window that is application-modal. If the *Transient* argument is specified, transient (secondary) windows are crossed (otherwise, if only window is specified, traversal is done only to the last focused window in a transient group). If an *Icon* function argument is specified, the function applies only to icons. If a *Window* function argument is specified, the function applies only to windows.
f.quit_mwm	Stops the **mwm** command (but *not* the X Window System client).

Item	Description
f.raise [*-Client* | **within** | **freeFamily**]	Raises a client window to the top of the window stack (where it is obscured by no other window). Raises the secondary window (transient window or dialog box) within the client family. The arguments to this function are mutually exclusive. The *Client* argument indicates the name or class of a client to raise. If the *Client* argument is not specified, the context in which the function was started indicates the window or icon to raise. Specifying **within** raises the secondary window within the family but does not raise the client family in the global window stack. Specifying **freeFamily** raises the window to the top of its local family stack and raises the family to the top of the global window stack.
f.raise_lower [**within** | **freeFamily**]	Raises a primary window to the top of the window stack if it is partially obscured by another window; otherwise, it lowers the window to the bottom of the window stack. The arguments to this function are mutually exclusive. Specifying **within** raises a secondary window within the family (staying above the parent window), if it is partially obscured by another window in the application's family; otherwise, it lowers the window to the bottom of the family stack. It has no effect on the global stacking order. Specifying **freeFamily** raises the window to the top of its local family stack, if obscured by another window, and raises the family to the top of the global window stack; otherwise, it lowers the window to the bottom of its local family stack and lowers the family to the bottom of the global window stack.
f.refresh	Causes all windows to be redrawn.
f.refresh_win	Causes a client window to be redrawn.
f.resize	Causes a client window to be interactively resized.
f.restart	Causes the **mwm** command to be restarted (effectively stopped and restarted).
f.restore	Restores the previous state of an icon's associated window. If a maximized window is iconified, the **f.restore** function restores it to its maximized state. If a normalized window is iconified, the **f.restore** function restores it to its normalized state.

Item	Description
f.restore_and_raise	Restores the previous state of an icon's associated window and raises the window to the top of the window stack. If a maximized window is iconified, the **f.restore_and_raise** function restores it to its maximized state and raises it to the top of the window stack. If a normalized window is iconified, the **f.restore_and_raise** function restores it to its normalized state and raises it to the top of the window stack.
f.screen [**next** | **prev** | **back** | *ScreenNumber*]	Causes the pointer to warp to a specific screen number or to the next, previous, or last visited screen. The arguments to this function are mutually exclusive. The following arguments are available: ***ScreenNumber*** Indicates the screen number to which the pointer is warped. Screens are numbered starting from screen 0. **next** Warps the pointer to the next managed screen (skipping over any unmanaged screens). **prev** Warps the pointer to the previous managed screen (skipping over any unmanaged screens). **back** Warps the pointer to the last visited screen.
f.send_msg *MessageNumber*	Sends a client message of the **_MOTIF_WM_MESSAGES** type with the *MessageType* indicated by the *MessageNumber* function argument. The client message is sent only if *MessageNumber* is included in the client's **_MOTIF_WM_MESSAGES** property. A menu item label is unavailable if the menu item is used to perform the **f.send_msg** function of a message that is not included in the client's **_MOTIF_WM_MESSAGES** property.
f.separator	Causes a menu separator to be put in the menu pane at the specified location (the label is ignored).

Resource Description File Syntax *(continued)*

Item	Description
f.set_behavior	Causes the window manager to restart with the default behavior (if a custom behavior is configured) or revert to the custom behavior. By default this is bound to the Shift+Ctrl+Meta+! key sequence. The Meta+Shift+Ctrl+! key sequence switches (that is, toggles) between the default and custom behaviors. When the user switches to the default MWM behavior, a number of **mwm** resources assume their default values and the **mwm** command restarts. When the user switches back to the custom behavior, the resource values that were changed to default values are reset with the custom values and the **mwm** command restarts. When an **f.set_behavior** function is performed, the following user interaction occurs: 1. A system-modal dialog box is displayed prompting the user for confirmation of the **f.set_behavior** action. 2. The user can cancel the action at this point. 3. The window manager restarts. 4. The window manager applies the new (custom or default) configuration values. 5. Window manager components are mapped. When the default MWM behavior is being set, default resource values are applied and, if specified, client properties that control window manager behavior are applied. This includes the **_MOTIF_WM_HINTS** and **_MOTIF_WM_MENU** properties. These properties might alter default MWM behavior, but it is done in a way that is consistent for all users.
f.title	Inserts a title in the menu pane at the specified location.

Function Contexts

Each function may be constrained as to which resource types can specify the function (for example, menu pane) and also what context the function can be used in (for example, the function is done to the selected client window). The following are the function contexts:

Function contexts

Item	Description
root	No client window or icon is selected as an object for the function.
window	A client window is selected as an object for the function. This includes the window's title bar and frame. Some functions are applied only when the window is in its normalized state (for example, **f.maximize**) or its maximized state (for example, **f.normalize**).
icon	An icon is selected as an object for the function.

If a function's context is specified as **icon**|**window** and the function is started in an icon box, the function applies to the icon box, not to the icons inside.

If a function is specified in a type of resource where it is not supported or is started in a context that does not apply, the function is treated as **f.nop**. The following table indicates the resource types and function contexts in which window manager functions apply:

Function contexts		
Function	**Contexts**	**Resources**
f.beep	root, icon, window	button, key, menu
f.circle_down	root, icon, window	button, key, menu
f.circle_up	root, icon, window	button, key, menu
f.exec	root, icon, window	button, key, menu
f.focus_color	root, icon, window	button, key, menu
f.focus_key	root, icon, window	button, key, menu
f.kill	icon, window	button, key, menu
f.lower	icon, window	button, key, menu
f.maximize	icon, window (normal)	button, key, menu
f.menu	root, icon, window	button, key, menu
f.minimize	window	button, key, menu
f.move	icon, window	button, key, menu
f.next_cmap	root, icon, window	button, key, menu
f.next_key	root, icon, window	button, key, menu
f.nop	root, icon, window	button, key, menu

Normalize item description		
Item	**Description**	
f.normalize	icon, window (maximized)	button, key, menu
f.normalize_and_raise	icon, window	button, key, menu
f.pack_icons	root, icon, window	button, key, menu
f.pass_keys	root, icon, window	button, key, menu
f.post_wmenu	root, icon, window	button, key
f.prev_cmap	root, icon, window	button, key, menu
f.prev_key	root, icon, window	button, key, menu
f.quit_mwm	root, icon, window	button, key, menu (root only)
f.raise	icon, window	button, key, menu
f.raise_lower	icon, window	button, key, menu
f.refresh	root, icon, window	button, key, menu
f.refresh_win	window	button, key, menu
f.resize	window	button, key, menu

Normalize item description *(continued)*		
Item	**Description**	
f.restart	root, icon, window	button, key, menu
f.restore	icon, window	button, key, menu
f.restore_and_raise	icon, window	button, key, menu
f.screen	root, icon, window	button, key, menu
f.send_msg	icon, window	button, key, menu
f.separator	root, icon, window	menu
f.set_behavior	root, icon, window	button, key, menu
f.title	root, icon, window	menu

Window Manager Event Specification

Events are indicated as part of the specifications for button and key-binding sets and for menu panes.

Button events have the following syntax:

```
Button = [ModifierList]<ButtonEventName>
ModifierList = Modifier Name {ModifierName}
```

All modifiers specified are interpreted as being exclusive (this means that only the specified modifiers can be present when the button event occurs). Following is a list that indicates the values that can be used for the *ModifierName* parameter. The Alt key is frequently labeled Extend or Meta. Alt and Meta can be used interchangeably in event specification.

Window Manager Event Specification

Item	**Description**
Modifier	**Description**
Ctrl	Control key
Shift	Shift key
Alt	Alt or Meta key
Meta	Meta or Alt key
Lock	Lock key
Mod1	Modifier1
Mod2	Modifier2
Mod3	Modifier3
Mod4	Modifier4
Mod5	Modifier5

Following is a list that indicates the values that can be used for the *ButtonEventName* parameter.

Button Event Name descriptions

Button	**Description**
Btn1Down	Button 1 press

Button Event Name descriptions *(continued)*

Button	Description
Btn1Up	Button 1 release
Btn1Click	Button 1 press and release
Btn1Click2	Button 1 double click
Btn2Down	Button 2 press
Btn2Up	Button 2 release
Btn2Click	Button 2 press and release
Btn2Click2	Button 2 double click
Btn3Down	Button 3 press
Btn3Up	Button 3 release
Btn3Click	Button 3 press and release
Btn3Click2	Button 3 double click
Btn4Down	Button 4 press
Btn4Up	Button 4 release
Btn4Click	Button 4 press and release
Btn4Click2	Button 4 double click
Btn5Down	Button 5 press
Btn5Up	Button 5 release
Btn5Click	Button 5 press and release
Btn5Click2	Button 5 double click.

Key events that are used by the window manager for menu mnemonics and for binding to window manager functions are single key presses; key releases are ignored. Key events have the following syntax:

```
Key = [ModifierList] <Key> KeyName
ModifierList = ModifierName {ModifierName}
```

All modifiers specified are interpreted as being exclusive (this means that only the specified modifiers can be present when the key event occurs). Modifiers for keys are the same as those that apply to buttons. The *KeyName* parameter is an X11 keysym name. Key symbol names can be found in the **keysymdef.h** file (remove the *XK_* prefix).

The key symbol names will be resolved to a single specific key code by the Window Manager during startup and will not change unless the Window Manager is restarted.

Button Bindings

The **buttonBindings** resource value is the name of a set of button bindings that are used to configure window manager behavior. A window manager function can be used when a button press occurs with the pointer over a framed client window, an icon, or the root window. The context for indicating where the button press applies is also the context for starting the window manager function when the button press is done (significant for functions that are context-sensitive).

Following is the button binding syntax:

```
Buttons BindingsSetName
        {
```

```
Button Context Function
Button Context Function
                .
                .
Button Context Function
        }
```

Following is the syntax for the context specification:

```
Context = Object[|Context]
Object = root | icon | window | title | frame | border | app
```

The *Context* specification indicates where the pointer must be for the button binding to be effective. For example, a context of **window** indicates that the pointer must be over a client window or window management frame for the button binding to be effective. The **frame** context is for the window management frame around a client window (including the border and title bar), the **border** context is for the border part of the window management frame (not including the title bar), the **title** context is for the title area of the window management frame, and the **app** context is for the application window (not including the window management frame).

If an **f.nop** function is specified for a button binding, the button binding is not done.

Key Bindings

The **keyBindings** resource value is the name of a set of key bindings that are used to configure window manager behavior. A window manager function can be done when a particular key is pressed. The context in which the key binding applies is indicated in the key binding specification. The valid contexts are the same as those that apply to button bindings.

Following is the key binding syntax:

```
Keys BindingsSetName
{
Key Context Function
Key Context Function
                .
                .
Key Context Function
}
```

If an **f.nop** function is specified for a key binding, the key binding is not done. If an **f.post_wmenu** or **f.menu** function is bound to a key, the **mwm** command automatically uses the same key for removing the menu from the screen after it is open.

The *Context* specification syntax is the same as for button bindings. For key bindings, the **frame**, **title**, **border**, and **app** contexts are equivalent to the **window** context. The context for a key event is the window or icon that has the keyboard input focus (**root** if no window or icon has the keyboard input focus).

Menu Panes

Menus can be opened using the **f.post_wmenu** and **f.menu** window manager functions. The context for window manager functions that are done from a menu is **root**, **icon**, or **window**, depending on how the menu is opened. In the case of the window menu or menus opened with a key binding, the location of the keyboard input focus indicates the context. For menus opened using a button binding, the context of the button binding is the context of the menu.

Following is the menu pane specification syntax:

```
Menu MenuName
{
Label [Mnemonic] [Accelerator] Function
Label [Mnemonic] [Accelerator] Function
```

```
                .
                .
Label [Mnemonic] [Accelerator] Function
}
```

Each line in the *Menu* specification identifies the label for a menu item and the function to be completed if the menu item is selected. Optionally, a menu button mnemonic and a menu button keyboard accelerator can be specified. Mnemonics are functional only when the menu is posted and keyboard traversal applies.

The label can be a string or a bitmap file. The *Label* specification has the following syntax:

```
Label = Text | BitmapFile
BitmapFile = @FileName
Text = QuotedItem | UnquotedItem
```

The string encoding for labels must be compatible with the menu font that is used. Labels are not available for menu items that use the **f.nop** function, an invalid function, or a function that does not apply in the current context.

A *Mnemonic* specification has the following syntax:

```
Mnemonic = _Character
```

The first matching *Character* in the label is underlined. If there is no matching *Character* in the label, no mnemonic is registered with the window manager for that label. Although the *Character* must exactly match a character in the label, the mnemonic does not perform if any modifier (such as the Shift key) is pressed with the character key.

The *Accelerator* specification is a key event specification with the same syntax that is used for key bindings to window manager functions.

Environment

The **mwm** command does the following:

- Uses the **HOME** environment variable to specify the user's home directory.
- Uses the **LANG** environment variable to specify the user's choice of language for the **mwm** message catalog and the **mwm** resource description file.
- Uses the **XFILESEARCHPATH**, **XUSERFILESEARCHPATH**, **XAPPLRESDIR**, **XENVIRONMENT**, **LANG**, and **HOME** environment variables to determine search paths for resource defaults files. The **mwm** command can also use the **XBMLANGPATH** environment variable to search for bitmap files.
- Reads the **$HOME/.motifbind** file, if it exists, to install a virtual key bindings property on the root window.
- Uses the **MWMSHELL** environment variable (or **SHELL** if **MWMSHELL** is not set) to specify the shell to use when running commands through the **f.exec** function.

Exit Status

This command returns the following exit values:

Exit Status

Item	Description
0	Indicates successful completion.
>1	Indicates an error occurred.

Files

/usr/lib/X11/$LANG/system.mwmrc

/usr/lib/X11/system.mwmrc

/usr/lib/X11/app-defaults/Mwm

$HOME/Mwm

$HOME/.Xdefaults

$HOME/$LANG/.mwmrc

$HOME/.mwmrc

$HOME/.motifbind

n

The following AIX commands begin with the letter *n*.

named Daemon

Purpose

Provides the server function for the Domain Name Protocol.

Syntax

Refer to the syntax for either the **named8** or the **named9** daemon.

Description

AIX 7.1 supports only BIND version 9. By default, named links to nsupdate to nsupdate4, named-xfer to named-xfer4. To use a different version of named, you must relink the symbolic links accordingly for the named and named-xfer daemons.

For example, to use named8:

```
ln -fs /usr/sbin/named8 /usr/sbin/named
ln -fs /usr/sbin/named8-xfer /usr/sbin/named-xfer
```

nsupdate4 can be used with named8, but nsupdate9 must be used with named9 because the security process is different. It does not matter what named-xfer is linked to when using named9 because the daemon does not use it.

Files

Item	Description
/usr/sbin/named	Contains the **named** daemon.
/usr/sbin/named9	Contains the **named9** daemon.
/etc/resolv.conf	Specifies the use of domain name services.
/etc/services	Defines socket service assignments.
/usr/samples/tcpip/named.boot	Contains the sample **named.boot** file with directions for its use.
/usr/samples/tcpip/named.data	Contains the sample DOMAIN data file with directions for its use.
/usr/samples/tcpip/hosts.awk	Contains the sample **awk** script for converting an **/etc/hosts** file to an **/etc/named.rev** file. This file also contains directions for its use.
/usr/samples/tcpip/named.dynamic	Contains a dynamic database setup.

named-checkconf Command

Purpose

Syntax checking tool of a named configuration file.

Syntax

named-checkconf [**-v**] [**-j**] [**-t** *directory*] *filename* [**-z**]

Description

The **named-checkconf** command checks the syntax, but not the semantics, of a named configuration file.

Flags

Item	Description
-j	Reads the journal if it exists when loading a zonefile.
-t *directory*	Changes the present directory to the directory specified so that included directives in the configuration file are processed.
-v	Prints the version of the **named-checkconf** program and exits.
-z	Performs a check and loads the master zone files found in the **named.conf** file.
filename	Specifies the name of the configuration file to be checked. If not specified, the default value is **/etc/named**.

Exit Status

Item	Description
0	Indicates a successful completion.
1	Indicates errors.

named-checkzone, named-compilezone Commands

Purpose

Zone file validity checking or converting tool of a named configuration file.

Syntax

named-checkzone [**-d**] [**-j**] [**-q**] [**-v**] [**-c** *class*] [**-f** *format*] [**-F** *format*] [**-i** *mode*] [**-k** *mode*] [**-m** *mode*] [**-M** *mode*] [**-n** *mode*] [**-o** *filename*] [**-s** *style*] [**-S** *mode*] [**-t** *directory*] [**-w** *directory*] [**-D**] [**-W** *mode*] *zonename filename*

named-compilezone [**-d**] [**-j**] [**-q**] [**-v**] [**-c** *class*] [**-f** *format*] [**-F** *format*] [**-i** *mode*] [**-k** *mode*] [**-m** *mode*] [**-n** *mode*] [**-o** *filename*] [**-s** *style*] [**-t** *directory*] [**-w** *directory*] [**-D**] [**-W** *mode*] *zonename filename*

Description

The **named-checkzone** command checks the syntax and integrity of a zone file. It performs the same checks as the **named** daemon does when loading a zone. This makes the **named-checkzone** command useful for checking zone files before configuring them into a name server.

The **named-compilezone** command is similar to the **named-checkzone** command, but it always dumps the zone contents to a specified file in a specified format. Additionally, it applies stricter check levels by default, since the dump output will be used as an actual zone file loaded by the named daemon. When manually specified otherwise, the check levels must at least be as strict as those specified in the named configuration file.

Flags

Item	Description
-c *class*	Specifies the class of the zone. If not specified, the class is set to "IN" by default.
-d	Enables debugging.
-D	Dumps zone file in canonical format. This is always enabled for the **named-compilezone** command.
-i *mode*	Performs post load zone integrity checks. The *mode* parameter can be one of the following values: **full** Checks if MX records, SRV records, and delegation NS records refer to A or AAAA record (both in-zone and out-of-zone host names). It also checks if glue addresses records in the zone match those advertised by the child. **full-sibling** Disables sibling glue checks but is otherwise the same as mode **full**. **local** Only checks if MX records, SRV records, and delegation NS records refer to in-zone host names or if some required glue exists, that is when the name server is in a child zone. **local-sibling** Disables sibling glue checks but is otherwise the same as mode **local**. **none** Disables the checks.
-j	Reads the journal if it exists when loading the zone file.
-f *format*	Specifies the format of the zone file. Possible formats are "text" (default) and "raw".
-F *format*	Specifies the format of the output file specified. Possible formats are "text" (default) and "raw". This flag does not cause any effects unless it dumps the zone contents.
-k *mode*	Performs "check-names" checks with the specified failure mode. Possible modes are "fail", "warn" (default) and "ignore".
-m *mode*	Specifies whether MX records must be checked to see if they are addresses. Possible modes are "fail", "warn" (default) and "ignore".
-M *mode*	Checks if a MX record refers to a CNAME. Possible modes are "fail", "warn" (default) and "ignore".
-n *mode*	Specifies whether NS records must be checked to see if they are addresses. Possible modes are "fail", "warn" (default) and "ignore".
-o *filename*	Writes zone output to the file specified by the *filename* value.
-q	Indicates quiet mode (exits code only).
-s *style*	Specifies the style of the dumped zone file. Possible styles are "full" (default) and "relative". The "full" format is most suitable for processing automatically by a separate script. On the other hand, the "relative" format is more human-readable and is thus suitable for editing by hand. This flag does not cause any effects unless it dumps the zone contents. It also does not have any meaning if the output format is not text.

Item	Description
-S *mode*	Checks if a SRV record refers to a CNAME. Possible modes are "fail", "warn" (default) and "ignore".
-t *directory*	Changes the directory to the *directory* so that included directives in the configuration file are processed.
-v	Prints the version of the **named-checkzone** command and exits.
-w *directory*	Changes the current directory to the *directory* so that relative file names in master file $INCLUDE directives work. This is similar to the directory clause in the **named.conf** file.
-W *mode*	Specifies whether to check for non-terminal wildcards. Non-terminal wildcards are almost always the result of a failure to understand the wildcard matching algorithm (RFC 1034). Possible modes are "warn" (default) and "ignore".
zonename	Specifies the domain name of the zone being checked.
filename	Specifies the name of the zone file.

Exit Status

Item	Description
0	Indicates a successful completion.
1	Indicates errors.

named8 Daemon

Purpose

Provides the server function for the Domain Name Protocol.

Syntax

/usr/sbin/named8 [**-d** *DebugLevel*] [**-p** *PortNumber*] [**-c** *ConfFile*] [**-w** *WorkingDirectory*] [**-t** *RootDirectory*] [**-q**] [**-r**] [**-f**]

Description

The **/usr/sbin/named8** daemon is the server for the Domain Name Protocol (DOMAIN). The **named8** daemon runs on name server hosts and controls the domain-name resolution function.

Selection of which name server daemon to use is controlled by the **/usr/sbin/named** and **/usr/sbin/named-xfer** symbolic links.

Note: The **named8** daemon can be controlled using the System Resource Controller (SRC) or the System Management Interface Tool (SMIT). Use the **rc.tcpip** file to start the daemon with each system startup.

The **named8** daemon listens for name-server requests generated by resolver routines running on foreign hosts. The daemon listens to the socket defined in the **/etc/services** file; the entry in the **/etc/services** file begins with `domain`. However, this socket assignment can be overridden using the **-p** flag on the command line.

Note: The **/etc/resolv.conf** file tells the local kernel and resolver routines to use the DOMAIN protocol. The **/etc/resolv.conf** file must exist and contain either the local host's address or the loopback address (127.0.0.1) to use the **named8** daemon on the DOMAIN name server host. If the **/etc/resolv.conf** file does not exist, the local kernel and resolver routines use the **/etc/hosts** database. When this occurs, the **named8** daemon does not function properly.

Manipulating the named8 Daemon with the System Resource Controller

The **named8** daemon is a subsystem controlled by the System Resource Controller (SRC). The **named8** daemon is a member of the **tcpip** system group. This daemon is disabled by default and can be manipulated by the following SRC commands:

Item	Description
startsrc	Starts a subsystem, group of subsystems,or a subserver.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
refresh	Causes the **named8** daemon to reread the **/etc/named.conf** file. Depending on the contents of the file, the **refresh** command may or may not reload the listed databases.
traceson	Enables tracing of a subsystem, group of subsystems, or a subserver.
tracesoff	Disables tracing of a subsystem, group of subsystems, or a subserver.
lssrc	Gets the status of a subsystem, group of subsystems, or a subserver.

Flags

Item	Description
-b | **-c***ConfFile*	Specifies an alternate configuration file.
-d*DebugLevel*	Provides a debugging option. The **-d** flag causes the **named8** daemon to write debugging information to a file named by default **/var/tmp/named.run**. The *DebugLevel* variable determines the level of messages printed, with valid levels from 1 to 11, where level 11 supplies the most information.
-p*PortNumber*	Reassigns the Internet socket where the **named8** daemon listens for DOMAIN requests. If this variable is not specified, the **named8** daemon listens to the socket defined in the **/etc/services** file; the entry in the **/etc/services** file begins with `domain`.
-w*WorkingDirectory*	Changes the working directory of the **named8** daemon. This option can be specified or overridden by the "directory" configuration option.
-t*RootDirectory*	Specifies a directory to be the new root directory for the **named8** daemon using the **chroot** command.
-q	Enables logging of all name service queries.
-r	Disables the server's ability to recurse and resolve queries outside of the server's local databases.
-f	Indicates to run the name server daemon in the foreground rather than becoming a background job.

Signals

The following signals have the specified effect when sent to the **named8** daemon process using the **kill** command:

Item	Description
SIGHUP	The **named8** daemon rereads the **/etc/named.conf**file. Depending on the contents of the file, the **SIGHUP** signal may or may not reload the listed databases.
SIGILL	Dumps statistics data into **named.stats**. Statistics data is appended to the file.

Item	Description
SIGINT	The **named8** daemon dumps the current database to a file named **/var/tmp/named_dump.db**. In the dump file, names with the label **name error** indicate negative cache entries. This happens when a server responds that the specified domain name does not exist. Names labeled as **data error** also indicate negative cache entries. This happens when a server responds that there are no records of the specified type for the (valid) domain name.
SIGUSR1	The **named8** daemon turns on debugging; each subsequent **SIGUSR1** signal increments the debugging level. The debugging information is written to the **/var/tmp/named.run** file.
SIGUSR2	The **named8** daemon turns off debugging.

Examples

1. To start the **named8** daemon normally, enter the following:

   ```
   startsrc -s named
   ```

   This command starts the daemon. You can use this command in the **rc.tcpip** file or on the command line. The **-s** flag specifies that the subsystem that follows is to be started. The process ID of the **named8** daemon is stored in the **/etc/named.pid** file upon startup.
2. To stop the **named8** daemon normally, enter:

   ```
   stopsrc -s named
   ```

   This command stops the daemon. The **-s** flag specifies that the subsystem that follows is to be stopped.
3. To get short status from the **named8** daemon, enter:

   ```
   lssrc -s named
   ```

   This command returns the name of the daemon, the process ID of the daemon, and the state of the daemon (active or inactive).
4. To enable debugging for the **named8** daemon, enter:

   ```
   traceson -s named
   ```

   OR

   ```
   kill -30 `cat /etc/named.pid`
   ```

   The **named8** daemon turns on debugging in response to either of these commands; each subsequent command increments the debugging level. The debugging information is written to the **/var/tmp/named.run** file.
5. To turn off debugging for the **named8** daemon, enter:

   ```
   tracesoff
   ```

   OR

   ```
   kill -31 `cat /etc/named.pid`
   ```

   Either of these commands immediately turns off all debugging.

6. To start the **named8** daemon at the highest debugging level using the **startsrc** command, enter the following:

```
startsrc -s named -a -d11
```

This command writes debugging messages to the **/var/tmp/named.run** file.

Files

Item	Description
/usr/sbin/named8	Contains the **named8** daemon.
/usr/sbin/named8-xfer	Provides the functionality of the worker name server's inbound zone transfer.
/etc/named.conf	Specifies the configuration of the **named8** daemon including some basic behaviors, logging options, and locations of the local databases.
/etc/resolv.conf	Specifies the use of domain name services.
/etc/rc.tcpip	Initializes daemons at each system restart.
/etc/named.pid	Stores process ID.
/etc/services	Defines socket service assignments.
/usr/samples/tcpip/named.conf	Contains the sample **named.conf** file with directions for its use.
/usr/samples/tcpip/named.data	Contains the sample DOMAIN data file with directions for its use.
/usr/samples/tcpip/hosts.awk	Contains the sample **awk** script for converting an **/etc/hosts** file to an **/etc/named.data** file. This file also contains directions for its use.
/usr/samples/tcpip/addrs.awk	Contains the sample **awk** script for converting an **/etc/hosts** file to an **/etc/named.rev** file. This file also contains directions for its use.

named9 Daemon

Purpose

Internet domain name server.

Syntax

named9 [**-4**] [**-6**] [**-c** *config-file*] [**-d** *debug-level*] [**-f**] [**-g**] [**-n** *#cpus*] [**-p** *port*] [**-s**] [**-t** *directory*] [**-u** *user*] [**-v**] [**-x** *cache-file*]

Description

named9 is a Domain Name System (DNS) server, part of the BIND 9 distribution from ISC. For more information on the DNS, see RFCs 1033, 1034, and 1035. When invoked without arguments, the **named9** daemon reads the default configuration file **/etc/named.conf**, reads any initial data, and listens for queries.

Flags

Item	Description
-4	Uses IPv4 only even if the host machine is capable of IPv6. The **-4** and **-6** options are mutually exclusive.
-6	Uses IPv6 only even if the host machine is capable of IPv4. The **-4** and **-6** options are mutually exclusive.
-c *config-file*	Uses *config-file* as the configuration file instead of the default, **/etc/named.conf**. To ensure that reloading the configuration file continues to work after the server has changed its working directory due to a possible directory option in the configuration file, the *config-file* value must be an absolute path name.
-d *debug-level*	Sets the daemon's debug level to the *debug-level* value. Debugging traces from the **named9** daemon become more verbose as the debug level increases.
-f	Runs the server in the foreground.
-g	Runs the server in the foreground and forces all logging to the standard error **stderr**.
-n *#cpus*	Creates *#cpus* worker threads to take advantage of multiple CPUs. If not specified, the **named9** daemon tries to determine the number of CPUs present and creates one thread per CPU. If it is unable to determine the number of CPUs, the **named9** daemon creates a single worker thread.
-p *port*	Listens for queries on port *port*. If not specified, the default is port 53.
-s	Writes memory usage statistics to the standard output **stdout** on exit.
-t *directory*	Changes the present directory to the directory specified after processing the command line arguments, but before reading the configuration file. **Warning:** You must use this option in conjunction with the **-u** option, as changing the present directory of a process running as root does not enhance security on most systems.
-u *user*	Sets the process user ID to the user specified after completing privileged operations, such as creating sockets that listen on privileged ports.
-v	Reports the version number and exit.
-x *cache-file*	Loads data from *cache-file* into the cache of the default view.

Signals

In routine operation, you cannot use signals to control the name server; you must use the **rndc** command.

Item	Description
SIGHUP	Forces a reload of the server.
SIGINT, SIGTERM	Shuts down the server.

The result of sending any other signals to the server is undefined.

Configuration

For a complete description of the **named9** configuration file, refer to the BIND 9 Administrator Reference Manual.

Files

Item	Description
/usr/sbin/named9	Contains the **named9** daemon.
/etc/named.conf	The default configuration file.
/etc/named.pid	The default process-id file.

namerslv Command

Purpose

Directly manipulates domain name server entries for local resolver routines in the system configuration database.

Syntax

To Add a Name Server Entry

namerslv -a { **-i** *IPAddress* | **-D** *DomainName*| **-S** *SearchList*}

To Delete a Name Server Entry

namerslv -d { **-i** *IPAddress* | **-n** | **-l**}

To Delete All Name Server Entries

namerslv -X [**-I**]

To Change a Name Server Entry

namerslv -c *DomainName*

To Display a Name Server Entry

namerslv -s [**-I** | **-n** | **-l**] [**-Z**]

To Create the Configuration Database File

namerslv -b [**-i** *IPAddress* [**-D** *DomainName*] [**-S** *SearchList*]]

To Rename the Configuration Database File

namerslv -E *FileName*

To Move the Configuration Database File to Prevent Name Server Use

namerslv -e

To Import a File into the Configuration Database File

namerslv -B*FileName*

To Change a Search List Entry

namerslv -C*Search List*

Description

The **namerslv** low-level command adds or deletes domain name server entries for local resolver routines in the system configuration database. By default, the system configuration database is contained in the **/etc/resolv.conf** file is moved to the file specified by the *FileName* variable.

Item	Description
-a	Adds an entry to the system configuration database. The **-a** flag must be used with either the **-i** or **-D** flag.
-B *FileName*	Restores the **/etc/resolv.conf** file from the file specified by the **FileName** variable.
-b	Creates the system configuration database, using the **/etc/resolv.conf.sv** file. If the **/etc/resolv.conf.sv** file does not exist, an error is returned. **Note:** The **/etc/resolv.conf.sv** file is not shipped with the system. You have to create the file before the **-b** flag will work.
-C	Changes the search list in the **/etc/resolv.conf** file.
-c *DomainName*	Changes the domain name in the system configuration database.
-D	Indicates that the command deals with the domain name entry.
-d	Deletes an entry in the system configuration database. It must be used with the **-i IPAddress** flag or the **-n** flag. The **-i** flag deletes a name server entry. The **-n** flag deletes the domain name entry.
-E *FileName*	Renames the system configuration database file, so you can stop using a name server. The **/etc/resolv.conf** file is moved to the file specified by the *FileName* variable.
-e	Moves the **/etc/resolv.conf** file to the **/etc/resolv.conf.sv** file, preventing use of a name server.
-I	(Uppercase i) Specifies that the **-s** flag or **-X** flag should print all name server entries.
-i *IPAddress*	Indicates that the command deals with a name server entry. Use dotted decimal format for the given IP address.
-l	(Lowercase L) Specifies that the operation is on the search list. Use this flag with the **-d** and **-s** flag.
-n	Specifies that the operation is on the domain name. Use this flag with the **-d** flag and the **-s** flag.
-S *SearchList*	Changes the search list in the system configuration database.
-s	Shows all domain and name server entries in the configuration system database. If you use the **-i** flag, the **namerslv** command shows all name server entries. If you use the **-n** flag, the **namerslv** command shows the domain name entry found in the database.
-X	Deletes all entries in the database. Use the **-I** flag with this flag to delete all name server entries.

Item	Description
-Z	Generates the output of the query in colon format. This flag is used when the **namerslv** command is called from the SMIT usability interface.

Examples

1. To add a domain entry with a domain name of `abc.aus.century.com`, type:

   ```
   namerslv  -a  -D abc.aus.century.com
   ```

2. To change the `abc.aus.century.com` domain entry to the domain name `xyz.aus.century.com`, type:

   ```
   namerslv xyz.aus.century.com
   ```

3. To add a name server entry with IP address `192.9.201.1`, type:

   ```
   namerslv  -a  -i 192.9.201.1
   ```

4. To show all system configuration database entries related to domain name server information used by local resolver routines, type:

   ```
   namerslv  -s
   ```

   The output is given in the following format:

   ```
   domain xyz.aus.century.com
       name server 192.9.201.1
   ```

5. To rename the **/etc/resolv.conf** file to stop using the name server and specify the new file name, `/etc/resolv.back`, type:

   ```
   namerslv  -E /etc/resolv.back
   ```

Files

Item	Description
/usr/sbin/namerslv	Contains the **namerslv** command.
/etc/html	

ncheck Command

Purpose

Generates path names from i-node numbers.

Syntax

ncheck [[[**-a**] [**-i** *InNumber ...*]] | [**-s**]] [**-o** *Options*] [*FileSystem*]

Description

The **ncheck** command displays the i-node number and path names for filesystem files. It uses question marks (??) displayed in the path to indicate a component that could not be found. Path names displayed with ... (ellipses) at the beginning indicate either a loop or a path name of greater than 10 entries.
The **ncheck** command uses a simple hashing alogrithm to reconstruct the path names that it displays. Because of this, it is restricted to filesystems with less than 50,000 directory entries.

Flags

Item	Description
-a	Lists the . (dot) and .. (dot dot) file names.
-i *InNumber*	Lists only the file or files specified by the *InNumber* parameter.
-o *Options*	Specifies a comma-separated list of implementation-specific options for a virtual file system. The following options are specific to the enhanced journaled file system (JFS2 **-o snapshot=***snapName*): Specifies the name of the internal snapshot subject to the **ncheck** command. The file system owning the snapshot must be mounted.
-s	Lists only special files and files with set-user-ID mode.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list the i-node number and path name of each file in the default file systems, enter:

   ```
   ncheck
   ```

2. To list all the files in a specified file system, enter:

   ```
   ncheck  -a /
   ```

   This lists the i-node number and path name of each file in the **/** (root) file system, including the .(dot) and .. (dot dot) entries in each directory.

3. To list the name of a file when you know its i-node number, enter:

   ```
   ncheck  -i 690 357 280 /tmp
   ```

   This lists the i-node number and path name for every file in the **/tmp** file system with i-node numbers of 690, 357, or 280. If a file has more than one link, all of its path names are listed.

4. To list special and set-user-ID files, enter:

   ```
   ncheck  -s /
   ```

   This lists the i-node and path name for every file in the **/** (root) file system that is a special file (also called a device file) or that has set-user-ID mode enabled.

nddctl Command

Purpose

Issues commands to network device drivers (NDDs).

Syntax

nddctl { **-r** } *Device*

Description

The **nddctl** command allows the user to control an NDD device at runtime (that is, without having to reconfigure the device driver, which usually entails disruption to the network connection).

Flags

Item	Description
-r	Forces the NDD device to renegotiate its link attributes (speed and duplexity) at runtime. **Note:** Forcing link renegotiation entails resetting the device; this might cause a loss of network connectivity, lasting a few seconds, while the device re-initializes itself.

Parameters

Item	Description
Device	Specifies the NDD device on which to perform the specified command.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Examples

1. To force the device `ent0` to renegotiate its link attributes at runtime, type:

```
nddctl -r ent0
```

Location

/usr/sbin

ndp Command

Purpose

IPv6 neighbor discovery display and control.

Syntax

ndp [**-n**] *hostname*

ndp [**-n**] **-a**

ndp [**-d**] *hostname* | *IpAddress*

ndp [**-i** *interface_index*] **-s** *hostname addr* [**temp**]

Description

The **ndp** program displays and modifies the IPv6-to-Ethernet, IPv6-to-TokenRing, or IPv6-to-InfiniBand address translation tables used by the IPv6 neighbor discovery protocol.

With no flags, the program displays the current **ndp** entry for *hostname*. The host may be specified by name or by number, using IPv6 textual notation.

Flags

Item	Description
- a	Displays all of the current **ndp** entries.
- d	Lets a super-user delete an entry for the host called *hostname* with the **-d** flag.
- i *interface_index*	Specifies the index of the interface to use when an **ndp** entry is added with the **-s** flag (useful with the local-link interface).
- n	Shows network addresses as numbers (normally **ndp** attempts to display addresses symbolically).
- s *hostname addr*	Creates an **ndp** entry for *hostname* with the Hardware address *addr*. The Hardware address is given as six hex bytes separated by colons. The entry is permanent unless the **temp** is specified in the command.

Examples

This is an example output from the **- a** flag:

```
# ndp -a
e-crankv6 (::903:9182) at link#2 0:20:af:db:b8:cf
e-crankv6-11 (fe80:0:100::20:afdb:b8cf) at link#2 0:20:af:db:b8:cf
e-crankv6-11 (fe80::2:c903:1:1e85) at link#5 SQP:0xe SLID0x49 DQP:0x48 DLID:0xf
0:48:fe80::2:c903:1:1e85 [InfiniBand]
# ndp -d e-crankv6-11
e-crankv6-11 (fe80:0:100::20:afdb:b8cf) deleted
# ndp -d fe80::2:c903:1:1e85
```

ndpd-host Daemon

Purpose

Neighbor Discovery Protocol (NDP) daemon for a host.

Syntax

ndpd-host [-d] [-v] [-t] [-c *conffile*][-r [*ValidLifetime PreferredLifetime*]] [-g]

Description

The **ndpd-host** command manages the Neighbor Discovery Protocol (NDP) for nonkernel activities, such as Router Discovery, Prefix Discovery, Parameter Discovery, and Redirects. The **ndpd-host** command handles the default route, which includes the default router, the default interface, and the default interface address. However, the **ndpd-host** command does not overwrite the static default routes that are set on the host. When the daemon is stopped, the daemon cleans up the prefix addresses and the routes that are created during its lifetime.

Interfaces

The **ndpd-host** command knows about IEEE and CTI point to point interfaces. The **ndpd-host** command exchanges packets on all the known interfaces UP with a Link-Local Address. Any change of status of an interface is detected. If an interface goes down or loses its Link-Local address, the NDP processing is stopped on this interface. If an interface goes up, the NDP processing is started.

The IEEE interfaces are configured by using the **autoconf6** command. The PPP interfaces are configured by using the **pppd** daemon. The token negotiation defines the Link-Local addresses. To send the Router Advertisements over a CTI configured tunnel, it must have local and distant Link-Local addresses.

ndpd-host can generate Temporary Addresses as per RFC 4941. You can enable or disable temporary address generation for a particular prefix or interface by configuring the daemon in the `tempaddr.conf` file format. You can set the default preferred and valid lifetimes of Temporary Addresses by using the `-r` option.

Note: For all the up point to point interfaces, **ndpd-host** sets a local route through the `lo0` for local addresses.

Flags

Item	Description
`-c`*conffile*	Specifies the SEND configuration file. By default, the configuration file is the `/etc/ndpd/ndpdh.cnf` file. To enable the SEND option, you must install the `clic.rte` fileset and OpenSSL.
`-d`	Enables debugging (exceptional conditions and dump).
`-g`	Allows the **ndpd-host** command to retain all the static global IPv6 address during initialization.
`-r` [*ValidLifetime PreferredLifetime*]	Enables Temporary Address generation. Along with `-r` flag, user can optionally specify default valid and preferred lifetimes for Temporary Addresses generated. By default, Temporary addresses are not generated, if this flag is not given.
`-t`	Adds a time stamp in each log.
`-v`	Logs all interesting events (`daemon.info` and console).

Signals

Item	Description
SIGUSR1	Turns on verbose.
SIGUSR2	Turns off verbose.
SIGINT	Dumps the current state of **ndpd-host** to **syslog** or **stdout**.
SIGTERM	Cleans up **ndpd-host** and exits.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
`/etc/ndpd/ndpdh.cnf`	Specifies the SEND file locations.
`/etc/ndpd/cgaparams.sec`	Specifies the configuration for each interface by using the SEND option.
`/etc/ndpd/sendh_anchor`	Specifies the trusted anchor values necessary for the SEND option.

Item	Description
`/etc/ndpd/tempaddr.conf`	Specifies whether the generation of the Temporary Address for the router prefixes must be denied or allowed. The contents of the file are read only when **ndpd-host** is started with the `-r` flag.

ndpd-router Daemon

Purpose

NDP and RIPng daemon for a router.

Syntax

ndpd-router [**-r**] [**-p**] [**-M**] [**-O**] [**-s**] [**-q**] [**-g**] [**-n**] [**-R**] [**-S**] [**-d**] [**-t**] [**-v**] [**-H**] [**-m**] [**-u** *port*] [**-D** *max*[*min*[**/***life*]]] [**-P** [*invlife*]**/**[*deplife*]] [**-T** [*reachtim*]**/**[*retrans*]**/**[*hlim*]] [**-e** [**off** | **compatible** | **only**]]

Description

The **ndpd-router** daemon manages the Neighbor Discovery Protocol (NDP) for non-kernel activities. It receives Router Solicitations and sends Router Advertisements. It can also exchange routing information using the RIPng protocol.

The **/etc/gateway6** file provides options for **ndpd-router**. This file can be modified while the program is running. The changes are checked before any emission or reception of message, or on reception of the HUP signal. The file contains directives, one by line (with # as comment). All the IPv6 addresses and prefixes in the file must be in numeric form. No symbolic name is allowed. Except for the gateway directive, each line begins with a keyword and is made of options of the form *key=argument*.

Interfaces

The **ndpd-router** daemon knows about IEEE and CTI point to point interfaces. The **ndpd-router** daemon exchanges packets on all the known interfaces UP with a Link-Local Address. Any change of status of an interface is detected. If an interface goes down or loses its Link-Local address, the NDP and RIPng processing is stopped on this interface. If an interface goes up, the NDP and RIPng processing is started.

To send Router Advertisements or RIPng packets or both, local *and* remote Link-Local addresses must be configured.

Flags

Item	Description
-e [**off** | **compatible** | **only**]	Specifies the SEND mode: **off** Implies that the SEND option is not enabled. For example, the router behaves as is prior to RFC 3971/3972. **compatible** Implies that the router complies to RFC 3971/3972 but does not require the options specified in the RFC. The environment can be one where certain nodes are SEND capable while others are not. However, if the SEND options are embedded in the incoming packets, they must be correct. **only** Implies that all message must conform to RFC 3971/3972, or the message will be rejected. In order to enable the SEND option, you must install the clic.rte fileset and OpenSSL.

Item	Description
-H	Enables the system to process NDP features needed to function as a mobile IPv6 home agent
-m	Enables the system to aid movement detection for mobile IPv6 mobile nodes.
-D *max* [*min*[**/***life*]]	Sends Unsolicited Router Advertisements at intervals from *min* to *max* seconds. Default *max* value is 600 seconds, valid range is 4 to 1800 seconds. Default *min* equals to *max* / 3, valid range is from 1 to 0.75 * *max*. The router lifetime is set with *life*, default value is 10 * max. Valid range is 0 to 65535 seconds.
-T [*reachtim*] **/** [*retrans*] **/** [*hlim*]	Sets the BaseReachableTime field to *reachim* seconds, if *reachim* is not zero. If *retrans* is not zero, sets the RetransTime field to *retrans* seconds. If *hlim* is not zero, sets the hop limit field in Router Advertisements to *hlim*.
-M	Sets the **M** flag (stateful configuration) in advertisements.
-O	Sets the **O** flag (other stateful information) in advertisements
-p	Does not offer prefixes (learned from interface configuration).
-P [invlife]/ [deplife]	Sets the invalid life value and the deprecated life value for announced prefixes (in seconds). The default value is 0xffffffff (infinite).
-r	Does not offer to be the default router in Router Advertisements.
-s	Enables the RIPng protocol (the default is: RIPng disabled).
-q	Enables the RIPng protocol, but does not send RIPng packets.
-g	Broadcast a default route in RIPng.
-n	Does not install routes received by RIPng.
-u *port*	Uses UDP port *port* for RIPng. The default is 521.
-R	Uses split horizon without corrupting reverse for RIPng.
-S	Does not use any split horizon for RIPng.
-d	Enables debugging (exceptional conditions and dump).
-v	Logs all interesting events (daemon.info and console).
-t	Adds time stamps in logged messages.

Available directives

The main directives for the **/etc/gateway6** file are:

option [*option-directive* ...]
Sets per-interface/default options.

prefix [*prefix-directive* ...]
Sets per-interface/default prefix processing options.

filter [*filter-directive* ...]
Sets per-interface/default filters.

gateway directives
Sets routes in RIPng packets or in the kernel.

Each of these directives is explained in more detail below.

The option directive

Sets different per-interface options.

Any value settings for the **option** directive which follow the **if** option must appear in a comma-separated list.

Note: At least one option (other than the **if** option) must be specified following the **option** directive. If the **if** option is specified, it must be the first option following the **option** directive. There must be a space between the **if** option and any comma-separated list of options which follow.

Syntax:

option [**if**=*n1*,*n2*] **ripin**=(**y**|**n**),**ripout**=(**y**|**n**|**S**|**R**),**rtadv**=(**y**|**n**|*min*[/*max*]),**flag**=[**M**|**O**],**life**=*Seconds*,**reach**=*Seconds*,**retrans**=*Seconds*

Item	Description
if=*list* **interface=***list*	If there is no keyword, the option directive is a default option. If there is an interface field, the option parameters apply only to the listed interfaces. The list is comma-separated. You can use `le*` to match all the leX interfaces. The default option must be the first line in the **/etc/gateway6** file.
mtu[**=***mtuval*]	Advertises a MTU value of *mtuval* in router advertisements. If there is no *mtuval* argument, the advertised MTU is the MTU of the interface. If *mtuval* is 0, suppress the advertisement of MTU.
ripin=(**n**|**y**)	Does not listen (listen) to incoming RIPng packets. Does not send (send) RIPng packets. With the **-S** flag, do not use split horizon. With the **-R** flag, use split horizon without poisoning reverse.
rtadv=(**n**|**y**|*min* [**/***max*])	Does not send (send) router advertisements. With *min*[**/***max*] option, set the interval (in seconds) between router advertisements.
flag={**M**|**O**}	Sets the stateful mode flags in router advertisements. **M** Uses stateful configuration **O** Uses stateful configuration, but not for addresses
life=*Seconds*	Sets the router life field in router advertisements (in seconds).
reach=*Seconds*	Sets the reachable field in router advertisements (in seconds).
retrans=*Seconds*	Sets the retransmit interval field in router advertisements (in seconds).

The prefix directive

Defines the prefixes announced in Router advertisement directives. If there is no prefix-directive for an interface, the router advertisement contains the list of prefixes deduced from the address list of the interface. If there are prefix-directives, the router advertisement contains the list of prefixes defined by the different prefix directives (in order). No prefix is installed in the kernel. If there is one directive of the form prefix *prefix=none*, no prefix list is advertised.

Syntax:

prefix if=n **prefix**=(**none**|*xxx*::/*PrefixLength*) **flag**=[**L**][**A**] **valid**=*Seconds* **deprec**=*Seconds*

Item	Description
if=*Interface* or **interface**=*Interface*	Specifies the interface on which the directive applies. The **if** keyword is mandatory for the **prefix** directive. It is not an option.
prefix=*xxx*::/*PrefixLength*	The advertised prefix.
flag=[**L**][**A**]	Set the **L** and/or **A** flag for the prefix (the default is **LA**).
deprec=*Seconds*	Set the deprecated time (in seconds) for the prefix.
valid=*Seconds*	Set the validity time (in seconds) for the prefix.

The filter directive

Define a filter pattern for incoming (**filter**=**in**) or outgoing (**filter**=**out**) RIPng packets. There is one incoming and one outgoing filter per interface, and one default incoming and one default outgoing filter for interfaces without explicit filter.

Any received RIPng information is tested against the input filter of the interface, or, if there is none, against the default input filter. The static interface routes are seen as input information coming from the interface and from a gateway with the link local address of the interface. The routes set by a gateway directive with a **gateway** keyword are seen as input information coming from the specified interface and gateway. The default route (**-g** flag) and the routes set by a gateway directive without a **gateway** keyword are seen as input information coming from gateway :: and no interface (the default input filter applies).

Any sent RIPng information is tested against the output filter of the interface, or, if there is none, against the default output filter.

Each filter is a sequence of matching patterns. The patterns are tested in order. Each pattern can test the prefix length, the source gateway (for input filters and that the prefix (padded with zeroes) matches a fixed prefix. If a pattern contains more than one test description, the match is the conjunction of all the tests. The first matching pattern defines the action to perform. If no pattern matches, the default action is accept. The possible actions are accept, reject and truncate/*NumberOfBits*. The truncate/*NumberOfBits* action means: if the pattern matches and if prefix length is greater or equal to *NumberOfBits*, accept the prefix with new length *NumberOfBits*. The accepted prefix is immediately accepted, that is, not checked again against the filters.

For example, the following directive inhibits sending host routes on any interface without an explicit outgoing filter:

```
filter=out length==128 action=reject
```

Syntax:

filter=(**in**|**out**) [**if**=*n1*,*n2*] **prefix**=*xx*::/*NumberOfBits* **gateway**=*xxx* **length**=(=|>=|<=|<|>)*NumberOfBits* **action**=(**accept**|**reject**|**truncate**/*xx*)

Item	Description
if=*list* or **interface**=*list*	If there is no interface keyword, the filter directive is a default option. If there is an interface field, the filter pattern is added at the end of the filters of all specified interfaces. The list is comma-separated. For example, you can specify **interface**=le* to specify all the leX interfaces.
prefix=*xxx*::/*NumberOfBits*	The pattern matches only if *xxx*::/*NumberOfBits* is a prefix of the prefix in the RIPng packet.

Item	Description
gateway=*xxx*	The pattern matches only if the RIPng message comes from source address *xxx*, only in incoming filters.
length=(=|>=|<=|<|>)*NumberOfBits*	The pattern match only if the prefix length in the RIPng message is equal to (or greater than, less than, etc., depending on the operator specified) to *NumberOfBits*.
action=(**accept**|**reject**|**truncate**/*NumberOfBits*)	Specify the action to perform if the pattern matches: accept the message, reject the message, accept but truncate the prefix to *NumberOfBits* bits.

Gateway directives

The gateway directives allow the user to set up routes in RIPng packets and/or in the kernel. These directives must appear at the end of the **/etc/gateway6** file, after the other directives.

Syntax:

xxx::/*NumberOfBits* **metric** *Value*

xxx::/*NumberOfBits* **metric** *Value* **gateway** *IPv6Address ifname*

The second syntax is used to add the route to the kernel.

Examples

The following examples are of the **/etc/gateway6** file.

On a site where all addresses are of the form `5f06:2200:c001:0200:xxxx`, the following example means that only one route, describing all the site, is exported on all the Configured Tunnel Interface (CTI) **ctiX** interfaces. The keyword abbreviations shown are valid.

```
filt=out if=cti* pref=5f06:2200:c001:0200::/64 len=>=64 act=trunc/64
```

Setting a default outgoing route:

```
::/0 metric 2 gateway 5f06:2200:c102:0200::1 cti0
```

Declare that any CTI interface active with RIPng defines a default route:

```
filter=in if=cti* act=trunc/0
```

The following example defines a site with an exterior connection cti0, which aggregates other sites connected through ctiX, and which uses split horizon without poisoned reverse. The order of the lines is important, as all filter descriptions apply to cti0.

```
option if=cti* ripout=R
filter=out if=cti0 prefix=5f06:2200::/24 len=>=24 act=trunc/24
filt=out if=cti* pref=5f06:2200:c001:0200::/64 len=>=64 act=trunc/64
filter=in if=cti0 act=trunc/0
filter=in if=cti* prefix=5f06:2200::/24 len=>=24 act=trunc/64
filter=in if=cti* act=reject
```

Diagnostics

All errors are logged at the **daemon.err** level, unless the debug option is set. This includes all the syntax errors in the **/etc/gateway6** file and configuration mismatches between different routers.

Signals

ndpd-router responds to the following signals:

Item	Description
SIGINT	Dumps its current state to syslog, if syslog is defined. Otherwise, dumped to stdout.
SIGHUP	The **/etc/gateway6** file is read again.
SIGUSR1	Verbosity is incremented.
SIGUSR2	Verbosity is reset.
SIGTERM	Resets to a resonable state and stops.
SIGQUIT	Resets to a resonable state and stops.

Files

Item	Description
/etc/gateway6	
/etc/ndpd/sendr_anchor	The SEND router anchor file for the certificate chain.

ndx Command

Purpose

Creates a subject-page index for a document.

Syntax

ndx [*SubjectFile*] **"** *FormatterCommandLine* **"**

Description

The **ndx** command, given a list of subjects (*SubjectFile*), searches a specified English-language document and writes a subject-page index to standard output.

The document must include formatting directives for the **mm**, **mmt**, **nroff**, or **troff** commands. The formatter command line informs the **ndx** command whether the **troff** command, **nroff** command, **mm** command, or **mmt** command can be used to produce the final version of the document. These commands do the following:

Item	Description
troff or **mmt**	Specifies the **troff** command as the formatting program.
nroff or **mm**	Specifies the **nroff** command as the formatting program.

Parameters

Item	Description
SubjectFile	Specifies the list of subjects that are included in the index. Each subject must begin on a new line and have the following format: *word1*[*word2*...][,*wordk*...] For example: `printed circuit boards` `arrays` `arrays, dynamic storage` `Smith, W.P.` `printed circuit boards, channel-oriented multi-layer` `Aranoff` `University of Illinois` `PL/1` The subject must start in column one.
FormatterCommandLine	Creates the final form of the document. The syntax for this parameter is as follows: *Formatter* [*Flag...*] *File...* **mm -Tlp** File(s) **nroff -mm -Tlp -rW60** File(s) **troff -rB2 -Tibm3816 -r01.5i** File(s) For more information on the formatter command line, see the **mmt** command, **nroff** command, and html

neqn Command

Purpose

Formats mathematical text for the **nroff** command.

Syntax

neqn [**-d** *Delimiter1Delimiter2*] [**-f** *Font*] [**-p** *Number*] [**-s** *Size*] [**—**] [*File* ... | **-**]

Description

The **neqn** command is an **nroff** preprocessor for formatting mathematical text on typewriter-like terminals. Pipe the output of the **neqn** command into the **nroff** command as follows:

neqn [*Flag...*] *File...* **| nroff** [*Flag...*] **|** [*Printer*]

The **neqn** command reads one or more files. If no files are specified for the *File* parameter or the **-** (minus sign) flag is specified as the last parameter, standard input is read by default. A line beginning with the **.EN** macro. These lines are not altered by the **nroff** command, so they can be defined in macro packages to provide additional formatting functions such as centering and numbering.

The **—** (double dash) delimiter indicates the end of flags.

Depending on the target output devices, **neqn** command output formatted by the **nroff** command may need to be post-processed by the **eqn** command gives more information about the input format and keywords used.

Flags

Item	Description
-d*Delimiter1Delimiter2*	Sets two ASCII characters, *Delimiter1* and *Delimiter2,* as delimiters of the text to be processed by the **neqn** command, in addition to input enclosed by the **.EQ** and **.EN** macros. The text between these delimiters is treated as input to the **neqn** command. Within a file, you can also set delimiters for **neqn** text using the **delim** *Delimiter1Delimiter2* request. These delimiters are turned off by the **delim off** request. All text that is not between delimiters or the **.EN** macro is passed through unprocessed.
-f*Font*	Changes font in all the **neqn** command-processed text to the value specified by the *Font* variable. The *Font* value (a font name or position) must be one or two ASCII characters.
-p*Number*	Reduces subscripts and superscripts to the specified number of points in size. The default is 3 points.
-s*Size*	Changes point size in all the **neqn** command-processed text to the value specified by the *Size* variable.
-	Reads from standard input.
—	(double dash) Marks the end of the flags.

Files

Item	Description
/usr/share/lib/pub/eqnchar	Contains special character definitions.

netcd Daemon

Purpose

Launches the network caching (netcd) daemon.

Syntax

netcd [**-l** *file*] [**-c** *file*] [**-d** *level*] [**-h**]

Description

The **netcd** daemon reduces the time taken by the local, DNS, NIS, and user loadable module services to respond to a query by caching the response retrieved from resolvers.

When the **netcd** daemon is running and configured for a resolver (for example, DNS) and a map (for example, hosts), the resolution is first made using the cached answers. If it fails, the resolver is called and the response is cached by the **netcd** daemon.

The type of the maps that are supported for the local, NIS, and user loadable modules resolutions are hosts, services, networks, protocols and netgroup. For DNS, hosts is the only type of map that you can use.

In addition, for the specific case of Yellow Pages, the following maps have been added:

- passwd.byname
- passwd.byuid

- group.byname
- group.bygid
- netid.byname
- passwd.adjunct.byname

You can use a configuration file to specify the resolvers and maps that you want to configure. You can also set other **netcd** parameters using this file. By default, the configuration file used is the **/etc/netcd.conf** file. You can change the path of this configuration file using the **-c** argument of the **netcd** daemon. If the **/etc/netcd.conf** file does not exist, the **netcd** daemon uses the default parameters. You can find a sample of this file under the **/usr/samples/tcpip** file. Do not use this file as a configuration file because it will be overwritten by a new installation of the package containing the file.

You can specify the level of debugging using the **-d** argument. The debugging levels are similar to the one used by the **syslogd** daemon. Log messages are written to the **/var/tmp/netcd.log** file. You can override the default using the netcd configuration file. As with the **syslogd** daemon, you can specify rotation for the netcd log file.

netcd Parameters

When an entry is inserted in a netcd cache, a time-to-live (TTL) is associated to it. You can configure this TTL using the netcd configuration file (cache declarations). For DNS, this TTL is the one contains the response from the DNS.

To clean the caches of outdated entries, you must run two tasks periodically, one to clean local caches and the other to clean the other caches. You can set the frequency of these tasks using the *local_scan_frequency* and *net_scan_frequency* parameters in the netcd configuration file.

Caches are hashed tables. The size of the hash tables can be controlled using the netcd configuration file and the **netcdctrl** command.

To communicate between the applications, the **netcd** daemon uses a socket (**/dev/netcd**). You can configure the size of the message queue using the netcd configuration file.

netcd supports the System Resource Controller

The **netcd** daemon is part of the netcd System Resource Controller (SRC) group. The following are the SRC commands you can use to manage the **netcd** daemon:

- You can start the **netcd** daemon using the **startsrc** command, or stop the **netcd** daemon using the **stopsrc** command.
- The **lssrc** command provides a short status output that includes the Process ID (PID) and the status of the **netcd** daemon.
- The **lssrc -l** command provides a long status output that includes the PID, the status of the **netcd** daemon, the configuration file used when starting the **netcd** daemon, and the configured caches.

Note: You cannot use the **refresh** command with the **netcd** daemon.

Flags

Item	Description
-c *file*	Specifies a configuration file. The default file name is **/etc/netcd.conf**.
-d *level*	Specifies the logging level. The *level* value must be an integer between 0 and 7.
-h	Displays help information.
-l *file*	Loads caches from the specified binary file created by the **netcdctrl** command. The local files (for example, **/etc/hosts**, **/etc/services**) are loaded depending on the configuration file.

Examples

1. To launch the **netcd** daemon using the SRC, enter:

```
startsrc -s netcd
```

2. To display the status of the **netcd** daemon using the SRC, enter:

```
lssrc -s netcd
```

   This command produces the following output:

```
Subsystem         Group            PID          Status
netcd             netcd            299064       active
```

3. To display the status of the **netcd** daemon in long form using the SRC, enter:

```
lssrc -l -s netcd
```

   This command produces the following output:

```
Subsystem         Group            PID          Status
netcd             netcd            299064       active
Configuration File       /etc/netcd.conf
Configured Cache         local services
Configured Cache         local protocols
Configured Cache         local hosts
Configured Cache         local networks
Configured Cache         local netgroup
```

4. To launch the **netcd** daemon without using the SRC, enter:

```
netcd
```

netcdctrl Command

Purpose

Manages the network caching (netcd) daemon caches.

Syntax

netcdctrl [**-t** *type* **-e** *type* [**-a** *file* | **-b** *file* | **-f** | **-s** *file*]] [**-l** *level*] [**-h**]

Description

The **netcdctrl** command provides the following functions:

- Dumps specific caches in ASCII format: provides a readable output of the caches content.
- Dumps specific caches in binary format. The binary format can be used later to reload the caches when starting the **netcd** daemon. Dumping avoids reloading the caches from the beginning.
- Displays statistics on caches use. The caches are tables, and the access to these tables is controlled by a hash algorithm. This output helps you size the table for a given resolution and a given map using the netcd configuration file.
- Flushes specific caches. The content of the specified caches are erased, and local caches are then reloaded. Other caches are reloaded by resolver's responses.
- Changes the logging level dynamically.

Requirement: You must have the root authority to issue the **netcdctrl** command.

Flags

Item	Description
-a *file*	Specifies ASCII dumping of the specified caches.
-b *file*	Specifies binary dumping of the specified caches (local caches are not dumped).
-e *type*	Specifies the map. The *type* parameter can be one of the following values: • hosts • protocols • servers • networks • netgroup • a yellow pages map name (for example passwd.byname or group.bygid) • all Use this flag only with the **-b**, **-a**, **-f** and **-s** flags.
-f	Flushes the specified caches.
-h	Displays help information.
-l *level*	Changes the logging level of the **netcd** daemon. The *level* value must be an integer of 0 through 7.
-s *file*	Provides statistics on caches use.
-t *type*	Specifies the resolution. The *type* parameter can be one of the following values: • local • dns • nis • yp • ulm • a specific module name as provided in the **netcd.conf** file • all Use his flag only with the **-b**, **-a**, **-f** and **-s** flags.

Examples

1. To flush all the caches, enter:

```
netcdctrl -t all -e all -f
```

2. To dump all the NIS caches in binary format, enter:

```
netcdctrl -t nis -e all -b /tmp/netcd_nis_binary_dump
```

3. To dump the local cache for hosts in ASCII format, enter:

```
netcdctrl -t local -e hosts -a /tmp/netcd_dns_hosts
```

4. To set the level of logging to obtain all possible traces, enter:

```
netcdctrl -l 7
```

netpmon Command

Purpose

Monitors activity and reports statistics on network I/O and network-related CPU usage.

Syntax

netpmon [**-o** *File*] [**-d**] [**-T** *n*] [**-P**] [**-t**] [**-v**] [**-r PURR**] [**-O** *ReportType ...*] [**-i** *Trace_File* **-n** *Gensyms_File*] [**-@** [*WparList* | **ALL**]]

Description

The **netpmon** command monitors a trace of system events, and reports on network activity and performance during the monitored interval. By default, the **netpmon** command runs in the background while one or more application programs or system commands are being executed and monitored. The **netpmon** command automatically starts and monitors a trace of network-related system events in real time. By default, the trace is started immediately; optionally, tracing may be deferred until the user issues a **trcon** command. When tracing is stopped by a **trcstop** command, the **netpmon** command generates all specified reports and exits.

The **netpmon** command can also work in offline mode, that is, on a previously generated trace file. In this mode, a file generated by the **gensyms** command is also required. The gensyms file should be generated immediately after the trace has been stopped, and on the same machine. When running in offline mode, the **netpmon** command cannot recognize protocols used by sockets, which limits the level of detail available in the socket reports.

The **netpmon** command reports on the following system activities:

CPU Usage
The **netpmon** command monitors CPU usage by all threads and interrupt handlers. It estimates how much of this usage is due to network-related activities.

Network Device-Driver I/O
The **netpmon** command monitors I/O operations through token-ring and Fiber-Distributed Data Interface (FDDI) network device drivers. In the case of transmission I/O, the command also monitors utilizations, queue lengths, and destination hosts. For receive ID, the command also monitors time in the demux layer.

Internet Socket Calls
The **netpmon** command monitors all **send**, **recv**, **sendto**, **recvfrom**, **read**, and **write** subroutines on Internet sockets. It reports statistics on a per-process basis, for each of the following protocol types:

- Internet Control Message Protocol (ICMP)
- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)

NFS I/O
The **netpmon** command monitors **read** and **write** subroutines on client Network File System (NFS) files, client NFS remote procedure call (RPC) requests, and NFS server read or write requests. The command reports subroutine statistics on a per-process or optional per-thread basis and on a per-file basis for each server. The **netpmon** command reports client RPC statistics for each server, and server read and write statistics for each client.

Any combination of the preceding report types can be specified with the command line flags. By default, all the reports are produced.

> **Notes:** The reports produced by the **netpmon** command can be quite long. Consequently, the **-o** flag should usually be used to write the report to an output file. The **netpmon** command obtains performance data using the system trace facility. The trace facility only supports one output stream. Consequently, only one **netpmon** or **trace** process can be active at a time. If

another **netpmon** or **trace** process is already running, the **netpmon** command responds with the message:

```
/dev/systrace: Device busy
```

While monitoring very network-intensive applications, the **netpmon** command may not be able to consume trace events as fast as they are produced in real time. When that happens, the error message:

```
Trace kernel buffers overflowed, N missed entries
```

displays on standard error, indicating how many trace events were lost while the trace buffers were full. The **netpmon** command continues monitoring network activity, but the accuracy of the report diminishes by some unknown degree. One way to avoid overflow is to increase the trace buffer size using the **-T** flag, to accommodate larger bursts of trace events before overflow. Another way to avoid overflow problems all together is to run netpmon in offline mode.

When running in memory-constrained environments (where demand for memory exceeds supply), the **-P** flag can be used to pin the text and data pages of the real-time **netpmon** process in memory so the pages cannot be swapped out. If the **-P** flag is not used, allowing the **netpmon** process to be swapped out, the progress of the **netpmon** command may be delayed such that it cannot process trace events fast enough to prevent trace buffer overflow.

If the **/unix** file and the running kernel are not the same, the kernel addresses will be incorrect, causing the **netpmon** command to exit.

Flags

Item	Description
-d	Starts the **netpmon** command, but defers tracing until the **trcon** command has been executed by the user. By default, tracing is started immediately.
-i *Trace_File*	Reads trace records from the file *Trace_File* produced with the **trace** command instead of a live system. The trace file must be rewritten first in raw format using the **trcpt -r** command. This flag cannot be used without the **-n** flag.
-n *Gensyms_File*	Reads necessary mapping information from the file *Gensyms_File* produced by the **gensyms** command. This flag is mandatory when the **-i** flag is used.
-o *File*	Writes the reports to the specified *File*, instead of to standard output.

Item	Description
-O *ReportType ...*	Produces the specified report types. Valid report type values are: **cpu** CPU usage **dd** Network device-driver I/O. This report is not available inside a workload partition (WPAR) in online mode or in the global WPAR with the '*-@ WparList*' flag. **so** Internet socket call I/O **nfs** NFS I/O (any version) **nfs2** NFS Version 2 I/O **nfs3** NFS Version 3 I/O **nfs4** NFS Version 4 I/O **all** All reports are produced. This is the default value when the **netpmon** command is run in the global WPAR without the **-@** flag.
-P	Pins monitor process in memory. This flag causes the **netpmon** text and data pages to be pinned in memory for the duration of the monitoring period. This flag can be used to ensure that the real-time **netpmon** process does not run out of memory space when running in a memory-constrained environment.
-r PURR	Uses PURR time instead of TimeBase in percent and CPU time calculation. Elapsed time calculations are unaffected.
-t	Prints CPU reports on a per-thread basis.
-T *n*	Sets the kernel's trace buffer size to *n* bytes. The default size is 64000 bytes. The buffer size can be increased to accommodate larger bursts of events, if any. (A typical event record size is on the order of 30 bytes.) **Note:** The trace driver in the kernel uses double buffering, so actually two buffers of size *n* bytes will be allocated. These buffers are pinned in memory, so they are not subject to paging.
-v	Prints extra information in the report. All processes and all accessed remote files are included in the report instead of only the 20 most active processes and files.
-@ [*WparList*|**ALL**]	Specifies that reports are limited to the list of WPARs that are passed as an argument.

Reports

The reports generated by the **netpmon** command begin with a header, which identifies the date, the machine ID, and the length of the monitoring period in seconds. This is followed by a set of summary and detailed reports for all specified report types.

CPU Usage Reports

Process CPU Usage Statistics: Each row describes the CPU usage associated with a process. Unless the verbose option is specified, only the 20 most active processes are listed. At the bottom of the report, CPU usage for all processes is totaled, and CPU idle time is reported.

Process
Process name

PID
Process ID number

CPU Time
Total amount of CPU time used by this process

CPU %
CPU usage for this process as a percentage of total time

Network CPU %
Percentage of total time that this process spent executing network-related code

Thread CPU Usage Statistics
If the **-t** flag is used, each process row described above is immediately followed by rows describing the CPU usage of each thread owned by that process. The fields in these rows are identical to those for the process, except for the name field. (Threads are not named.)

First-Level Interrupt Handler Usage Statistics: Each row describes the CPU usage associated with a first-level interrupt handler (FLIH). At the bottom of the report, CPU usage for all FLIHs is totaled.

FLIH
First-level interrupt handler description

CPU Time
Total amount of CPU time used by this FLIH

CPU %
CPU usage for this interrupt handler as a percentage of total time

Network CPU %
Percentage of total time that this interrupt handler executed on behalf of network-related events

Second-Level Interrupt Handler Usage Statistics: Each row describes the CPU usage associated with a second-level interrupt handler (SLIH). At the bottom of the report, CPU usage for all SLIHs is totaled.

SLIH
Second-level interrupt handler description

CPU Time
Total amount of CPU time used by this SLIH

CPU %
CPU usage for this interrupt handler as a percentage of total time

Network CPU %
Percentage of total time that this interrupt handler executed on behalf of network-related events

Summary Network Device-Driver Reports

Network Device-Driver Statistics (by Device): Each row describes the statistics associated with a network device.

Device
Path name of special file associated with device

Xmit Pkts/s
Packets per second transmitted through this device

Xmit Bytes/s
Bytes per second transmitted through this device

Xmit Util
Busy time for this device, as a percent of total time

Xmit Qlen
Number of requests waiting to be transmitted through this device, averaged over time, including any transaction currently being transmitted

Recv Pkts/s
Packets per second received through this device

Recv Bytes/s
Bytes per second received through this device

Recv Demux
Time spent in demux layer as a fraction of total time

Network Device-Driver Transmit Statistics (by Destination Host): Each row describes the amount of transmit traffic associated with a particular destination host, at the device-driver level.

When hosts are on the same subnet, the destination host name is displayed. When hosts are in a different subnet, the destination host can be bridges, routers, or gateways as resolved by ARP protocol.

Host
Destination host name. An * (asterisk) is used for transmissions for which no host name can be determined.

Pkts/s
Packets per second transmitted to this host

Xmit Bytes/s
Bytes per second transmitted to this host

Summary Internet Socket Reports

- *On-line mode*: **Socket Call Statistics for Each Internet Protocol (by Process)**: Each row describes the amount of **read/write** subroutine activity on sockets of this protocol type associated with a particular process. Unless the verbose option is specified, only the top 20 processes are listed. At the bottom of the report, all socket calls for this protocol are totaled.
- *Off-line mode*: **Socket Call Statistics for Each Process**: Each row describes the amount of **read/write** subroutine activity on sockets associated with a particular process. Unless the verbose option is specified, only the top 20 processes are listed. At the bottom of the report, all socket calls are totaled.

Process
Process name

PID
Process ID number

Read Calls/s or Read Ops/s
Number of **read** , **recv** , and **recvfrom** subroutines per second made by this process on sockets of this type

Read Bytes/s
Bytes per second requested by the above calls

Write Calls/s or Write Ops/s
Number of **write** , **send** , and **sendto** subroutines per second made by this process on sockets of this type

Write Bytes/s
Bytes per second written by this process to sockets of this protocol type

Summary NFS Reports

NFS Client Statistics for Each Server (by File): Each row describes the amount of **read**/**write** subroutine activity associated with a file mounted remotely from this server. Unless the verbose option is specified, only the top 20 files are listed. At the bottom of the report, calls for all files on this server are totaled.

File
Simple file name

Read Calls/s or Read Ops/s
Number of **read** subroutines per second on this file

Read Bytes/s
Bytes per second requested by the above calls

Write Calls/s or Write Ops/s
Number of **write** subroutines per second on this file

Write Bytes/s
Bytes per second written to this file

NFS Client RPC Statistics (by Server): Each row describes the number of NFS remote procedure calls being made by this client to a particular NFS server. At the bottom of the report, calls for all servers are totaled.

Server
Host name of server. An * (asterisk) is used for RPC calls for which no hostname could be determined.

Calls/s or Ops/s
Number of NFS RPC calls per second being made to this server.

NFS Client Statistics (by Process): Each row describes the amount of NFS **read**/**write** subroutine activity associated with a particular process. Unless the verbose option is specified, only the top 20 processes are listed. At the bottom of the report, calls for all processes are totaled.

Process
Process name

PID
Process ID number

Read Calls/s or Read Ops/s
Number of NFS **read** subroutines per second made by this process

Read Bytes/s
Bytes per second requested by the above calls

Write Calls/s or Write Ops/s
Number of NFS **write** subroutines per second made by this process

Write Bytes/s
Bytes per second written to NFS mounted files by this process

NFS Server Statistics (by Client): Each row describes the amount of NFS activity handled by this server on behalf of particular client. At the bottom of the report, calls for all clients are totaled.

Client
Host name of client

Read Calls/s or Read Ops/s
Number of remote read requests per second processed on behalf of this client

Read Bytes/s
Bytes per second requested by this client's read calls

Write Calls/s or Write Ops/s
Number of remote write requests per second processed on behalf of this client

Write Bytes/s
Bytes per second written by this client

Other Calls/s or Ops/s
Number of other remote requests per second processed on behalf of this client

Detailed Reports

Detailed reports are generated for any of the specified report types. For these report types, a detailed report is produced for most of the summary reports. The detailed reports contain an entry for each entry in the summary reports with statistics for each type of transaction associated with the entry.

Transaction statistics consist of a count of the number of transactions of that type, followed by response time and size distribution data (where applicable). The distribution data consists of average, minimum, and maximum values, as well as standard deviations. Roughly two-thirds of the values are between `average - standard deviation` and `average + standard deviation`. Sizes are reported in bytes. Response times are reported in milliseconds.

Detailed Second Level Interrupt Handler CPU Usage Statistics:

SLIH
Name of second-level interrupt handler

Count
Number of interrupts of this type

CPU Time (Msec)
CPU usage statistics for handling interrupts of this type

Detailed Network Device-Driver Statistics (by Device):

Device
Path name of special file associated with device

Recv Packets
Number of packets received through this device

Recv Sizes (Bytes)
Size statistics for received packets

Recv Times (msec)
Response time statistics for processing received packets

Xmit Packets
Number of packets transmitted to this host

Demux Times (msec)
Time statistics for processing received packets in the demux layer

Xmit Sizes (Bytes)
Size statistics for transmitted packets

Xmit Times (Msec)
Response time statistics for processing transmitted packets

Detailed Network Device-Driver Transmit Statistics (by Host):

Host
Destination host name

Xmit Packets
Number of packets transmitted through this device

Xmit Sizes (Bytes)
Size statistics for transmitted packets

Xmit Times (Msec)
Response time statistics for processing transmitted packets

Detailed Socket Call Statistics for Each Internet Protocol (by Process): (*on-line* mode) **Detailed Socket Call Statistics for Each Process**: (*off-line* mode)

Process
Process name

PID
Process ID number

Reads
Number of **read** , **recv** , **recvfrom** , and **recvmsg** subroutines made by this process on sockets of this type

Read Sizes (Bytes)
Size statistics for **read** calls

Read Times (Msec)
Response time statistics for **read** calls

Writes
Number of **write** , **send** , **sendto** , and **sendmsg** subroutines made by this process on sockets of this type

Write Sizes (Bytes)
Size statistics for **write** calls

Write Times (Msec)
Response time statistics for **write** calls

Detailed NFS Client Statistics for Each Server (by File):

File
File path name

Reads
Number of NFS **read** subroutines for this file

Read Sizes (Bytes)
Size statistics for **read** calls

Read Times (Msec)
Response time statistics for **read** calls

Writes
Number of NFS **write** subroutines for this file

Write Sizes (Bytes)
Size statistics for **write** calls

Write Times (Msec)
Response time statistics for **write** calls

Detailed NFS Client RPC Statistics (by Server):

Server
Server host name

Calls
Number of NFS client RPC calls made to this server

Call Times (Msec)
Response time statistics for RPC calls

Detailed NFS Client Statistics (by Process):

Process
Process name

PID
Process ID number

Reads
Number of NFS **read** subroutines made by this process

Read Sizes (Bytes)
Size statistics for **read** calls

Read Times (Msec)
Response time statistics for **read** calls

Writes
Number of NFS **write** subroutines made by this process

Write Sizes (Bytes)
Size statistics for **write** calls

Write Times (Msec)
Response time statistics for **write** calls

Detailed NFS Server Statistics (by Client):

Client
Client host name

Reads
Number of NFS read requests received from this client

Read Sizes (Bytes)
Size statistics for read requests

Read Times (Msec)
Response time statistics for read requests

Writes
Number of NFS write requests received from this client

Write Sizes (Bytes)
Size statistics for write requests

Write Times (Msec)
Response time statistics for write requests

Other Calls
Number of other NFS requests received from this client

Other Times (Msec)
Response time statistics for other requests

Examples

1. To monitor network activity during the execution of certain application programs and generate all report types, type:

   ```
   netpmon
   <run application programs and commands here>
   trcstop
   ```

   The **netpmon** command automatically starts the system trace and puts itself in the background. Application programs and system commands can be run at this time. After the **trcstop** command is issued, all reports are displayed on standard output.

2. To generate CPU and NFS report types and write the reports to the `nmon.out` file, type:

   ```
   netpmon -o nmon.out -O cpu,nfs
   <run application programs and commands here>
   trcstop
   ```

   The **netpmon** command immediately starts the system trace. After the **trcstop** command is issued, the I/O activity report is written to the `nmon.out` file. Only the CPU and NFS reports will be generated.

3. To generate all report types and write verbose output to the `nmon.out` file, type:

   ```
   netpmon -v -o nmon.out
   <run application programs and commands here>
   trcstop
   ```

   With the verbose output, the **netpmon** command indicates the steps it is taking to start up the trace. The summary and detailed reports include all files and processes, instead of just the 20 most active files and processes.

4. To use the **netpmon** command in offline mode, type:

   ```
   trace -a
   run application programs and commands here
   trcoff
   gensyms > gen.out
   trcstop
   netpmon -i tracefile -n gen.out -o netpmon.out
   ```

netrule Command

Purpose

Adds, removes, lists, or queries rules, flags and security labels for interfaces and hosts.

Syntax

netrule hl [**i** | **o** | **io**]

netrule hq { **i** | **o** } *src_host_rule_specification dst_host_rule_specification*

netrule h- [**i** | **o**][**u**] [*src_host_rule_specification dst_host_rule_specification*]

netrule h+ { **i** | **o** } [**u**] *src_host_rule_specification dst_host_rule_specification* [*flags*][*RIPSO/CIPSO options*] *security_label_information*

netrule il

netrule iq *interface*

netrule i- [**u**][*interface*]

netrule i+ [**u**] *interface* [*flags*][*RIPSO/CIPSO options*] *security_label_information*

netrule eq

netrule e { **on** | **off** }

Description

The **netrule** command lists, queries, adds and removes rule specifications for interfaces and hosts. The system default interface rules are set using the interface name. When an interface is removed using the **i-** flag, it will be given these default interface rules. The default interface rules are also set using the **tninit load** command.

Note: Because there must always be an interface rule for an interface, the remove operation sets the interface rule to its default state. All of the command line flags must follow the order as shown in the syntax statements.

Flags

Item	Description
e { **on** | **off** }	Sets the policy for sending the ICMP error response to incoming packets that are not accepted by the system. This setting is off by default and must be set with this flag to be on. You cannot specify the **e** flag when you specify the **h** or **i** flag.
h	Specifies that the object of the **netrule** command is a host. You cannot specify the **h** flag when you specify the **i** or **e** flag.
i	Specifies that the object of the **netrule** command is an interface. You cannot specify the **i** flag when you specify the **h** or **e** flag.
l	Lists all rules for interfaces or hosts.
o	Specifies the host out rules (for host rule only).
q	Queries an interface, a host rule, or the status of the error response setting.
u	Specifies that the **/etc/security/rules.host** and **/etc/security/rules.int** files will be updated after the host or interface rule is successfully added or removed.
+	Adds an interface or a host rule.

Item	Description
-	Removes an interface or a host rule.
interface	Specifies an interface name.
src_host_rule_specification	This parameter takes the following format: `src_host [/ mask][ = proto [:start_port_range [:end_port_range]]]` **Requirement:** There is a space or tab in between each field. **src_host** A source IPv6 address, or an IPv4 address, or a host name. **mask** The subnet mask number indicates how many bits are set, starting from the most significant bit. For example, 24 means 255.255.255.0 for an IPv4 address. **proto** A protocol. **start_port_range** A particular port number or name to begin from. **end_port_range** A particular port number or name to end at.
dst_host_rule_specification	This parameter takes the following format: `dst_host [/ mask][ = proto [:start_port_range [:end_port_range]]]` **Requirement:** There is a space or tab in between each field. **dst_host** A destination IPv6 address, or an IP v4 address, or a host name. **mask** The subnet mask number, which indicates how many bits are set, starting from the most significant bit. For example, 24 means 255.255.255.0 for an IPv4 address. **proto** A protocol. **start_port_range** A particular port number or name to begin in range from. **end_port_range** A particular port number or name to end at.

Item	Description
flags	This parameter takes the following format: `-d drop` ***drop*** AIX Trusted Network can be configured to drop all packets. You can specify one of the following values: **r** Drops all packets **n** Does not drop all packets (interface default). **i** Uses interface default (host default, host only). `-f rflag:tflag` ***rflags*** Security option requirement on incoming (received) packets. You can specify one of the following values: **r** Revised Interconnection Protocol Security Option (RIPSO) only. **c** Commercial Internet Protocol Security Option (CIPSO) only. **e** Either RIPSO or CIPSO. **n** Neither RIPSO or CIPSO (system default). **a** No restrictions. **i** Uses interface or system default (default). ***tflag*** Security option handling on outgoing (transmitted) packets. You can specify one of the following values: **r** Transmits RIPSO. **c** Transmits CIPSO. **n** Does not transmit any security options (interface default). **i** Uses interface default (host default, host only).

Item	Description
RIPSO/CIPSO options	This parameter takes the following format: **-rpafs=*PAF_field*[,*PAF_field*...]** Specifies the PAF fields that are used to receive IPSO packets. This is a list of PAF fields that are accepted. There can be up to 256 fields. ***PAF_field*: NONE | *PAF* [+*PAF*...]** Specifies PAF fields, which are collections of PAFs. The following are the five PAFs that can be included in a single PAF field: • GENSER • SIOP-ESI • SCI • NSA • DOE A PAF field is a combination of these values separated by a plus sign (**+**). For example, a PAF field containing both GENSER and SCI is represented as GENSER+SCI. You can use the PAF field NONE to specify the PAF field without any specified PAFs. **-epaf=*PAF_field*** Specifies the PAF field that is attached to error responses for incoming IPSO packets that were not accepted by the system. **-tpaf=*PAF_field*** Specifies the PAF field that is included in the IPSO options of outgoing packets. **-DOI = *doi*** Specifies the domain of interpretation (DOI) for CIPSO packets. Incoming packets must have this DOI and outgoing packets will be given this DOI. **-tags=*tag*[,*tag*...]** `tag = 1 | 2 | 5` Specifies the set of tags that are accepted and available to be transmitted by CIPSO options. This is a combination of 1, 2 and 5. For example 1,2 would enable tags 1 and 2.
security_label_information	This parameter takes the following format: **+min +max +default | -s input_file** Specifies the standard output (SL) that will apply when adding a rule. You can also specify the **-s** flag and include the SLs in the file in the following order, specifying one per line: • min SL • max SL • default SL You cannot include any comments in the file. Use a backslash (\) at the end of the line if more than one line is needed. If you are not using a file, list the sensitivity labels delimited by a plus sign (+) for the minimum level, the maximum level, and the default or implicit level for unmarked packets.

Security

A user must have the **aix.mls.network.config** and the **aix.mls.network.init** authorizations to run the **netrule** command.

Examples

1. To add in host rule, and update the local database after in host rule is successfully added to kernel, enter:

   ```
   netrule h+iu 9.3.149.25 9.41.86.19 +impl_lo +ts all +pub
   ```

2. To add out host rule, enter:

   ```
   netrule h+o 9.41.86.19  9.3.149.25 -s /tmp/rule
   ```

   or:

   ```
   impl_lo
   ts all
   pub
   ```

   The following are the contents of the input **/tmp/rule** file:

   ```
   impl_lo
   ts \
   all
   pub
   ```

3. To drop all incoming UDP packets from a host, enter:

   ```
   netrule h+i 192.0.0.5 =udp 9.41.86.19 =udp -dr +impl_lo +impl_lo +impl_lo
   ```

4. To remove all host rules and update the local, enter:

   ```
   netrule h-u
   ```

5. To list all host rules, enter:

   ```
   netrule hl
   ```

6. To list all interface rules, enter:

   ```
   netrule il
   ```

7. To add an interface rule, enter:

   ```
   netrule i+ en0 -dn -fa:n +public +ts +secret
   ```

8. To remove a particular host rule, enter:

   ```
   netrule h-i 192.0.0.5 =udp 9.41.86.19 =udp
   ```

9. To add a particular host rule, enter:

   ```
   netrule h+i 9.41.86.19 /24 =tcp :ftp :telnet 9.3.149.6 /28 +public +ts +secret
   ```

10. To set the default interface rule, enter:

    ```
    netrule i+ default -dn -fa:n +impl_lo +ts all +impl_lo
    ```

11. To set the default interface rule to the system drop-all-packets default, enter:

    ```
    netrule i- default
    ```

12. To set the interface to send and only receive CIPSO packets, enter:

```
netrule i+ en0 -fc:c +impl_lo +ts all +impl_lo
```

13. To set the interface to receive either CIPSO or RIPSO packets and send RIPSO packets with PAF values, a CIPSO DOI, and CIPSO flags, enter:

```
netrule i+ en0 -fe:r -rpafs=SCI,NSA+DOE -epaf=SCI -tpaf=NSA -DOI=0x010
-tags=1,2 +impl_lo +ts all +impl_lo
```

14. To set the system-wide policy for sending ICMP responses on incoming packets that are not valid, enter:

```
netrule e on
```

netstat Command

Purpose

Shows network status.

Syntax

To Display Active Sockets for Each Protocol or Routing Table Information

/bin/netstat [**-n**] [{**-A** **-a** } | { **-r** **-C** **-i** **-I** *Interface* }] [**-f** *AddressFamily*] [[**-p** *Protocol*] | [**-@** *WparName*]] [*Interval*]

To Display the Contents of a Network Data Structure

/bin/netstat [**-m** | **-M** | **-s** | **-ss** | **-u** | **-v**] [**-f** *AddressFamily*] [[**-p** *Protocol*] | [**-@** *WparName*]] [*Interval*]

To Display the Virtual Interface Table and Multicast Forwarding Cache

/bin/netstat -g

To Display the Packet Counts Throughout the Communications Subsystem

/bin/netstat -D

To Display the Network Buffer Cache Statistics

/bin/netstat -c

To Display the Data Link Provider Interface Statistics

/bin/netstat -P

To Clear the Associated Statistics

/bin/netstat [**-Zc** | **-Zi** | **-Zm** | **-Zs**]

Description

The **netstat** command symbolically displays the contents of various network-related data structures for active connections. The *Interval* parameter, which is specified in seconds, continuously displays information regarding packet traffic on the configured network interfaces. The *Interval* parameter takes no flags.

Flags

Item	Description
-A	Shows the address of any protocol control blocks associated with the sockets. This flag acts with the default display and is used for debugging purposes.
-a	Shows the state of all sockets. If this flag is not specified, sockets that are used by server processes that are not bound to an interface are not shown.
-c	Shows the statistics of the Network Buffer Cache. The Network Buffer Cache is a list of network buffers that contain data objects that can be transmitted to networks. The Network Buffer Cache grows dynamically as data objects are added to or removed from it. The Network Buffer Cache is used by some network kernel interfaces for performance enhancement on the network I/O. The **netstat -c** command prints the following statistic: `Network Buffer Cache Statistics:` `Current total cache buffer size: 0` `Maximum total cache buffer size: 0` `Current total cache data size: 0` `Maximum total cache data size: 0` `Current number of cache: 0` `Maximum number of cache: 0` `Number of cache with data: 0` `Number of searches in cache: 0` `Number of cache hit: 0` `Number of cache miss: 0` `Number of cache newly added: 0` `Number of cache updated: 0` `Number of cache removed: 0` `Number of successful cache accesses: 0` `Number of unsuccessful cache accesses: 0` `Number of cache validation: 0` `Current total cache data size in private segments: 0` `Maximum total cache data size in private segments: 0` `Current total number of private segments: 0` `Maximum total number of private segments: 0` `Current number of free private segments: 0` `Current total NBC_NAMED_FILE entries: 0` `Maximum total NBC_NAMED_FILE entries: 0`

Item	Description
-C	Shows the routing tables, including the user-configured and current costs of each route. The user-configured cost is set by using the **-hopcount** flag of the **route** command. The current cost can be different than the user-configured cost if Dead Gateway Detection has changed the cost of the route. In addition to the costs of the route, it also shows the weight and policy information associated with each route. These fields are applicable only when the Multipath Routing Feature is used. The policy information displays the routing policy that has been currently selected to choose between the multiple routes available. The policies available are: • Default - Weighted Round Robin (WRR) • Hashed (HSH) • Random (RND) • Weighted Random (WRND) • Lowest Utilization (LUT) If multiple routes are present for the same destination (multipath routes), one of these routes display the policy value of `WRR`, `HSH`, `RND`, `WRND`, or `LUT`. All the other routes in this set display the policy information as `-"-`. This means that all the routes in this set are using the same routing policy displayed by the first route. The weight field is a user-configured weight associated with the route that will be used for Weighted Round-Robin and Weighted Random Policies. For more information about these policies, see the **no** command.
-D	Shows the number of packets received, transmitted, and dropped in the communications subsystem. **Note:** In the statistics output, a `N/A` displayed in a field value indicates the count is not applicable. For the NFS/RPC statistics, the number of incoming packets that pass through RPC are the same packets that pass through NFS, so these numbers are not summed in the `NFS/RPC Total` field, thus the `N/A`. NFS has no outgoing packet or outgoing packet drop counters specific to NFS and RPC. Therefore, individual counts have a field value of `N/A`, and the cumulative count is stored in the `NFS/RPC Total` field.
-f *AddressFamily*	Limits reports of statistics or address control blocks to those items specified by the *AddressFamily* variable. The following address families are recognized: **inet** Indicates the AF_INET address family. **inet6** Indicates the AF_INET6 address family. **unix** Indicates the AF_UNIX address family.
-g	Shows Virtual Interface Table and Multicast Forwarding Cache information. If used in conjunction with the **-s** flag, it will show the multicast routing information.

Item	Description
-i	Shows the state of all configured interfaces. See Interface Display **Note:** The collision count for Ethernet interfaces is not supported.
-I *Interface*	Shows the state of the configured interface specified by the *Interface* variable.
-M	Shows network memory's mbuf cluster pool statistics.
-m	Shows statistics recorded by the memory management routines.
-n	Shows network addresses as numbers. When this flag is not specified, the **netstat** command interprets addresses where possible and displays them symbolically. This flag can be used with any of the display formats.
-o	Used in conjunction with the **-a** flag to display detailed data about a socket, such as socket options, flags, and buffer statistics.
-p *Protocol*	Shows statistics about the value specified for the *Protocol* variable, which is either a well-known name for a protocol or an alias for it. Some protocol names and aliases are listed in the **/etc/networks** file.
-P	Shows the statistics of the Data Link Provider Interface (DLPI). The **netstat -P** command prints the following statistic: `DLPI statistics:` `Number of received packets = 0` `Number of transmitted packets = 0` `Number of received bytes = 0` `Number of transmitted bytes = 0` `Number of incoming pkts discard = 0` `Number of outgoing pkts discard = 0` `Number of times no buffers = 0` `Number of successful binds = 0` `Number of unknown message types = 0` `Status of phys level promisc = 0` `Status of sap level promisc = 0` `Status of multi level promisc = 0` `Number of enab_multi addresses = 0` If DLPI is not loaded, it displays: `can't find symbol: dl_stats`
-r	Shows the routing tables. When used with the **-s** flag, the**-r** flag shows routing statistics. See Routing Table Display.
-s	Shows statistics for each protocol.
-ss	Displays all the non-zero protocol statistics and provides a concise display.
-u	Displays information about domain sockets.
-v	Shows statistics for CDLI-based communications adapters. This flag causes the **netstat** command to run the statistics commands for the **netstat**, **tokstat**, and **fddistat** commands. No flags are issued to these device driver commands. See the specific device driver statistics command to obtain descriptions of the statistical output.
-Zc	Clear network buffer cache statistics.
-Zi	Clear interface statistics.
-Zm	Clear network memory allocator statistics.
-Zs	Clear protocol statistics. To clear statistics for a specific protocol, use -p <protocol>. For example, to clear TCP statistics, type **netstat -Zs -p tcp**.

Item	Description
-@ ***WparName***	Displays the network statistics associated with workload partition (***WparName***). If no ***WparName*** is specified, then show the network statistics for all workload partitions.

Notes:

1. The **-C**, **-D**, **-c**, **-g**, **-m**, **-M**, **-P**, **-r** , **-v**, and **-Z** flags are not supported in the global environment when used in conjunction with the `-@` *WparName* option.
2. The **-C**, **-D**, **-c**, **-g**, **-m**, **-M**, **-P**, **-r** , **-v**, and **-Z** flags are not supported in system workload partitions.

Default Display

The default display for active sockets shows the following items:

- Local and remote addresses
- Send and receive queue sizes (in bytes)
- Protocol
- Internal state of the protocol

Internet address formats are of the form `host.port` or `network.port` if a socket's address specifies a network but no specific host address. The host address is displayed symbolically if the address can be resolved to a symbolic host name, while network addresses are displayed symbolically according to the `/etc/networks` file.

If a symbolic name for a host is not known or if the **-n** flag is used, the address is printed numerically, according to the address family. Unspecified addresses and ports appear as an * (asterisk).

Interface Display (netstat -i)

The interface display format provides a table of cumulative statistics for the following items:

- Errors
- Collisions

  Note: The collision count for Ethernet interfaces is not supported.
- Packets transferred

The interface display also provides the interface name, number, and address as well as the maximum transmission units (MTUs).

Routing Table Display (netstat -r)

The routing table display indicates the available routes and their statuses. Each route consists of a destination host or network and a gateway to use in forwarding packets.

A route is given in the format *A.B.C.D***/***XX*, which presents two pieces of information. *A.B.C.D* indicates the destination address and *XX* indicates the netmask associated with the route. The netmask is represented by the number of bits set. For example, the route `9.3.252.192/26` has a netmask of `255.255.255.192`, which has 26 bits set.

The routing table contains the following fields:

Item	Description
`WPAR`	Displays the name of the workload partition to which this route belongs. This field is only present when the **-@** flag is used with the **-r** flag. For routes belonging to the global system, `Global` is displayed in this column.

Item	Description
Flags	The **flags** field of the routing table shows the state of the route: **A** An Active Dead Gateway Detection is enabled on the route. **U** Up. **H** The route is to a host rather than to a network. **G** The route is to a gateway. **D** The route was created dynamically by a redirect. **M** The route has been modified by a redirect. **L** The link-level address is present in the route entry. **c** Access to this route creates a cloned route. **W** The route is a cloned route. **1** Protocol specific routing flag #1. **2** Protocol specific routing flag #2. **3** Protocol specific routing flag #3. **b** The route represents a broadcast address. **e** Has a binding cache entry. **l** The route represents a local address. **m** The route represents a multicast address. **P** Pinned route. **R** Host or net unreachable. **S** Manually added. **u** Route usable. **s** The Group Routing stopsearch option is enabled on the route. Direct routes are created for each interface attached to the local host.
Gateway	The **gateway** field for these entries shows the address of the outgoing interface.

Item	Description
Refs	Gives the current number of active uses for the route. Connection-oriented protocols hold on to a single route for the duration of a connection, while connectionless protocols obtain a route while sending to the same destination.
Use	Provides a count of the number of packets sent using that route.
PMTU	Gives the Path Maximum Transfer Unit (PMTU). AIX 5.3 does not display the PMTU column.
Interface	Indicates the network interfaces utilized for the route.
Exp	Displays the time (in minutes) remaining before the route expires.
Groups	Provides a list of group IDs associated with that route.
Netmasks	Lists the netmasks applied on the system.
Route Tree for Protocol Family	Specifies the active address families for existing routes. Supported values for this field are: **1** Specifies the UNIX address family. **2** Specifies the Internet address family (for example, TCP and UDP). For more information on other address families, refer to the **/usr/include/sys/socket.h** file.

When the **-@** flag is used with the **netstat -r** command and no *WparName* parameter is specified, all of the routes in the system's route table are displayed. If the *WparName* parameter is specified and the WPAR-specific routing is enabled for that WPAR, only the routes associated with that WPAR are displayed. If the *WparName* parameter is specified and the WPAR specific routing is disabled for that WPAR, the routes associated with the global system are displayed.

When a value is specified for the *Interval* parameter, the **netstat** command displays a running count of statistics related to network interfaces. This display contains two columns: a column for the primary interface (the first interface found during autoconfiguration) and a column summarizing information for all interfaces.

The primary interface may be replaced with another interface by using the **-I** flag. The first line of each screen of information contains a summary of statistics accumulated since the system was last restarted. The subsequent lines of output show values accumulated over intervals of the specified length.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display routing table information for an Internet interface, enter the following command:

   ```
   netstat -r -f inet
   ```

   This produces the following output:

   ```
   Routing tables
   Destination   Gateway      Flags Refs Use  PMTU If  Exp Groups Netmasks:
   (root node)
   ```

```
(0)0 ffff f000 0
(0)0 ffff f000 0
(0)0 8123 262f 0 0 0 0 0
(root node)

Route Tree for Protocol Family 2:
(root node)
default        129.35.38.47   UG    0  564   -   tr0   -
loopback       127.0.0.1      UH    1  202   -   lo0   -
129.35.32      129.35.41.172  U     4   30   -   tr0   -   +staff
129.35.32.117  129.35.41.172  UGHW  0   13  1492 tr0   30
192.100.61     192.100.61.11  U     1  195   -   en0   -
(root node)

Route Tree for Protocol Family 6:
(root node)
(root node)
```

The `-r -f inet` flags indicate a request for routing table information for all configured Internet interfaces. The network interfaces are listed in the `Interface` column; en designates a Standard Ethernet interface, while `tr` specifies a Token-Ring interface. Gateway addresses are in dotted decimal format.

Note: AIX 5.3 does not display the PMTU column.

2. To display statistics for GRE Protocol, enter the following command:

```
netstat -s -p gre
```

This produces the following output:

```
GRE Interface gre0
        10 number of times gre_input got called
        8 number of times gre_output got called
        0 packets received with protocol not supported
        0 packets received with checksum on
        0 packets received with routing present
        0 packets received with key present
        0 packets received with sequence number present
        0 packets received with strict source route  present
        0 packets received with recursion control present
        0 packets received where reserved0 non-zero
        0 packets received where version non-zero
        0 packets discarded
        0 packets dropped due to network down
        0 packets dropped due to protocol not supported
        0 packets dropped due to error in ip output routine
        0 packets got by NAT
        0 packets got by NAT but not TCP packet
        0 packets got by NAT but with IP options
```

3. To display statistics for the IPv4 over IPv6 tunnel (GIF tunnel), enter the following command:

```
netstat -s -p gif
```

The command produces the following output:

```
GIF Interface gif0
44 total packets received
50 total packets sent
0 packets received with protocol not supported
0 packets received with checksum on
0 packets received with routing present
0 packets received with strict source route present
0 packets received where version non-zero
0 packets discarded
0 packets dropped due to network down
0 packets dropped due to protocol not supported
0 packets dropped due to error in ipv6 output routine
```

4. To display interface information for an Internet interface, enter the following command:

```
netstat -i -f inet
```

This produces the following output:

```
Name Mtu    Network     Address           Ipkts  Ierrs  Opkts  Oerrs  Coll
lo0  16896  Link#1                        5161      0    5193      0     0
lo0  16896  127         localhost         5161      0    5193      0     0
lo0  16896  ::1                           5161      0    5193      0     0
en1  1500   Link#2      8.0.38.22.8.34    221240    0  100284      0     0
en1  1500   129.183.64  infoserv.frec.bul 221240    0  100284      0     0
```

The `-i -f inet` flags indicate a request for the status of all configured Internet interfaces. The network interfaces are listed in the `Name` column; `lo` designates a loopback interface, `en` designates a Standard Ethernet interface, while `tr` specifies a Token-Ring interface.

5. To display statistics for each protocol, enter the following command:

```
netstat -s -f inet
```

This produces the following output:

```
ip:
:
  44485 total packets received
  0 bad header checksums
  0 with size smaller than minimum
  0 with data size < data length
  0 with header length < data size
  0 with data length < header length
  0 with bad options
  0 with incorrect version number
  0 fragments received
  0 fragments dropped (dup or out of space)
  0 fragments dropped after timeout
  0 packets reassembled ok
  44485 packets for this host
  0 packets for unknown/unsupported protocol
  0 packets forwarded
  0 packets not forwardable
  0 redirects sent
  1506 packets sent from this host
  0 packets sent with fabricated ip header
  0 output packets dropped due to no bufs, etc.
  0 output packets discarded due to no route
  0 output datagrams fragmented
  0 fragments created
  0 datagrams that can't be fragmented
  0 IP Multicast packets dropped due to no receiver
  0 successful path MTU discovery cycles
  0 path MTU rediscovery cycles attempted
  0 path MTU discovery no-response estimates
  0 path MTU discovery response timeouts
  0 path MTU discovery decreases detected
  0 path MTU discovery packets sent
  0 path MTU discovery memory allocation failures
  0 ipintrq overflows

icmp:
  0 calls to icmp_error
  0 errors not generated 'cuz old message was icmp
  Output histogram:
    echo reply: 6
  0 messages with bad code fields
  0 messages < minimum length
  0 bad checksums
  0 messages with bad length
  Input histogram:
    echo: 19
  6 message responses generated

igmp:defect
  0 messages received
  0 messages received with too few bytes
  0 messages received with bad checksum
  0 membership queries received
  0 membership queries received with invalid field(s)
  0 membership reports received
  0 membership reports received with invalid field(s)
  0 membership reports received for groups to which we belong
  0 membership reports sent
```

```
tcp:
  1393 packets sent
    857 data packets (135315 bytes)
    0 data packets (0 bytes) retransmitted
    367 URG only packets
    0 URG only packets
    0 window probe packets
    0 window update packets
    170 control packets
  1580 packets received
    790 acks (for 135491 bytes)
    60 duplicate acks
    0 acks for unsent data
    638 packets (2064 bytes) received in-sequence
    0 completely duplicate packets (0 bytes)
    0 packets with some dup. data (0 bytes duped)
    117 out-of-order packets (0 bytes)
    0 packets (0 bytes) of data after window
    0 window probes
    60 window update packets
    0 packets received after close
    0 discarded for bad checksums
    0 discarded for bad header offset fields
  0 connection request
  58 connection requests
  61 connection accepts
  118 connections established (including accepts)
  121 connections closed (including 0 drops)
  0 embryonic connections dropped
  845 segments updated rtt (of 847 attempts)
  0 resends due to path MTU discovery
  0 path MTU discovery terminations due to retransmits
  0 retransmit timeouts
    0 connections dropped by rexmit timeout
  0 persist timeouts
  0 keepalive timeouts
    0 keepalive probes sent
    0 connections dropped by keepalive

udp:
  42886 datagrams received
:
  0 incomplete headers
  0 bad data length fields
  0 bad checksums
  0 dropped due to no socket
  42860 broadcast/multicast datagrams dropped due to no

socket
  0 socket buffer overflows
  26 delivered
  106 datagrams output
```

`ip` specifies the Internet Protocol; `icmp` specifies the Information Control Message Protocol; `tcp` specifies the Transmission Control Protocol; udp specifies the User Datagram Protocol.

Note: AIX 5.3 does not display the PMTU statistics for the IP protocol.

6. To display device driver statistics, enter the following command:

```
netstat -v
```

The `netstat -v` command displays the statistics for each CDLI-based device driver that is up. To see sample output for this command, see the **tokstat** command, the **entstat** command, or the **fddistat** command.

7. To display information regarding an interface for which multicast is enabled, and to see group membership, enter the following command:

```
netstat -a -I interface
```

For example, if an 802.3 interface was specified, the following output will be produced:

```
Name  Mtu  Network Address      Ipkts  Ierrs  Opkts  Oerrs  Coll
et0   1492 <Link>                  0      0      2      0     0
```

```
et0   1492 9.4.37  hun-eth          0      0      2      0     0
                   224.0.0.1
                   02:60:8c:0a:02:e7
                   01:00:5e:00:00:01
```

If instead of **-I** *interface* the flag **-i** is given, then all configured interfaces will be listed. The network interfaces are listed in the Name column; **lo** designates a loopback interface, **et** designates an IEEE 802.3 interface, **tr** designates a Token-Ring interface, while **fi** specifies an FDDI interface.

The address column has the following meaning. A symbolic name for each interface is shown. Below this symbolic name, the group addresses of any multicast groups that have been joined on that interface are shown. Group address 224.0.0.1 is the special *all-hosts-group* to which all multicast interfaces belong. The MAC address of the interface (in colon notation) follows the group addresses, plus a list of any other MAC level addresses that are enabled on behalf of IP Multicast for the particular interface.

8. To display the packet counts in the communication subsystem, enter the following command:

```
netstat -D
```

The following output will be produced:

```
Source                    Ipkts     Opkts     Idrops      Odrops
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
tok_dev0                   720       542       0          0
ent_dev0                   114         4       0          0
                  - - - - - - - - - - - - - - - - - - - - - - - - - - -
Devices Total              834       546       0          0
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
tok_dd0                    720       542       0          0
ent_dd0                    114         4       0          0
                  - - - - - - - - - - - - - - - - - - - - - - - - - - -
Drivers Total              834       546       0          0
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
tok_dmx0                   720       N/A       0          N/A
ent_dmx0                   114       N/A       0          N/A
                  - - - - - - - - - - - - - - - - - - - - - - - - - - -
Demuxer Total              834       N/A       0          N/A
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
IP                         773       767       0          0
TCP                        536       399       0          0
UDP                        229        93       0          0
                  - - - - - - - - - - - - - - - - - - - - - - - - - - -
Protocols Total           1538      1259       0          0
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
lo_if0                     69        69       0          0
en_if0                     22         8       0          0
tr_if0                    704       543       0          1
                  - - - - - - - - - - - - - - - - - - - - - - - - - - -
Net IF Total               795       620       0          1
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
NFS/RPC Client             519       N/A       0          N/A
NFS/RPC Server               0       N/A       0          N/A
NFS Client                 519       N/A       0          N/A
NFS Server                   0       N/A       0          N/A
                  - - - - - - - - - - - - - - - - - - - - - - - - - - -
NFS/RPC Total              N/A       519       0          0
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
(Note:  N/A -> Not Applicable)
```

9. To display detailed data of active sockets, enter the following command:

```
netstat -aon
```

Output similar to the following is displayed:

```
Active Internet connections (including servers)
Proto Recv-Q Send-Q  Local Address          Foreign Address        (state)
tcp4       0      0  *.13                   *.*                    LISTEN
      so_options: (ACCEPTCONN|REUSEADDR)
      q0len:0 qlen:0 qlimit:1000      so_state: (PRIV)
      timeo:0 uid:0
      so_special: (LOCKBALE|MEMCOMPRESS|DISABLE)
      so_special2: (PROC)
```

```
      sndbuf:
             hiwat:16384 lowat:4096 mbcnt:0 mbmax:65536
      rcvbuf:
             hiwat:16384 lowat:1 mbcnt:0 mbmax:65536
             sb_flags: (SEL)
      TCP:
             mss:512

tcp        0      0  *.21                  *.*                    LISTEN

      so_options: (ACCEPTCONN|REUSEADDR)
      q0len:0 qlen:0 qlimit:1000      so_state: (PRIV)
      timeo:0 uid:0
      so_special: (LOCKBALE|MEMCOMPRESS|DISABLE)
      so_special2: (PROC)
      sndbuf:
             hiwat:16384 lowat:4096 mbcnt:0 mbmax:65536
      rcvbuf:
             hiwat:16384 lowat:1 mbcnt:0 mbmax:65536
             sb_flags: (SEL)
      TCP:
      mss:512

...................
...................
```

10. To display the routing table, enter the following command:

```
netstat -rn
```

Output similar to the following is displayed:

```
Routing tables
Destination      Gateway           Flags   Refs     Use  If   PMTU Exp Groups

Route Tree for Protocol Family 2 (Internet):
default          9.3.149.65        UG        0       24  en0     -   -
9.3.149.64       9.3.149.88        UHSb      0        0  en0     -   -  =>
9.3.149.64/27    9.3.149.88        U         1        0  en0     -   -
9.3.149.88       127.0.0.1         UGHS      0        1  lo0     -   -
9.3.149.95       9.3.149.88        UHSb      0        0  en0     -   -
127/8            127.0.0.1         U        11      174  lo0     -   -

Route Tree for Protocol Family 24 (Internet v6):
::1              ::1               UH        0        0  lo0     -   -
```

Note: AIX 5.3 does not display the PMTU column.

The character => at the end of the line means the line is a duplicate route of the route on the next line.

The loopback route (9.3.149.88, 127.0.0.1) and the broadcast routes (with the flags field containing b indicating broadcast) are automatically created when an interface is configured. Two broadcast routes are added: one to the subnet address and one to the broadcast address of the subnet. The presence of the loopback routes and broadcast routes improve performance.

11. To display the routing table of a workload partition named `wpar1`, enter the following command:

```
netstat –rn@ wpar1
```

Output similar to the following is displayed:

```
Routing tables
WPAR Destination    Gateway         Flags      Refs      Use    If   Exp  Groups

Route Tree for Protocol Family 2 (Internet):
wpar1 default        9.4.150.1         UG        1     13936    en1    -    -
wpar1 9.4.150.0      9.4.150.57        UHSb      0         0    en1    -    -  =>
wpar1 9.4.150/24     9.4.150.57        U         0         0    en0    -    -
wpar1 9.4.150.57     127.0.0.1         UGHS      0         0    lo0    -    -
wpar1 9.4.150.255    9.4.150.57        UHSb      0         3    en0    -    -
```

newaliases Command

Purpose

Builds a new copy of the alias database from the mail aliases file.

Syntax

newaliases

Description

The **newaliases** command builds a new copy of the alias database from the **/etc/aliases** file. It must be run each time this file is changed in order for the changes to take effect. Running this command is equivalent to running the **sendmail** command with the **-bi** flag.

Exit Status

Item	Description
0	Exits successfully.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/usr/sbin/newaliases	Contains the **newaliases** command.
/etc/mail html	

newform Command

Purpose

Changes the format of a text file.

Syntax

newform [**-s**] [**-f**] [**-a** [*Number*]] [**-b** [*Number*]] [**-c** [*Character*]] [**-e** [*Number*]] [**-i** [*TabSpec*]] [**-l** [*Number*]] [**-o** [*TabSpec*]] [**-p** [*Number*]] [*File* ...]

Description

The **newform** command takes lines from the files specified by the *File* parameter (standard input by default) and writes the formatted lines to standard output. Lines are reformatted in accordance with the command-line flags in effect.

Except for the **-s** flag, you can enter command-line flags in any order, repeated, and mixed with the *File* parameter. However, the system processes command-line flags in the order you specify. For example, the

-c flag modifies the behavior of the **-a** and **-p** flags, so specify the **-c** flag before the **-p** or **-a** flag for which it is intended. The **-l** (lowercase L) flag modifies the behavior of the **-a**, **-b**, **-e**, and **-p** flags, so specify the **-l** flag before the flags for which it is intended. For example, flag sequences like **-e**15 **-l**60 yield results that are different from **-l**60 **-e**15. Flags are applied to all files specified on the command line.

An exit value of 0 indicates normal execution; an exit value of 1 indicates an error.

Note:

1. The **newform** command normally only keeps track of physical characters; however, for the **-i** and **-o** flags, the **newform** command keeps track of backspaces to line up tabs in the appropriate logical columns.
2. The **newform** command does not prompt you if the system reads a *TabSpec* variable value from standard input (by use of the **-i-** or **-o-** flag).
3. If you specify the **-f** flag, and the last **-o** flag you specified was **-o-** preceded by either an **-o-** or an **-i-**, the tab-specification format line is incorrect.
4. If the values specified for the **-p**, **-l**, **-e**, **-a**, or **-b** flag are not valid decimal numbers greater than 1, the specified value is ignored and default action is taken.

Flags

Item	Description
-a [*Number*]	Adds the specified number of characters to the end of the line when the line length is less than the effective line length. If no number is specified, the **-a** flag defaults to 0 and adds the number of characters necessary to obtain the effective line length. See also the **-c** [*Character*] and **-p** [*Number*] flags.
-b [*Number*]	Truncates the specified number of characters from the beginning of the line if the line length is greater than the effective line length. If the line also contains fewer characters than specified by the *Number* parameter, the entire line is deleted and a blank line is displayed in its place. See also the **-I** [*Number*] flag. If you specify the **-b** flag with no *Number* variable, the default action truncates the number of characters necessary to obtain the effective line length. This flag can be used to delete the sequence numbers from a COBOL program, as follows: `newform -l1-b7 file-name` The **-l1** flag must be used to set the effective line length shorter than any existing line in the file so that the **-b** flag is activated.
-c [*Character*]	Changes the prefix/add character to that specified by the *Character* variable. Default character is a space and is available when specified before the **-a** and **-p** flags.
-e [*Number*]	Truncates the specified number of characters from the end of the line. Otherwise, the flag is the same as the **-b** [*Number*] flag.
-f	Writes the tab-specification format line to standard output before any other lines are written. The displayed tab-specification format line corresponds to the format specified by the final **-o** flag. If no **-o** flag is specified, the line displayed contains the default specification of -8.

Item	Description
-i [*TabSpec*]	Replaces all tabs in the input with the number of spaces specified by the *TabSpec* variable. This variable recognizes all tab specification forms described in the **tabs** command. If you specify a **-** (minus sign) for the value of the *TabSpec* variable, the **newform** command assumes that the tab specification can be found in the first line read from standard input. The default *TabSpec* value is -8. A *TabSpec* value of -0 expects no tabs. If any are found, they are treated as having a value of -1.
-l [*Number*]	Sets the effective line length to the specified number of characters. If no *Number* variable is specified, the **-l** flag defaults to 72. The default line length without the **-l** flag is 80 characters. Note that tabs and backspaces are considered to be one character (use the **-i** flag to expand tabs to spaces). You must specify the **-l** flag before the **-b** and **-e** flags.
-o [*TabSpec*]	Replaces spaces in the input with a tab in the output, according to the tab specifications given. The default *TabSpec* value is -8. A *TabSpec* value of -0 means that no spaces are converted to tabs on output.
-p [*Number*]	Appends the specified number of characters to the beginning of a line when the line length is less than the effective line length. The default action is to append the number of characters that are necessary to obtain the effective line length. See also the **-c** flag.
-s	Removes leading characters on each line up to the first tab and places up to 8 of the removed characters at the end of the line. If more than 8 characters (not counting the first tab) are removed, the 8th character is replaced by an ***** (asterisk) and any characters to the right of it are discarded. The first tab is always discarded. The characters removed are saved internally until all other specified flags are applied to that line. The characters are then added to the end of the processed line.

Note: The values for the **-a**, **-b**, **-e**, **-l** (lowercase L), and **-p** flags cannot be larger than **LINE_MAX** or 2048 bytes.

Examples

To convert from a file with:

- Leading digits
- One or more tabs
- Text on each line

to a file:

- Beginning with the text, all tabs after the first expanded to spaces
- Padded with spaces out to column 72 (or truncated to column 72)
- Leading digits placed starting at column 73

type the following:

```
newform -s -i -l -a -e filename
```

The **newform** command displays the following error message and stops if the **-s** flag is used on a file without a tab on each line.

```
newform: 0653-457 The file is not in a format supported by the -s flag.
```

newgrp Command

Purpose

Changes a user's real group identification.

Syntax

newgrp [**-**] [**-l**] [*Group*]

Description

The **newgrp** command changes a user's real group identification. When you run the command, the system places you in a new shell and changes the name of your real group to the group specified with the *Group* parameter. By default, the **newgrp** command changes your real group to the group specified in the **/etc/passwd** file.

> **Note:** The **newgrp** command does not take input from standard input and cannot be run from within a script.

The **newgrp** command recognizes only group names, not group ID numbers. Your changes only last for the current session. You can only change your real group name to a group you are already a member of. If you are a root user, you can change your real group to any group regardless of whether you are a member of it or not.

> **Note:** When you run the **newgrp** command, the system always replaces your shell with a new one. The command replaces your shell regardless of whether the command is successful or not. For this reason, the command does not return error codes.

Flags

Item	Description
-	Changes the environment to the login environment of the new group.
-l	Indicates the same value as the **-** flag.

Security

Access Control: This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the root user with the **setuid** (SUID) bit set.

Exit Status

If the **newgrp** command succeeds in creating a new shell execution environment, regardless if the group identification was changed successfully, the exit status will be that of the current shell. Otherwise, the following exit value is returned:

Item	Description
>0	An error occurred.

Examples

1. To change the real group ID of the current shell session to `admin`, enter:

```
newgrp admin
```

2. To change the real group ID back to your original login group, enter:

```
newgrp
```

Files

Item	Description
/etc/passwd	Indicates the password file; contains user IDs.

newkey Command

Purpose

Creates a new key in the **/etc/publickey** file.

Syntax

/usr/sbin/newkey [**-h** *HostName*] [**-u** *UserName*]

Description

The **newkey** command creates a new key in the **/etc/publickey** file. This command is normally run by the network administrator on the Network Information Services (NIS) master machine to establish public keys for users and root users on the network. These keys are needed for using secure Remote Procedure Call (RPC) protocol or secure Network File System (NFS).

The **newkey** command prompts for the login password of the user specified by the *UserName* parameter. Then, the command creates a new key pair in the **/etc/publickey** file and updates the **publickey** database. The key pair consists of the user's public key and secret key and is encrypted with the login password of the given user.

Use of this program is not required. Users may create their own keys using the **chkey** command.

You can use the System Management Interface Tool (SMIT) **smit newkey** fast path to run this command.

Flags

Item	Description
-h *HostName*	Creates a new public key for the root user at the machine specified by the *HostName* parameter. Prompts for the root password of this parameter.
-u *UserName*	Creates a new public key for a user specified by the *UserName* parameter. Prompts for the NIS password of this parameter.

Examples

1. To create a new public key for a user, enter:

```
newkey -u john
```

   In this example, the **newkey** command creates a new public key for the user named `john`.

2. To create a new public key for the root user on host `zeus`, enter:

```
newkey -h zeus
```

In this example, the **newkey** command creates a new public key for the root user on the host named zeus.

Files

Item	Description
/etc/publickey	Stores encrypted keys for users.

news Command

Purpose

Writes system news items to standard output.

Syntax

news [**-a** | **-n** | **-s** | *Item ...*]

Description

The **news** command writes system news items to standard output. This command keeps you informed of news concerning the system. Each news item is contained in a separate file in the **/var/news** directory. Most users run the **news** command followed by the **-n** flag each time they log in by including it in their **$HOME/.profile** file or in the system's **/etc/profile** file. Any user having write permission to this directory can create a news item. It is not necessary to have read permission to create a news item.

If you run the **news** command without any flags, it displays every current file in the **/var/news** file, showing the most recent first. This command, used with the **-a** flag, displays all news items. If you specify the **-n** flag, only the names of the unread news items are displayed. Using the **-s** flag displays the number of unread news items. You can also use the *Item* parameter to specify the files that you want displayed.

Each file is preceded by an appropriate header. To avoid reporting old news, the **news** command stores a currency time. The **news** command considers your currency time to be the date the **$HOME/.news_time** file was last modified. Each time you read the news, the modification time of this file changes to that of the reading. Only news item files posted after this time are considered current.

Pressing the Interrupt (Ctrl-C) key sequence during the display of a news item stops the display of that item and starts the next. Pressing the Ctrl-C key sequence again ends the **news** command.

Note: News items can contain multibyte characters.

Flags

Item	Description
-a	Displays all news items, regardless of the currency time. The currency time does not change.
-n	Reports the names of current news items without displaying their contents. The currency time does not change.
-s	Reports the number of current news items without displaying their names or contents. The currency time does not change.

Examples

1. To display the items that have been posted since you last read the news, enter:

```
news
```

2. To display all the news items, enter:

   ```
   news -a | pg
   ```

   All of the news items display a page at a time (| pg), regardless of whether you have read them yet.
3. To list the names of the news items that you have not read yet, enter:

   ```
   news -n
   ```

   Each name is a file in the **/var/news** directory.
4. To display specific news items, enter:

   ```
   news newusers services
   ```

   This command sequence displays news about `newusers` and `services`, which are names listed by the **news -n** command.
5. To display the number of news items that you have not yet read, enter:

   ```
   news -s
   ```
6. To post news for everyone to read, enter:

   ```
   cp schedule /var/news
   ```

   This copies the `schedule` file into the system **/var/news** directory to create the **/var/news/schedule** file. To do this, you must have write permission to the **/var/news** directory.

Files

Item	Description
/usr/bin/news	Contains the **news** command.
/etc/profile	Contains the system profile.
/var/news	Contains system news item files.
$HOME/.news_time	Indicates the date the **news** command was last invoked.

next Command

Purpose

Shows the next message.

Syntax

next [**+***Folder*] [**-header** | **-noheader**] [**-showproc** *CommandString* | **-noshowproc**]

Description

The **next** command displays the number the system will assign to the next message filed in a Message Handler (MH) folder. The **next** command is equivalent to the **show** command with the **next** value specified as the message.

The **next** command links to the **show** program and passes any switches on to the **showproc** program. If you link to the **next** value and call that link something other than **next**, your link will function like the **show** command, rather than like the **next** command.

The **show** command passes flags it does not recognize to the program performing the listing. The **next** command provides a number of flags for the listing program.

Flags

Item	Description
+*Folder*	Specifies the folder that contains the message you want to show.
-header	Displays a one-line description of the message being shown. The description includes the folder name and message number. This is the default.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-noheader	Prevents display of a one-line description of each message being shown.
-noshowproc	Uses the **/usr/bin/cat** file to perform the listing. This is the default.
-showproc *CommandString*	Uses the specified command string to perform the listing.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To see the next message in the current folder, enter:

   ```
   next
   ```

   The system responds with a message similar to the following:

   ```
   (Message schedule: 10)
   ```

   The text of the message is also displayed. In this example, message `10` in the current folder `schedule` is the next message.
2. To see the next message in the `project` folder, enter:

   ```
   next  +project
   ```

   The system responds with the text of the message and a header similar to the following:

   ```
   (Message project: 5)
   ```

Files

Item	Description
$HOME/.mh_profile	Specifies a user's MH profile.
/usr/bin/next	Contains the **next** command.

nfs.clean Command

Purpose

Stops NFS and NIS operations.

Syntax

```
/etc/nfs.clean [-d][-y][-t nfs|nis]
```

Description

The `/etc/nfs.clean` command is used to shut down operations of NFS, NIS, or both. This script is used by the `shutdown` command but can be used to stop operations of only NFS or NIS. By default, all NFS and NIS daemons are stopped.

This command is recommended instead of using `stopsrc -g nfs` since the `nfs.clean` command shuts daemons down in the correct order. The `stopsrc` command has no notion of stopping daemons of a group in the proper order. This can cause problems if the `statd` and `lockd` daemons are running and the `statd` daemon is stopped before the `lockd` daemon.

Flags

Item	Description
-d	Stops only server-specific daemons. Daemons that can run on clients are not stopped.
-y	Stops only server-specific NIS daemons. This flag is presumed if the `-d` flag is used.
-t	Stops only the specified system. If `-t nfs` is specified, only the NFS daemons are stopped. If `-t nis` is specified, only the NIS daemons are stopped.

Exit Status

Item	Description
0	Command completed successfully.
1	Argument error.

Examples

1. To stop all NFS and NIS daemons, type:

   ```
   /etc/nfs.clean
   ```

2. To stop only NFS, type:

   ```
   /etc/nfs.clean -t nfs
   ```

3. To stop only NFS service daemons, type:

   ```
   /etc/nfs.clean -d -t nfs
   ```

Location

/etc/nfs.clean

nfs4cl Command

Purpose

Displays or modifies current NFSv4 statistics and properties.

Syntax

/usr/sbin/nfs4cl [*subcommand*] [*path*] [*argument*]

Description

Use the **nfs4cl** command to display all the fsid information on the client or modify filesystem options of an fsid.

Note: The **nfs4cl** updates affect newly accessed files in the filesystem. An unmount and remount are required to affect all previously accessed files.

Subcommands

resetfsoptions Subcommand

This subcommand resets all the options for the fsid back to the default options.

Note: The **cio** and **dio** options can be reset with the **resetfsoptions** subcommand, but the **cio** and **dio** behavior is not actually turned off until the NFS filesystem is unmounted and then remounted.

setfsoptions Subcommand

This subcommand will take a path and an argument. The path specifies the target fsid structure and the argument is the file system options. It will set the internal fsid to use the options specified by the argument. Here is the list of possible arguments:

Item	Description
rw	Specifies that the files or directories that bind to this path (fsid) are readable and writable.
ro	Specifies that the files or directories that bind to this path (fsid) are read only.
acdirmax	Specifies the upper limit for the directory attribute cache time out value.
acdirmin	Specifies the lower limit for the directory attribute cache time out value.
acregmax	Specifies the upper limit for the file attribute cache time out value.
acregmin	Specifies the lower limit for the file attribute cache time out value.
cio	Specifies the filesystem to be mounted for concurrent readers and writers. I/O on files in this filesystem behave as if the file was opened with **O_CIO** specified in the **open()** system call.
cior	Specifies to allow read-only files to open in the file system. I/O on files in this filesystem will behave as if they had been opened with **O_CIO | O_CIOR** specified in the **open()** system call.
dio	Specifies that I/O on the filesystem behaves as if all of the files were opened with **O_DIRECT** specified in the **open()** system call.
hard	Specifies that this fsid will use hard mount semantics.
intr	Specifies that the fsid operations are interruptible.

Item	Description
maxpout=*value*	Specifies the pageout level for files on this filesystem at which threads should be slept. If **maxpout** is specified, **minpout** must also be specified. This value must be non-negative and greater than **minpout**. The default is the kernel **maxpout** level.
minpout=*value*	Specifies the pageout level for files on this filesystem at which threads should be readied. If **minpout** is specified, **maxpout** must also be specified. This value must be non-negative. The default is the kernel **minpout** level.
noac	Does not use attribute cache.
nocto	Specifies no close-to-open consistency.
nointr	Specifies that the fsid is non-interruptible.
`prefer=`*servername*	Administratively sets the preferred server to use when data exists at multiple server locations. The server name can be in short name, long name, IPv4, or IPv6 format, but the client must be able to resolve the server name when the `nfs4cl` command is run.
rbr	Utilizes the release-behind-when-reading capability. When sequential reading of a file in this filesystem is detected, the real memory pages used by the file will be released once the pages are copied to internal buffers.
rsize	Specifies the read size for the RPC calls to the server.
retrans	Specifies the number of RPC retransmits to attempt with soft semantics.
soft	Specifies the fsid operation that will use soft mount semantics.
timeo	Specifies the time out value for the RPC calls to the server.
wsize	Specifies the write size for the RPC calls to the server.
nodircache	Does not use directory cache.

showfs Subcommand

This subcommand displays filesystem specific information on the server that is currently accessed by the client. The information includes server address, remote path, fsid, and local path. If path is provided, additional information, such as fs_locations and fsid options, are displayed.

showstat Subcommand

This subcommand shows information similar to what the **df** command prints out for each fsid that exists on the client. The information includes fields such as, Filesystem, 512-blocks, Free, %Used, Iused, %Iused, and Mounted on.

delegreturn Subcommand

This subcommand accepts file path as its input argument. This subcommand will allow a system administrator to instruct NFS v4 client to return delegations on the file specified by the input path name.

help Subcommand

This subcommand prints the usage statement.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Examples

1. To display all the fsid structure on the client, type:

```
nfs4cl showfs
```

2. To set the file system options of **/mnt/usr/sbin** to include only `retrans=3`, type:

```
nfs4cl setfsoptions /mnt/usr/sbin retrns=3
```

3. To reset the filesystem options for **/mnt/use/sbin**, type:

```
nfs4cl resetfsoptions /mnt/user/sbin
```

4. To show **df** command output for **/mnt/usr/sbin**, type:

```
nfs4cl showstat /mnt/usr/sbin
```

5. To make the client failover to server boo when replication occurs in **/mnt/usr/sbin**, type:

```
nfs4cl setfsoptions /mnt/usr/sbin prefer=boo
```

Location

/usr/sbin/nfs4cl

nfs4smctl Command

Purpose

Administers revocation of NFSv4 State.

Syntax

`/usr/sbin/nfs4smctl -r` [*hostname* | *IP_address*]

Description

Administers revocation of NFS v4 State.

Flags

Item	Description
`-r` *hostname IP_address*	Specifies the client of which state is to be revoked using either the *hostname* or *IP_address* parameter.

Files

Item	Description
`/usr/sbin/nfs4smctl`	Location of the `nfs4smctl` command.

nfsauthreset Command

Purpose

Notifies the Network File System (NFS) kernel extension to destroy the appropriate Generic Security Service API (GSSAPI) credentials from the kernel credentials cache.

Syntax

nfsauthreset

Description

To mark the cached context, the **nfsauthreset** command depends on whether a Process Authentication Group (PAG) is set in the process. If a PAG is set in the process, it marks the cached GSSAPI context having the same User ID (UID) and PAG to be destroyed. Otherwise, it marks the cached GSSAPI context having the same UID to be destroyed.

Examples

To destroy the cached kernel credentials after you have specified the **kinit** and the **kdestroy** commands, enter:

```
nfsauthreset
```

Files

Item	Description
/usr/sbin/nfsauthreset	Contains the **nfsauthreset** command.

nfsd Daemon

Purpose

Services client requests for file system operations.

Syntax

/usr/sbin/nfsd [**-a** | **-p** { *tcp* | *udp* }] [**-c** *max_connections*] [-gp on | off] [-gpx *count*] [-gpbypass] [**-w** *max_write_size*] [**-r** *max_read_size*] [-root directory] [-public directory] *nservers*

/usr/sbin/nfsd -getnodes

/usr/sbin/nfsd -getreplicas

Description

The **nfsd** daemon runs on a server and handles client requests for file system operations.

Each daemon handles one request at a time. Assign the maximum number of threads based on the load you expect the server to handle.

The **nfsd** daemon is started and stopped with the following System Resource Controller (**SRC**) commands:

```
startsrc -s nfsd
```

```
stopsrc -s nfsd
```

To change the number of daemons started with the SRC commands, use the **chnfs** command. To change the parameters of an SRC controlled daemon, use the **chssys** command.

Note: If the number of nfsd daemons is not sufficient to serve the client, a nonidempotent operation error is returned to the client. For example, if the client removes a directory, an ENOENT error is returned even though the directory on the server is removed.

Flags

Item	Description
-a	Specifies UDP and TCP transport will be serviced.
-c *max_connections*	Specifies the maximum number of TCP connections allowed at the NFS server.
-gp on|off	Controls the NFSv4 Grace Period enablement. The possible values are on or `off`. If no `-gp` option is specified, the grace period is disabled by default.
-gpbypass	Controls the NFSv4 Grace Period bypass. When this option is specified, the grace period will be bypassed regardless of how the `-gp` option is specified.
-gpx *count*	Controls the NFSv4 Grace Period automatic extension. The *count* parameter specifies the total number of automatic extensions allowed for the grace period. If no `-gpx` option is specified, the number of allowed automatic extensions defaults to 1. A single extension cannot extend the grace period for more than the length of the NFSv4 lease period. The NFSv4 subsystem uses runtime metrics (such as the time of the last successful NFSv4 reclaim operation) to detect reclamation of the state in progress, and extends the grace period for a length of time up to the duration of the given number of iterations.
nservers	Specifies the maximum number of concurrent requests that the NFS server can handle. This concurrency is achieved by dynamic management of threads within the NFS server, up to the maximum. The default maximum is 3891. The **chnfs**, **chssys**, or **nfso** command is used to change the maximum. Changing the maximum setting from the default is not recommended as this may limit server performance.
-p *tcp* or **-p** *udp*	Transports both UDP and TCP to the NFS clients (default). You can only specify UDP or TCP. For example, if **-p** *tcp* is used, the NFS server only accepts NFS client requests using the TCP protocol.
-r *max_read_size*	Specifies for NFS Version 3, the maximum size allowed for file read requests. The default and maximum allowed is 64K.
-w *max_write_size*	Specifies for NFS Version 3, the maximum size allowed for file write requests. The default and maximum allowed is 64K.
`-root directory`	Specifies the directory which should be the root node the NFS version 4 exported filesystem. By default, the root node is `/`. If the root node is set to something other than `/`, use `chnfs -r` to reset the node to `/`. This flag may be used while `nfsd` is running to change the root node, but only if no filesystems are currently exported. This flag might be removed in a future release. Use `chnfs -r` instead.
`-public directory`	Specifies the directory which should be the public node of the NFS version 4 exported filesystem. By default, the public node is the same as the root node. This flag may be use while `nfsd` is running to change the public node. The public node must be a descendant of the root node. This flag might be removed in a future release. Use `chnfs -p` instead.
`-getnodes`	Prints the current root and public nodes for the NFS version 4 server. This option will not cause the NFS server daemon to start.
`-getreplicas`	Prints the current replication enablement mode. If replicas have been specified for the `nfsroot`, they will be displayed.

Examples

1. To start **nfsd** daemons using an **src** command, enter:

   ```
   startsrc -s nfsd
   ```

   In this example, the `startsrc -s nfsd` entry starts the number of daemons specified in the script.
2. To change the number of daemons running on your system, enter:

   ```
   chssys -s nfsd -a 6
   ```

   In this example, the `chssys` command changes the number of `nfsd` daemons running on your system to 6.

nfshostkey Command

Purpose

Configures the host keys for an Network File System (NFS) server.

Syntax

`nfshostkey` `-l` | `-L` | {`-p` *principal* `-f` *file*} | { `-a` -p *principal* `-i` *address* } | { `-d` -p *principal* -i *address*}

Description

An NFS server (or full client) using RPCSEC_GSS RPC security must be able to acquire credentials for its host principal to accept requests. Use the **nfshostkey** command to configure this information.

All full clients and NFS servers must have a primary host principal. The following is the format of the host principal that the **nfshostkey** command sets:

```
nfs/fully_qualified_domain_name
```

After you set the primary host principal, you can use the **nfshostkey** command to set additional host principals for other network addresses. The server searches the list of addresses to find the one that an incoming request was sent to and use the appropriate principal. If none is found, the primary principal is used. The secondary host principals must have entries in the same `keytab` file that was passed in for the primary principal. They will not be used by full clients.

Flags

Item	Description
`-a`	Adds a new secondary host principal.
`-d`	Deletes a secondary host principal.
`-f` *file*	Specifies the path to a `keytab` file for the host principals.
`-i` *address*	Specifies the IP address corresponding to the secondary principal.
`-l`	Lists the primary host principal and keytab.
`-L`	Lists the primary host principal, keytab, and secondary host principals.
`-p` *principal*	Specifies the principal for this host.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To set a primary host principal, enter:

```
nfshostkey -p principal -f keytab file
```

2. To add a secondary host principal, enter:

```
nfshostkey -a -p principal -i ip address
```

3. To delete a host principal, enter:

```
nfshostkey -d -p principal -i ip address
```

nfshostmap Command

Purpose

Manage mapping from hosts to principals for an nfs client.

Syntax

/usr/sbin/nfshostmap -a *hostname alias1 alias2* | -d *hostname* | -e *hostname alias1 alias2* | -l

Description

All hosts defined as aliases will be mapped to the host defined as a *hostname* when constructing a kerberos request to the server. This is useful if, for example, a server has interfaces wizard.sub.austin.ibm.com and wizard.austin.ibm.com; if this server's kerberos principal is wizard.austin.ibm.com, nfshostmap -a wizard.austin.ibm.com wizard.sub.austin.ibm.com run on the client will take care of this problem.

This modifies /etc/nfs/princmap, which is read by the gssd daemon on startup.

Flags

Item	Description
-a *hostname alias1 alias2*	Adds a mapping from the aliases to *hostname*,
-d *hostname*	Deletes all aliases for *hostname*.
-e *hostname alias1 alias2*	Removes all previous mappings for *hostname* and replaces them with the given alias list.
-l	Prints the existing state of the respective files on the system.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

nfso Command

Purpose

Manages Network File System (NFS) tuning parameters.

Syntax

nfso [**-p** | **-r**] [**-c**] { **-o** *Tunable*[**=***newvalue*] }

nfso [**-p** | **-r**] { **-d** *Tunable* }

nfso [**-p** | **-r**] **-D**

nfso [**-p** | **-r**] **-a** [**-F**] [**-c**]

nfso -h [*Tunable*]

nfso -l [*hostname*]

nfso [**-F**] **-L** [*Tunable*]

nfso [**-F**] **-x** [*Tunable*]

nfso [**-@** *WparName*] [**-p** | **-r**] **-a** [**-c**]

nfso [**-@** *WparName*] [**-p** | **-r**] [**-c**] { **-o** *Tunable*[**=***newvalue*] }

nfso -H {*ha operation*}

Note: Multiple flags **-o**, **-d**, **-x**, and **-L** are allowed.

Description

Use the **nfso** command to configure Network File System tuning parameters. The **nfso** command sets or displays current or next boot values for Network File System tuning parameters. This command can also make permanent changes or defer changes until the next reboot. Whether the command sets or displays a parameter is determined by the accompanying flag. The **-o** flag performs both actions. It can either display the value of a parameter or set a new value for a parameter.

Understanding the Effect of Changing Tunable Parameters

Extreme care should be taken when using this command. If used incorrectly, the **nfso** command can make your system inoperable.

Before modifying any tunable parameter, you should first carefully read about all its characteristics in the Tunable Parameters section below, and follow any Refer To pointer, in order to fully understand its purpose.

You must then make sure that the Diagnosis and Tuning sections for this parameter truly apply to your situation and that changing the value of this parameter could help improve the performance of your system.

If the Diagnosis and Tuning sections both contain only "N/A", you should probably never change this parameter unless specifically directed by AIX development.

Flags

Item	Description
-a	Displays the current, reboot (when used in conjunction with **-r**) or permanent (when used in conjunction with **-p**) value for all tunable parameters, one per line in pairs *Tunable* **=** *Value*. For the permanent options, a value is only displayed for a parameter if its reboot and current values are equal. Otherwise NONE displays as the value.

Item	Description
-c	Changes the output format of the **nfso** command to colon-delineated format.
-d *Tunable*	Sets the *Tunable* variable back to its default value. If a *Tunable* needs to be changed that is, . it is currently not set to its default value) and is of type Bosboot or Reboot, or if it is of type Incremental and has been changed from its default value, and **-r** is not used in combination, it will not be changed but a warning displays instead.
-D	Sets all *Tunable* variables back to their default value. If *Tunable*s needing to be changed are of type Bosboot or Reboot, or are of type Incremental and have been changed from their default value, and the **-r** flag is not used in combination, they will not be changed but warnings display instead.
-F	Forces restricted tunable parameters to be displayed when the options **-a**, **-L** or **-x** are specified on the command line. If you do not specify the **-F** flag, restricted tunables are not included, unless they are specifically named in association with a display option.
-h [*Tunable*]	Displays help about *Tunable* parameter if one is specified. Otherwise, displays the **nfso** command usage statement.
-H {ha operation}	Runs an high availability (HA) operation. HA operations follow: **enable_ha** Turns on the HA function. **disable_ha** Turns off the HA function. **sm_register <hostname>** PowerHA SystemMirror registers this host. **sm_unregister <hostname>** PowerHA SystemMirror unregisters this host. **sm_gethost** PowerHA SystemMirror gets the host. **dump_dupcache <log device>** Dumps HA dupcache.
-l *hostname*	Allows a system administrator to release NFS file locks on an NFS server. The *hostname* variable specifies the host name of the NFS client that has file locks held at the NFS server. The **nfso -l** command makes a remote procedure call to the NFS server's **rpc.lockd** network lock manager to request the release of the file locks held by the *hostname* NFS client. If there is an NFS client that has file locks held at the NFS server and this client has been disconnected from the network and cannot be recovered, the **nfso -l** command can be used to release those locks so that other NFS clients can obtain similar file locks. **Note:** The **nfso** command can be used to release locks on the local NFS server only.

Item	Description
-L [*Tunable*]	Lists the characteristics of one or all *Tunable*, one per line, using the following format:

```
NAME                      CUR    DEF    BOOT   MIN    MAX
UNIT           TYPE
     DEPENDENCIES
------------------------------------------------------------------------------
------
portcheck                 0      0      0      0      1
On/Off            D
------------------------------------------------------------------------------
------
udpchecksum               1      1      1      0      1
On/Off            D
------------------------------------------------------------------------------
------
nfs_socketsize            600000 600000 600000 40000  1M
Bytes             D
------------------------------------------------------------------------------
------
nfs_tcp_socketsize        600000 600000 600000 40000  1M
Bytes             D
------------------------------------------------------------------------------
------
...
where:
    CUR = current value
    DEF = default value
    BOOT = reboot value
    MIN = minimal value
    MAX = maximum value
    UNIT = tunable unit of measure
    TYPE = parameter type: D (for Dynamic),
           S (for Static), R (for Reboot),B (for Bosboot), M (for Mount),
           I (for Incremental), C (for Connect), and d (for Deprecated)
    DEPENDENCIES = list of dependent tunable parameters, one per line
```

-o *Tunable*[**=***newvalue*]	Displays the value or sets *Tunable* to *newvalue*. If a tunable needs to be changed (the specified value is different than current value), and is of type Bosboot or Reboot, or if it is of type Incremental and its current value is bigger than the specified value, and **-r** is not used in combination, it will not be changed but a warning displays instead. When **-r** is used in combination without a new value, the nextboot value for the *Tunable* displays. When **-p** is used in combination without a *newvalue*, a value displays only if the current and next boot values for the *Tunable* are the same. Otherwise NONE displays as the value.
-p	Makes changes apply to both current and reboot values, when used in combination with **-o**, **-d** or **-D**, that is, it turns on the updating of the **/etc/tunables/nextboot** file in addition to the updating of the current value. These combinations cannot be used on Reboot and Bosboot type parameters because their current value cannot be changed. When used with **-a** or **-o** without specifying a new value, values are displayed only if the current and next boot values for a parameter are the same. Otherwise NONE displays as the value.
-r	Makes changes apply to reboot values when used in combination with **-o**, **-d** or **-D**, that is, it turns on the updating of the **/etc/tunables/nextboot** file. If any parameter of type Bosboot is changed, the user is prompted to run bosboot. When used with **-a** or **-o** without specifying a new value, next boot values for tunables display instead of current values.

Item	Description
-x [*Tunable*]	Lists characteristics of one or all tunables, one per line, using the following (spreadsheet) format: `tunable,current,default,reboot,min,max,unit,type,{dtunable }` `where:` `  current = current value` `  default = default value` `  reboot = reboot value` `  min = minimal value` `  max = maximum value` `  unit = tunable unit of measure` `TYPE = parameter type: D (for Dynamic),` `  S (for Static), R (for Reboot),B (for Bosboot), M (for Mount),` `  I (for Incremental), C (for Connect), and d (for Deprecated)` `  dtunable = space separated list of dependent tunable parameters`
-@ *WparName*	Sets or displays tunables for the specified workload partition. The **-@** flag can only be used when the **nfso** command is run in the global partition.

If you make any change (with **-o**, **-d** or **-D**) to a restricted tunable parameter, it results in a warning message that a tunable parameter of the restricted-use type has been modified. If you also specify the **-r** or **-p** options on the command line, you are prompted for confirmation of the change. In addition, at system reboot, the presence of restricted tunables in the **/etc/tunables/nextboot** file, which were modified to a value that is different from their default value (using a command line specifying the **-r** or **-p** options), results in an error log entry that identifies the list of these modified tunables.

If you make any change (with **-o**, **-d**, or **-D**) to a parameter of type Mount, it results in a warning message that the change is only effective for future mountings.

If you make any change (with **-o**, **-d** or **-D**) to a parameter of type Connect, it results in **inetd** being restarted, and a warning message that the change is only effective for future socket connections.

If you make any change (with **-o**, **-d**, or **-D**) to a parameter of type Bosboot or Reboot without **-r**, it results in an error message.

If you make any change (with **-o**, **-d**, or **-D** but without **-r**) to the current value of a parameter of type Incremental with a new value smaller than the current value, it results in an error message.

Note: Tunable variables that apply to the entire system can not be modified within a workload partition.

Note: When the **nfso** command is run within a workload partition (or if the **-@** flag is specified), only the following tunables can be set with the **-o** flag:

- **nfs_dynamic_retrans**
- **nfs_iopace_pages**
- **nfs_use_reserved_port**
- **nfs_v4_fail_over_timeout**
- **utf8_validation**
- **nfs_auth_rbr_trigger**
- **client_delegation**

Tunable Parameters Type

All the tunable parameters manipulated by the tuning commands (**no**, **nfso**, **vmo**, **ioo**, **schedo**, and `raso`) have been classified into these categories:

Item	Description
Dynamic	If the parameter can be changed at any time
Static	If the parameter can never be changed

Item	Description
Reboot	If the parameter can only be changed during reboot
Bosboot	If the parameter can only be changed by running bosboot and rebooting the machine
Mount	If changes to the parameter are only effective for future file systems or directory mounts
Incremental	If the parameter can only be incremented, except at boot time
Connect	If changes to the parameter are only effective for future socket connections
Deprecated	If changing this parameter is no longer supported by the current release of AIX.

For parameters of type Bosboot, whenever a change is performed, the tuning commands automatically prompt the user to ask if they want to execute the **bosboot** command. For parameters of type Connect, the tuning commands automatically restart the **inetd** daemon.

Note that the current set of parameters managed by the **nfso** command only includes Dynamic, Mount, and Incremental types.

Compatibility Mode

When running in pre 5.2 compatibility mode (controlled by the **pre520tune** attribute of **sys**0, see AIX 5.2 compatibility mode), reboot values for parameters, except those of type Bosboot, are not really meaningful because in this mode they are not applied at boot time.

In pre 5.2 compatibility mode, setting reboot values to tuning parameters continues to be achieved by imbedding calls to tuning commands in scripts called during the boot sequence. Parameters of type **Reboot** can therefore be set without the **-r** flag, so that existing scripts continue to work.

This mode is automatically turned ON when a machine is MIGRATED to AIX 5L Version 5.2. For complete installations, it is turned OFF and the reboot values for parameters are set by applying the content of the **/etc/tunables/nextboot** file during the reboot sequence. Only in that mode are the **-r** and **-p** flags fully functional. See Kernel Tuning in the *Performance Tools Guide and Reference* for details about the new 5.2 mode.

Tunable Parameters

For default values and range of values for tunables, refer the **nfso** command help (**-h** <*tunable_parameter_name*>).

Note: Starting with AIX Version 6.1 with the 6100-02 Technology Level, the following parameters are obsolete because the network file system (NFS) and the virtual memory manager (VMM) dynamically tunes the number of **buf** structures and page device tables (PDTs) based on workload:

- nfs_v2_pdts
- nfs_v2_vm_bufs
- nfs_v3_pdts
- nfs_v3_vm_bufs
- nfs_v4_pdts
- nfs_v4_vm_bufs

Item	Description
client_delegation	**Purpose:** Determine if the NFS version 4 client will accept delegations for open files. **Tuning:** A value of 0 disables delegations. A value of 1 enables delegations.
nfs_hang_log	**Purpose:** Sets the priority at which the NFS mount messages that are hung get logged to the `syslog` log file. **Tuning:** The values are in the range 1 - 7. • 1: LOG_ALERT • 2: LOG_CRIT • 3: LOG_ERR • 4: LOG_WARNING • 5: LOG_NOTICE • 6: LOG_INFO • 7: LOG_DEBUG The default value is 6.
nfs_max_read_size	**Purpose:** Allows the system administrator to control the NFS RPC sizes at the server. **Tuning:** Useful when all clients must have changes in the read sizes, and when the clients cannot be changed. Use the values of the client mount as the default value. The default value is required to reduce the V3 read sizes when the mounts cannot be manipulated directly on the clients, in particular during the NIM installations on networks where the network is dropping packets with the default read sizes. In this case, set the maximum size of 512 KB to a smaller value such that the value works on the network. This parameter is also useful when the network devices are dropping packets and a generic change is desired for communications with the server. The default value is 64 KB and the maximum value is 512 KB.
nfs_max_write_size	**Purpose:** Allows the system administrator to control the NFS RPC sizes at the server. **Tuning:** Useful when all clients must have changes in the write sizes, and when the clients cannot be changed. Use the values of the client mount as the default value. The default value is required to reduce the V3 read sizes when the mounts cannot be manipulated directly on the clients, in particular during the NIM installations on networks where the network is dropping packets with the default write sizes. In this case, set the maximum size of 512 KB to a smaller value such that the value works on the network. This parameter is also useful when the network devices are dropping packets and a generic change is desired for communications with the server. The default value is 64 KB and the maximum value is 512 KB.

Item	Description
nfs_rfc1323	**Purpose:** Enables very large TCP window size negotiation (greater than 65535 bytes) to occur between systems. **Tuning:** If using the TCP transport between NFS client and server, and both systems support it, this allows the systems to negotiate a TCP window size in a way that will allow more data to be in-flight between the client and server. This increases the throughput potential between client and server. Unlike the **rfc1323** option of the no command, this only affects NFS and not other applications in the system. Value of 0 means this is disabled, and value of 1 means it is enabled. If the no command parameter **rfc1323** is already set, this NFS option does not need to be set.
nfs_securenfs_authtimeout	**Purpose:** Sets the number of seconds for which a DES credential. **Tuning:** Value of 0 disables DES credential timeouts.
nfs_server_base_priority	**Purpose:** Sets the base priority of **nfsd** daemons. **Tuning:** By default, the **nfsd** daemons run with a floating process priority. Therefore, as they increase their cumulative CPU time, their priority will change. This parameter can be used to set a static parameter for the **nfsd** daemons. The value of 0 represents the floating priority (default). Other values within the acceptable range will be used to set the priority of the **nfsd** daemon when an NFS request is received at the server. This option can be used if the NFS server is overloading the system (lowering or making the **nfsd** daemon less favored). It can also be used if you want the **nfsd** daemons be one of the most favored processes on the server. Use caution when setting the parameter because it can render the system almost unusable by other processes. This situation can occur if the NFS server is very busy and will essentially lock out other processes from having run time on the server.

Item	Description
nfs_server_clread	**Purpose:** This option allows the NFS server to be very aggressive about the reading of a file. The NFS server can only respond to the specific NFS-read request from the NFS client. However, the NFS server can read data in the file which exists immediately after the current read request. This is normally referred to as read-ahead. The NFS server does read-ahead by default. **Tuning:** May be useful in cases where server memory is low and a lot of disk-to-memory activity is going on. With the **nfs_server_clread** option enabled, the NFS server becomes very aggressive about doing read-ahead for the NFS client. If value is 1, then aggressive read-ahead is done; If value is 0, normal system default read-ahead methods are used. Normal system read-ahead is controlled by VMM (for JFS file systems) and JFS2 (for JFS2 file systems). This more aggressive top-half read-ahead enabled via the nfs_server_clread option is less susceptible to read-ahead breaking down due to out-of-order requests (which are typical in the NFS server case). When the mechanism is activated, it will read an entire cluster (128 KB, the LVM logical track group size) at a time.
nfs_server_close_delay	**Purpose** Determines if the NFS version 4 server must avoid sending an NFS4ERR_DELAY response if the expected delay is not too long. If NFS clients are used that pause applications for a long time when encountering a NFS4ERR_DELAY response from the server, the server attempts to processes the delay on the server by using the **nfs_server_close_delay** option, which avoids pausing the application. **Tuning** A value of 0 turns off this feature. The default value is 0. A value of 1 enables local processing of short delays on the server side.
nfs_use_reserved_ports	**Purpose:** Specifies using non-reserved IP port number. **Tuning:** Value of 0 will use non-reserved IP port number when the NFS client communicates with the NFS server.
nfs_v3_server_readdirplus	**Purpose:** Determines if READDIRPLUS calls are supported by the server. **Tuning:** Value of 0 disables READDIRPLUS processing.

Item	Description
nfs_v4_fail_over_timeout	**Purpose:** Specifies a time out period which the NFS version 4 client operation will fail over to the replica provided by the NFS version 4 server. Measured in seconds. **Tuning:** If value of 0 is specified, the timeout value will be the timeout value for tcp multipled by 4. Values from 1 to 4 are reserved and the NFS version 4 client will treat it as 0. NFS version 4 allows client to fail over to other replica server if the main server is not responding. This value will determine how long a client has to wait for a respond from the server before it switch all the NFS version 4 request for that **fsid** to other replica server.
portcheck	**Purpose:** Checks whether an NFS request originated from a privileged port. **Tuning:** Value of 0 disables the port-checking that is done by the NFS server. A value of 1 directs the NFS server to do port checking on the incoming NFS requests. This is a configuration decision with minimal performance consequences.
server_delegation	**Purpose:** Determine if the NFS version 4 server will issue read delegations for open files. **Tuning:** A value of 0 disables delegation granting. A value of 1 enables delegation granting.
utf8_validation	**Purpose:** Determine if the NFS version 4 client and server will check string data for UTF-8 correctness. **Tuning:** A value of 0 disables the UTF-8 checking. A value of 1 enables the UTF-8 checking.
gss_window	**Purpose:** Enable or disable GSS window size checking. **Tuning:** A value of 0 disables GSS window size checking. A value of 1 enables GSS window size checking. The default value is 1.

Examples

1. To set the **portcheck** tunable parameter to a value of zero, type:

```
nfso -o portcheck=0
```

2. To set the **udpchecksum** tunable parameter to its default value of 1 at the next reboot, type:

```
nfso -r -d udpchecksum
```

3. To print, in colon-delimited format, a list of all tunable parameters and their current values, type:

```
nfso -a -c
```

4. To list the current and reboot value, range, unit, type and dependencies of all tunables parameters managed by the **nfso** command, type:

```
nfso -L
```

5. To display help information on **nfs_tcp_duplicate_cache_size**, type:

```
nfso -h nfs_tcp_duplicate_cache_size
```

6. To permanently turn off **nfs_dynamic_retrans**, type:

```
nfso -p -o nfs_dynamic_retrans=0
```

7. To list the reboot values for all Network File System tuning parameters, type:

```
nfso -r -a
```

8. To list (spreadsheet format) the current and reboot value, range, unit, type and dependencies of all tunables parameters managed by the **nfso** command, type:

```
nfso -x
```

nfsrgyd daemon

Purpose

Services translation requests between names and ids from servers and clients using NFS V4 or RPCSEC-GSS.

Syntax

`nfsrgyd` [`-f`] [`-T` *heartBeatInt*]

Description

The `nfsrgyd` daemon provides a name translation service for NFS servers and clients. This daemon must be running in order to perform translations between NFS string attributes and UNIX numeric identities.

The environment variables `NFS_NOBODY_USER` and `NFS_NOBODY_GROUP` affect the anonymous user and group owner strings used in the name translations. If these environment variables are not set, their default values of `nobody` will be used. They may be set in the file `/etc/environment`, or on the command line before `nfsrgyd` is started.

The local NFS domain must be set before running the `nfsrgyd` daemon. This may be set by using the `chnfsdom` command.

Note: The `nfsrgyd` daemon uses an ephemeral port.

Flags

Item	Description
`-f`	Creates a new process to flush the name translation cache and exits.
`-T`	Specifies the time interval between subsequent LDAP server reconnections. The valid values are 60-3600 seconds. The default value is 300.

Examples

1. The nfsrgyd daemon is started from the /etc/rc.nfs file. Using the following System Resource Controller (SRC) commands, you can start and stop the nfsrgyd daemon:

```
startsrc -s nfsrgyd
stopsrc -s nfsrgyd
```

2. To change the parameters passed to the nfsrgyd daemon using the chssys command, enter:

```
chssys -s nfsrgyd -a "-T 360"
```

Tip: The change does not take effect until the daemon is restarted. The value of the *heartBeatInt* interval will then be persistent after the nfsrgyd daemon is restarted.

Security

Users must have root authority.

Files

Item	Description
/etc/environment	Contains NFS environment variables.

nfsstat Command

Purpose

Displays statistical information about the Network File System (NFS) and Remote Procedure Call (RPC) calls.

Syntax

/usr/sbin/nfsstat [**-@** *WparName*] [**-c**] [**-d**] [**-s**] [**-n**] [**-r**] [**-m** [**-i**]] [-4] [**-z**] [**-t**] [**-b**] [-g]

Description

The **nfsstat** command displays statistical information about the NFS and Remote Procedure Call (RPC) interfaces to the kernel. You can also use this command to reinitialize this information. If no flags are given, the default is the **nfsstat -csnr** command. With this option, the command displays everything, but reinitializes nothing.

RPC Server Information

The server RPC display includes the following fields:

Item	Description
calls	Total number of RPC calls received. This number includes the NFS version 4 calls if the -4 flag is used. Otherwise, only the version 2 and version 3 total is displayed.
badcalls	Total number of calls rejected by the RPC layer. This number includes the NFS version 4 calls if the -4 flag is used. Otherwise, only the version 2 and version 3 total is displayed.
nullrecv	Number of times an RPC call was not available when it was thought to be received.
badlen	Number of RPC calls with a length shorter than a minimum-sized RPC call.
xdrcall	Number of RPC calls whose header could not be XDR decoded.
dupchecks	Number of RPC calls that looked up in the duplicate request cache.

Item	Description
dupreqs	Number of duplicate RPC calls found.

RPC Client Information

Item	Description
calls	Total number of RPC calls made
badcalls	Total number of calls rejected by the RPC layer
badxid	Number of times a reply from a server was received that did not correspond to any outstanding call
timeouts	Number of times a call timed out while waiting for a reply from the server
newcreds	Number of times authentication information had to be refreshed
badverfs	The number of times the call failed due to a bad verifier in the response.
timers	The number of times the calculated time-out value was greater than or equal to the minimum specified timed-out value for a call.
cantconn	The number of times the call failed due to a failure to make a connection to the server.
nomem	The number of times the calls failed due to a failure to allocate memory.
interrupts	The number of times the call was interrupted by a signal before completing.
retrans	The number of times a call had to be retransmitted due to a time-out while waiting for a reply from the server. This is applicable only to RPC over connection-less transports
dupchecks	The number of RPC calls that looked up in the duplicate request cache.
dupreqs	The number of duplicate RPC calls found.

NFS Server Information

The NFS server displays the number of NFS calls received (`calls`) and rejected (`badcalls`), as well as the counts and percentages for the various kinds of calls made.

NFS Client Information

The NFS client information displayed shows the number of calls sent and rejected, as well as the number of times a CLIENT handle was received (`clgets`), the number of times the client handle had no unused entries (`clatoomany`), and a count of the various kinds of calls and their respective percentages.

NFS Registry Daemon Information

The NFS registry daemon display shows the number of requests from the client and server to translate between UID/GID and string names.

-m Information

The **-m** flag displays information about **mount** flags set by **moun**t options, **mount** flags internal to the system, and other **mount** information. See the **mount** command for more information.

The following **mount** options are set by **mount** flags:

Item	Description
auth	Provides one of the following values: **none** No authentication. **unix** UNIX style authentication (UID, GID). **des** des style authentication (encrypted timestamps).
hard	Hard mount.
soft	Soft mount.
intr	Interrupts allowed on hard mount.
nointr	No interrupts allowed on hard mount.
noac	Client is not caching attributes.
rsize	Read buffer size in bytes.
wsize	Write buffer size in bytes.
retrans	NFS retransmissions.
nocto	No close-to-open consistency.
llock	Local locking being used (no lock manager.
grpid	Group ID inheritance.
vers	NFS version.
proto	Protocol.

The following **mount** options are internal to the system:

Item	Description
printed	Not responding message printed.
down	Server is down.
dynamic	Dynamic transfer size adjustment.
link	Server supports links.
symlink	Server supports symbolic links.
readdir	Use **readdir** instead of **readdirplus**.

-t Information

The `-t` flag displays information relating to translation requests of the NFS identity mapping subsystem.

Item	Description
ids_to_strings	The number of id-to-string translation requests.
strings_to_ids	The number of string-to-id translation requests.
resolve_errors	The number of failed translation requests due to missing data.
badowners	The number of failed translation requests due to invalid inputs.

Item	Description
`cache_hits`	The number of translation requests handled by the translation cache.
`cache_misses`	The number of translation requests not handled by the translation cache.
`cache_entries`	The number of entries in the translation cache.
`cache_recycles`	The number of entries in the translation cache that have expired.

Flags

Item	Description
-@ *WparName*	Displays statistics for the specified workload partition. The **-@** flag can only be used when the **nfsstat** command is executed in the global partition. If the **-@** flag is not used when the **nfsstat** command is executed from a workload partition, the statistics for the current workload partition are displayed. If the **-@** flag is not used when the **nfsstat** command is executed from the global partition, the sum statistics of all active workload partitions (and the global partition) are displayed. **Note:** If you use the **-@** *WparName* flag together with the **-m** flag, the **nfsstat** command displays statistics for the global partition instead of the specified workload partition.
-b	Displays additional statistics for the NFS version 4 server.
-c	Displays client information. Only the client side NFS and RPC information is printed. Allows the user to limit the report to client data only. The **nfsstat** command provides information about the number of RPC and NFS calls sent and rejected by the client. To print client NFS or RPC information only, combine this flag with the **-n** or **-r** option.
-d	Displays information related to NFS version 4 delegations.

Item	Description
-g	Displays RPCSEC_GSS information. The RPCSEC_GSS information sections contain: **activegss** Active RPCSEC_GSS contexts **discardgss** Discarded RPCSEC_GSS messages **krb5est** Established krb5 contexts **krb5iest** Established krb5i contexts **krb5pest** Established krb5p contexts **expgss** Expired RPCSEC_GSS contexts **badaccept** gss_accept_sec_context failures **badverify** gss_verify_mic failures **badgetmic** gss_get_mic failures **badwrap** gss_wrap failures **badunwrap** gss_unwrap failures
-m	Displays statistics for each NFS file system mounted along with the server name, mount flags, current read and write sizes, retransmission count, and the timers used for dynamic retransmission. **Note:** If you provide the **-m** option when you use the **nfsstat** command, you always get statistics for the global partition.
-i	When used along with **-m**, it prints the server's ip address as well. This flag is valid only along with the **-m** flag.
-n	Displays NFS information . Prints NFS information for both the client and server. To print only the NFS client or server information, combine this flag with the **-c** and **-s** options.
-r	Displays RPC information.
-s	Displays server information.
-t	Displays statistics related to translation requests of the NFS identity mapping subsystem. To print only the NFS client or server information, combine with the -c and -s options.
-4	When combined with the **-c**, **-n**, **-s**, or **-z** flags, includes information for the NFS version 4 client or server, in addition to the existing NFS version 2 and version 3 data. Without this option, the output is identical to output from the **nfsstat** command in AIX versions prior to version 5.3.
-z	Re-initializes statistics. This flag is for use by the root user only and can be combined with any of the above flags to zero particular sets of statistics after printing them.

Examples

1. To display information about the number of RPC and NFS calls sent and rejected by the client, enter:

```
nfsstat -c
```

2. To display and print the client NFS call-related information, enter:

```
nfsstat -cn
```

3. To display statistics for each NFS mounted file system, enter:

```
nfsstat -m
```

4. To display and print RPC call-related information for the client and server, enter:

```
nfsstat -r
```

5. To display information about the number of RPC and NFS calls received and rejected by the server, enter:

```
nfsstat -s
```

6. To reset all call-related information to zero on the client and server, enter:

```
nfsstat -z
```

   Note: You must have root user authority to use the **-z** flag.

7. To display information about the NFS client statistics for workload partition **abc**, enter:

```
nfsstat -@ abc -cn
```

nice Command

Purpose

Runs a command at a lower or higher priority.

Syntax

nice [**-** *Increment*| **-n** *Increment*] *Command* [*Argument ...*]

Description

The **nice** command lets you run a command at a priority lower than the command's normal priority. The *Command* parameter is the name of any executable file on the system. If you do not specify an *Increment* value the **nice** command defaults to an increment of 10. You must have root user authority to run a command at a higher priority. The priority of a process is often called its nice value.

The nice value can range from -20 to 19, with 19 being the lowest priority. For example, if a command normally runs at a priority of 10, specifying an increment of 5 runs the command at a lower priority, 15, and the command runs slower. The **nice** command does not return an error message if you attempt to increase a command's priority without the appropriate authority. Instead, the command's priority is not changed, and the system starts the command as it normally would.

The nice value is used by the system to calculate the current priority of a running process. Use the **ps** command with the **-l** flag to view a command's nice value. The nice value appears under the **NI** heading in the **ps** command output.

> **Note:** The **csh** command contains a built-in command named **nice**. The **/usr/bin/nice** command and the **csh** command's **nice** command do not necessarily work the same way. For information on the **csh** command's **nice** command, see the **csh** command.

Flags

Item	Description
-*Increment*	Increments a command's priority up or down. You can specify a positive or negative number. Positive increment values reduce priority. Negative increment values increase priority. Only users with root authority can specify a negative increment. If you specify an increment value that would cause the nice value to exceed the range of -20 to 19, the nice value is set to the value of the limit that was exceeded. This flag is equivalent to the **-n** *Increment* flag.
-n *Increment*	This flag is equivalent to the **-***Increment* flag.

Exit Status

If the command specified by the *Command* parameter is started, the exit status of the **nice** command is the exit status of the command specified by the *Command* parameter. Otherwise, the **nice** command exits with one of the following values:

Item	Description
1-125	An error occurred in the **nice** command.
126	The command specified by the *Command* parameter was found but could not be invoked.
127	The command specified by the *Command* parameter could not be found.

Examples

1. To specify a very low priority, enter:

   ```
   nice -n 15 cc -c *.c &
   ```

   This example runs the **cc** command in the background at a lower priority than the default priority set by the **nice** command.

2. To specify a very high priority, enter:

   ```
   nice --10 wall <<end
   System shutdown in 2 minutes!
   end
   ```

   This example runs the **wall** command at a higher priority than all user processes, which slows down everything else running on the system. The <<end and end portions of the example define a *here document*, which uses the text entered before the end line as standard input for the command.

   > **Note:** If you do not have root user authority when you run this command, the **wall** command runs at the normal priority.

3. To run a command at low priority, enter:

   ```
   nice cc -c *.c
   ```

   This example runs the **cc** command at low priority.

   > **Note:** This does not run the command in the background. The workstation is not available for doing other things.

4. To run a low-priority command in the background, enter:

   ```
   nice cc -c *.c &
   ```

   This example runs the **cc** command at low priority in the background. The workstation is free to run other commands while the **cc** command is running. Refer to the **Shells** in *Operating system and device management*for more information on background (asynchronous) processing.

Files

Item	Description
/usr/bin/nice	Contains the **nice** command.

nim Command

Purpose

Performs operations on Network Installation Management (NIM) objects.

Syntax

nim { **-o** *Operation*} [**-F**] [**-t** *Type*] [**-a** *Attribute*=*Value* . . .] *{ObjectName}*

Description

The **nim** command performs an operation on a NIM object. The type of operation that is performed is dependent on the type of object that is specified by the *ObjectName* parameter. Possible operations include initializing environments and managing resources. You can use the **lsnim** command to display the list of supported operations.

Flags

Item	Description
-a *Attribute* **=** *Value* . . .	Assigns the specified value to the specified attribute. Use the **lsnim -q** *Operation* **-t** *Type* command to get a list of valid attributes for a specific operation.
-F	Overrides some safety checks.

Item	Description
-o *Operation*	Specifies an operation to perform on a NIM object. The possible operations are: **activate** Starts a managed system. **allocate** Allocates a resource for use. **alt_disk_install** Performs an alternate disk installation. **alt_disk_mig** Creates a copy of **rootvg** to a free disk (or disks) and simultaneously upgrades it to a new version or release level of AIX. **bos_inst** Performs a BOS installation. **change** Changes an object's attributes. **check** Checks the status of a NIM object. **chwpar** Changes the characteristics of managed workload partitions. **create** Creates an instance of a managed system. **cust** Performs software customization. **deactivate** Stops a managed system. **deallocate** Deallocates a resource. **define** Defines an object. **destroy** Removes an instance of a managed system. **diag** Enables a system to boot a diagnostic image. **dkls_init** Initializes a diskless environment of a system. **dtls_init** Initializes a dataless environment of a system. **fix_query** Lists the fix information for an APAR or keyword. **linux_inst** Installs the Linux operating system on stand-alone clients. **lppchk** Verifies installed filesets on NIM systems and SPOTs. **lppmgr** Eliminates unnecessary software images in an **lpp_source**. **lslpp** Lists licensed program information about an object. **lswpar** Shows the characteristics of managed workload partitions. **maint** Performs software maintenance. **maint_boot** Enables a system to boot in maintenance mode. **reboot** Reboots a NIM client system.

Item	Description
-o *Operation* (Continued)	**remove** Removes an object. **reset** Resets an object's NIM state. **restvg** Performs a **restvg** operation. **select** Includes and excludes group members from operations that are performed on the group. **showlog** Displays a NIM client's installation, boot or customization log, or a SPOT's installation log from the NIM master. **showres** Displays the contents of a NIM resource. **sync** Synchronizes the NIM database with an alternate master. **sync_roots** Synchronizes root directories for diskless and dataless clients for a specific Shared Product Object Tree (SPOT). **syncwpar** Synchronizes the managed workload partition software with the managing system. **takeover** Allows a machine that is configured as an **alternate_master** to take control of the NIM environment. **unconfig** Unconfigures the NIM master fileset. **update** Adds software to an **lpp_source** or removes software from an **lpp_source**. **updateios** Performs software customization and maintenance on a virtual input-output server (VIOS) management server that is of the **vios** or **ivm** type. Use the **lsnim -POt** *Type* command to get a list of the valid operations for a specific type.

Item	Description
-t *Type*	Specifies the type of the NIM object for define operations. The possible types are: **resource types:** **adapter_def** Directory containing secondary adapter definition files. **boot** An internally managed NIM resource that is used to indicate that a boot image is allocated to a client. **bosinst_data** Configure file that is used during base system installation. **devexports** Device exports the file for workload partitions. **dump** Parent directory for client dump files. **exclude_files** Contains files to be excluded from a **mksysb** image. **fb_script** Executable script that is run during the first reboot of a machine. **fix_bundle** Fix (keyword) input file for the **cust** or **fix_query** operation. **home** Parent directory for client **/home** directories. **image_data** Configure file that is used during base system installation. **installp_bundle** **Installp** bundle file. **ios_mksysb** Represents a backup image that is taken from a VIOS management server that is of the **vios** or **ivm** type. **linux_source** Represents the Linux installation media. **log** Captures log data during a network installation. **lpp_source** Source device for optional product images. **mksysb** **mksysb** image. **nas_filer** A network-attached storage (NAS) device. **nim_script** An internally managed NIM resource that is used to indicate that NIM must run a script as a part of a NIM operation. **paging** Parent directory for the paging files of the client. **root** Parent directory for client **/** (root) directories. **resolv_conf** Name server configuration file. **savevg** A **savevg** image. **savewpar** Workload partition backup image. **script** Executable file that is run on a client. **secattrs** Security attributes file for workload partitions.

Item	Description
-t *Type* (Continued)	Specifies the type of the NIM object for define operations. The possible types are: **shared_home** /home directory that is shared by clients. **shared_root** / (root) directory that is shared by clients **spot** Shared Product Object Tree (SPOT) - equivalent to **/usr** file system. **tmp** Parent directory for client **/tmp** directories. **vg_data** Configuration file that is used during volume group restoration. **wpar_spec** Specification file for creating workload partitions. **machine types:** **alternate_master** A system that is reserved as a backup in case the primary NIM master ceases to function properly. **diskless** All file systems and resources remote. **dataless** Local paging, dump; remote **/**,**/usr**; others remote or local. **standalone** Local file systems and resources. **master** System that controls the NIM environment. **wpar** Workload partition hosted by the managing system. **management types:** **bcmm** A blade management module hardware. **cec** A central electronic complex hardware. **hmc** A Hardware Management Console system. **ivm** An integrated virtual management system. **vios** A Virtual I/O Server. **network types:** **tok** Token-Ring network. **ent** Ethernet network. **fddi** FDDI network. **atm** ATM network. **generic** Other TCP/IP networks. **hfi** Host Fabric Interface (HFI) network. **group types:** **mac_group** Group of machines. **res_group** Group of resources.

Security

Access Control: You must have root authority to run the **nim** command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations that are associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

The following examples are grouped by operation.

activate

1. To start the managed wpar1 workload partition, type:

```
nim -o activate wpar1
```

2. To start the managed wpar1 workload partition with additional **startwpar** command flags with verbose output, type:

```
nim -o activate -a cmd_flags="-v" wpar1
```

allocate

1. To allocate resources to a diskless workstation with the name syzygy and SPOT attribute value of spot1, type:

```
nim -o allocate -a spot=spot1 syzygy
```

2. To perform a base system installation on the system that is named krakatoa, resources must be allocated initially by entering:

```
nim -o allocate -a spot=myspot -a lpp_source=images krakatoa
```

   The NIM environment can be initialized to support the installation by performing the **bos_inst** operation, type:

```
nim -o bos_inst krakatoa
```

3. To install the software product, adt, into a standalone system,stand1, given that the installable option, adt, in the **lpp_source**, images, type:

```
nim -o allocate -a lpp_source=images stand1
```

   Then type:

```
nim -o cust -a filesets="adt" stand1
```

4. To install software products into a standalone system, stand1, such that the image for the installable option, adt, in the **lpp_source**, images, and the **installp_bundle**, bundle1, contains the name of the installable option, type:

```
nim -o allocate -a lpp_source=images \
-a installp_bundle=bundle1 stand1
```

   Then type:

```
nim -o cust stand1
```

5. To automatically configure a machine with name resolution services after a BOS installation, create the /exports/resolv.conf file, with contents similar to the following:

```
nameserver      129.35.143.253
nameserver      9.3.199.2
domain          austin.ibm.com
```

   then type:

```
nim -o define -t resolv_conf -a location=/exports/resolv.conf \
-a server=master rconf1
```

Prior to issuing the **bos_inst** operation, allocate this resource with other required and optional resources by typing:

```
nim -o allocate -a spot=spot1 -a lpp_source=images1 \
-a bosinst_data=bid1 -a resolv_conf=rconf1 client1
```

6. To allocate all resources applicable to standalone machines from the NIM resource group `res_grp1`, to the machine `mac1`, type:

```
nim -o allocate -a group=res_grp1 mac1
```

alt_disk_install

1. To install a **mksysb** resource `all_devices_mysysb` to client `roundrock`, on hdisk4 and hdisk5, using the **image_data** resource `image_data_shrink`, with debug turned on, type:

```
nim -o alt_disk_install -a source=mksysb\
-a image_data=image_data_shrink\
-a debug=yes\
-a disk='hdisk4 hdisk5' roundrock
```

2. To clone a **rootvg** on client `austin` to hdisk2, but only run phase1 and phase2 (leaving the **/alt_inst** file systems mounted), type:

```
nim -o alt_disk_install -a source=rootvg\
-a disk='hdisk2'\
-a phase=12 austin
```

bos_inst

1. To install the machine `blowfish`, using the resources `spot1`, `images1`, `bosinst_data1`, and `rconf1`, first allocate the resources by typing:

```
nim -o allocate -a spot=spot1 -a lpp_source=images1 \
-a bosinst_data=bosinst_data1 -a resolv_conf=rconf1 blowfish
```

Then, perform the BOS installation by typing:

```
nim -o bos_inst blowfish
```

2. To install the machine `blowfish` while allocating the resources `spot1`, `images1`, `bosinst_data1`, and `rconf1` automatically when the **bos_inst** operation starts, type:

```
nim -o bos_inst -a spot=spot1 -a lpp_source=images1 \
-a bosinst_data=bosinst_data1 -a resolv_conf=rconf1 blowfish
```

3. To use the default resources when installing the machine `mac1`, type:

```
nim -o bos_inst mac1
```

4. To install a machine, `deadfish`, with `spot1` and `lpp_source1` and use an **adapter_def** resource, `adapter_def1`, to configure secondary adapters, type:

```
  nim -o bos_inst -a spot=spot1 -a lpp_source=lpp_source1 \
  -a adapter_def=adapter_def1 deadfish
```

5. To install the machine blowfish and accept software license agreements, type:

```
nim -o bos_inst -a spot=spot1 -a lpp_source=images1 \
-a accept_licenses=yes -a resolv_conf=rconf1 blowfish
```

change

1. Machines on the BLDG905 network use the gateway905 gateway to reach the 0Z network. Machines on the 0Z network use the gateway0Z gateway to reach the BLDG905 network. To add a route between two networks named BLDG905 and 0Z, type:

```
nim -o change -a routing1="0Z gateway905 gateway0Z" BLDG905
```

2. The adapter that is identified by the host name sailfish2.austin.ibm.com is attached to a token ring network. To define a secondary interface for this adapter on the NIM master and instructing NIM to locate the NIM network representing the attached ethernet network and, if not found, have NIM define a NIM network with subnetmask 255.255.255.128, type:

```
nim -o change -a if2="find_net sailfish2.austin.ibm.com 0" \
-a net_definition="tok 255.255.255.128" -a ring_speed2=16 master
```

   Note: A default name is generated for the network, and no routing information is specified for the new network.

3. To define default routes for the networks net1 and net2 that use default gateways gw1 and gw2 respectively, type the following two commands:

```
nim -o change -a routing1="default gw1" net1
nim -o change -a routing1="default gw2" net2
```

4. To designate the resources that are defined by the resource group res_grp1 as the set of resources that are always allocated by default during any operation in which these resources are applicable, type:

```
nim -o change -a default_res=res_grp1 master
```

check

1. To have NIM check on the usability of a SPOT named myspot, type:

```
nim -o check myspot
```

2. To check the status of an **lpp_source** named images, type:

```
nim -o check images
```

chwpar

To add rset rs/cpus23 to the resource control attributes for the wpar1 workload partition, type:

```
nim -o chwpar -a cmd_flags="-R rset=rs/cpu23" wpar1
```

create

1. To create the wpar1 workload partition with host name and specification file resource basic_wpar, type:

```
nim -o create -a wpar_spec=basic_wpar wpar1
```

2. To create the wpar1 workload partition with the wpar-specification file resource wpar1_spec, type:

```
nim -o create -a wpar_spec=wpar1_spec wpar1
```

3. To create the wpar1 workload partition from the **savewpar** backup image resource wpar1_backup, type:

```
nim -o create -a savewpar=wpar_backup wpar1
```

cust

1. To install a software product into a spot, spot1, such that the image for the installable option,adt, resides in the **lpp_source**, images, type:

```
nim -o cust -a lpp_source=images -a filesets=adt spot1
```

2. To install a software product into a spot, spot1, such that the image for the installable option,adt, resides in the **lpp_source**,images, and the **installp_bundle**, bundle1, contains the name of the installable option, type:

```
nim -o cust -a lpp_source=images -a installp_bundle=bundle1 spot1
```

3. To install a software product into a spot, spot1, such that the image for the installable option,adt, resides on a tape that is in the tape drive that is local to the machine where the spot resides, type:

```
nim -o cust -a lpp_source=/dev/rmt0 -a filesets=adt spot1
```

4. To install a software product into a spot, spot1, such that the image for the installable option, adt, resides on a tape that is in the tape drive that is local to the machine where the spot resides, type:

```
nim -o cust -a lpp_source=/dev/rmt0 -a filesets=adt spot1
```

5. To install all fileset updates associated with APAR IX12345, residing on the tape **/dev/rmt0** into spot1 and any diskless or dataless clients to which spot1 is currently allocated, type:

```
nim -F -o cust -afixes=IX12345 -a lpp_source=/dev/rmt0 spot1
```

6. To update all software installed on the client Standalone1, with the latest updates in the **lpp_source** named updt_images, type:

```
nim -o allocate -a lpp_source=updt_images Standalone1
nim -o cust -afixes=update_all Standalone1
```

7. To install the machine catfish with the contents of the **installp_bundle** bundle1, first allocate the resources by typing:

```
nim -o allocate -a installp_bundle=bundle1 \
-a lpp_source=images1 catfish
```

   Then, perform the cust operation by typing:

```
nim -o cust catfish
```

8. To update all software that is installed on the client Standalone1, with the latest updates in the **lpp_source** named updt_images, type:

```
nim -o cust -a lpp_source=updt_images -a fixes=update_all \
Standalone1
```

9. To install the machine catfish with the contents of the **installp_bundle** bundle1, while allocating this resource and the **lpp_source** images1 when the **cust** operation runs, type:

```
nim -o cust -a installp_bundle=bundle1 -a lpp_source=images1 \
catfish
```

10. To configure secondary adapters on a client machine, deadfish, by using the secondary adapter configuration file in the **adaper_def** resource, adapter_def1, type:

```
  nim -o cust -a adapter_def=adapter_def1 deadfish
```

deactivate

1. To stop the managed wpar1 workload partition, type:

```
nim -o deactivate wpar1
```

2. To force the stop of the managed wpar1 workload partition, type:

```
nim -Fo deactivate wpar1
```

3. To stop the managed wpar1 workload partition with more **stopwpar** command flags to halt after 85 seconds, type:

```
nim -o deactivate -a cmd_flags="-t 85" wpar1
```

deallocate

To deallocate an **lpp_source** named images from the standalone machine client1, type:

```
nim -o deallocate -a lpp_source=images client1
```

define

1. To define a resource that is a directory that contains installable images that is on the server altoid and has a path name of /usr/sys/inst.images, and name that resource images, type:

```
nim -o define -t lpp_source -a server=altoid \
-a location=/usr/sys/inst.images images
```

2. To create a new SPOT resource named myspot on the NIM master in the /export/exec directory, by using an **lpp_source** named image, type:

```
nim -o define -t spot -a server=master -a location=/export/exec \
-a source=images myspot
```

3. To define a network object named BLDG905, with a subnetmask of 255.255.240.0 and an address of 129.35.129.0, type:

```
nim -o define -t tok -a snm=255.255.240.0 \
-a net_addr=129.35.129.0 BLDG905
```

4. To define a **mksysb** resource, mksysb1, from an existing mksysb image that is located in /resources/mksysb.image on the master, type:

```
nim -o define -t mksysb -a server=master \
-a location=/resources/mksysb.image mksysb1
```

5. To define a NIM network named ATMnet with a subnet mask of 255.255.240 and an address of 129.35.101.0 to represent an ATM network, use the generic network type as follows:

```
nim -o define -t generic -a snm=255.255.240.0 \
-a net.addr=129.35.101.0 ATMnet
```

6. To define a machine group named DisklsMacs1 with members that are NIM diskless machines named diskls1, diskls2, and diskls3, type:

```
nim -o define -t mac_group -a add_member=diskls1 \
-a add_member=diskls2 -a add_member=diskls3 DisklsMacs1
```

7. To define a resource group named DisklsRes1 with resources spot1, root1, dump1, paging1, home1, tmp1, type:

```
nim -o define -t res_group -a spot=spot1 -a root=root1 \
-a dump=dump1 -a paging=paging1 -a home=home1 -a tmp=tmp1 \
DisklsRes1
```

8. To display the space that is required to define a **mksysb** resource, mksysb2, and create a mksysb image of the client, client1, during the resource definition where the image is located in /resources/mksysb.image on the master, type:

Note: This action shows the space that is required for the operation, **mksysb**, or resource creation does NOT take place.

```
nim -o define -t mksysb -a server=master \
-a location=/resources/mksysb.image -a source=client1 \
-a mk_image=yes -a size_preview=yes mksysb2
```

9. To define a **mksysb** resource, `mksysb2`, and create the mksysb image of the client, `client1`, during the resource definition where the image is in `/resources/mksysb.image` on the master, type:

```
nim -o define -t mksysb -a server=master \
-a location=/resources/mksysb.image -a source=client1 \
-a mk_image=yes mksysb2
```

10. To define a **mksysb** resource, `mksysb2`, and create a mksysb image of the client, `client1`, during the resource definition where the mksysb flags used to create the image are **-em**, and the image is in `/resources/mksysb.image` on the master, type:

```
nim -o define -t mksysb -a server=master \
-a location=/resources/mksysb.image -a source=client1 \
-a mk_image=yes -a mksysb_flags=em mksysb2
```

11. To define an **exclude_files** resource, `exclude_file1`, located in `/resources/mksysb.image` on the master, type:

```
nim -o define -t exclude_files -a server=master \
-a location=/resources/exclude_file1 exclude_file1
```

12. A machine that is called `redfish`, hostname `redfish_t.lab.austin.ibm.com`, has its primary interface that is attached to a token-ring network with ring speed of 16 Megabits. To define `redfish` as a standalone machine in the NIM environment and instructing NIM to locate the name of the network that the machine's primary interface is attached, type:

```
nim -o define -t standalone  -a if1="find_net \
redfish_t.lab.austin.ibm.com 0" -a ring_speed1=16 redfish
```

13. A machine that is called `bluefish`, hostname is `bluefish_e.lab.austin.ibm.com`, has its primary interface that is attached to an ethernet network with **cable_type** of **bnc**. To define `bluefish` as a diskless machine in the NIM environment and instructing NIM to locate the name of the network that the machine's primary interface is attached, and if not found, have NIM define a NIM network with the name `ent_net`, subnetmask of `255.255.255.128` and default route by using the gateway with hostname `lab_gate`, type:

```
nim -o define -t diskless -a if1="find_net \
bluefish_e.lab.austin.ibm.com 0" -a net_definition="ent \
255.255.255.128 lab_gate 0 ent_net" -a cable_type=bnc bluefish
```

    Note: Specify 0 in place of the master gateway in the **net_definition** attribute if a default route for the master exists, otherwise you must specify the master gateway.

14. To define the **/export/nim/adapters** directory as an **adapter_def** resource, `adapter_def1`, on the master, type:

```
  nim -o define -t adapter_def -a server=master \
  -a location=/export/nim/adapters adapter_def1
```

    To populate the **adapter_def** resource with secondary adapter configuration files, run the **nimadapters** command.

15. To display the space that is required to define a **savevg** resource, `savevg2`, and create a **savevg** image of the client, `client1`, during the resource definition where the image is in `/export/nim/savevg` on the master and the **volume_group** to to backup is `myvg`, type:

```
nim -o define -t savevg -a server=master \
  -a location=/export/nim/savevg/savevg2 -a source=client1 \
  -a mk_image=yes -a size_preview=yes -a volume_group=myvg savevg2
```

    Note: This action shows the space that is required for the operation. **savevg** or resource creation does not take place.

16. To define a **savevg** resource, savevg2, and create the **savevg** image of the client, client1, during the resource definition where the image is in /export/nim/savevg on the master and the **volume_group** to backup is myvg, type:

```
nim -o define -t savevg -a server=master \
-a location=/export/nim/savevg -a source=client1 \
-a mk_image=yes -a volume_group=myvg savevg2
```

17. To define a **savevg** resource, savevg2, and create a **savevg** image of the client, client1, during the resource definition where the **savevg** flags used to create the image are **-em**, and the image is in /export/nim/savevg on the master, type:

```
nim -o define -t savevg -a server=master \
-a location=/export/nim/savevg -a source=client1 \
-a mk_image=yes -a volume_group=myvg -a savevg_flags=em savevg2
```

18. To define a **vg_data** resource, my_vg_data, on the master at the location /export/nim, type:

```
nim -o define -t vg_data -a server=master -a location=/export/nim/my_vg_data my_vg_data
```

19. To define the wpar1 workload partition that is managed by the yogi managing standalone machine with wpar1 as both the host name and the name of the workload partition on the managing system, type:

```
nim -o define -t wpar -a mgmt_profile1="yogi wpar1" -a if1="find_net wpar1 0" wpar1
```

20. To define a **savewpar** resource named wpar1backup and create the **savewpar** image of the yogi workload partition on the sterling server, type:

```
nim -o define -t savewpar \
   -a server=sterling -a location=/resources/wpar1.image \
   -a source=wpar1 -a mk_image=yes wpar1backup
```

21. To define a **savewpar** resource named wpar1backup and create the **savewpar** image of the yogi workload partition on the sterling server, excluding file patterns in the **exclude_files** resource wparexclude, and passing the flag to the **savewpar** resource to exclude files and creates a **image.data** file, type:

```
nim -o define -t savewpar \
   -a server=sterling -a location=/resources/wpar1.image -a source=wpar1 \
   -a exclude_files=wparexclude -a cmd_flags="-ei" mk_image=yes wpar1backup
```

22. To define a **ios_mksysb** resource such as **ios_mksysb1**, and create the **ios_mksysb** image of the **vios** client as **vios1**, during the resource definition where the image is located in /export/nim/ios_mksysb on the master, type:

```
nim -o define -t ios_mksysb -a server=master \
-a location=/export/nim/ios_mksysb -a source=vios1 \
-a mk_image=yes ios_mksysb1
```

destroy

1. To remove the managed wpar1 workload partition from its managing system, type:

```
nim -o destroy wpar1
```

2. To force the removal of the managed wpar1 workload partition, type:

```
nim -Fo destroy wpar1
```

dkls_init

1. To initialize the environment for a diskless workstation with the name of syzygy, by using the resources spot1, root1, dump1, and paging1, you must allocate the resources by typing:

```
nim -o allocate -a spot=spot1 -a root=root1 -a dump=dump1 \
-a paging=paging1 syzygy
```

Then initialize the resources for the client machine by typing:

```
nim -o dkls_init syzygy
```

2. To initialize the environment for a diskless workstation with the name of `syzygy`, type:

```
nim -o dkls_init syzygy
```

3. To exclude the member named `diskls2` from operations on the machine group `DisklsMacs1`, and then initialize the remaining members while allocating the diskless resources defined by the resource group named `DisklsRes1`, type the following two commands:

```
nim -o select -a exclude=diskls2 DisklsMacs1
nim -o dkls_init -a group=DisklsRes1 DisklsMacs1
```

4. To initialize the group of diskless machines that are defined by the machine group `dtgrp1`, while allocating the required and optional resources defined by the resource group `dk_resgrp1`, when the **dkls_init** operation runs, type:

```
nim -o dkls_init -a group=dtgrp1 dk_resgrp1
```

dtls_init

1. To initialize the environment for a dataless workstation with the name of `syzygy`, using the resources `spot1`, `root1`, and `dump1`, first allocate the resources by typing:

```
nim -o allocate -a spot=spot1 -a root=root1 -a dump=dump1 syzygy
```

Then initialize the resources for the client machine by typing:

```
nim -o dtls_init syzygy
```

2. To initialize the environment for a dataless workstation with the name of `syzygy`, type:

```
nim -o dtls_init syzygy
```

3. To exclude the member named `dataless1` from operations on the machine group `DatalsMacs1`, and then initialize the remaining members while allocating the dataless resources defined by the resource group named `DatalsRes1`, type the following two commands:

```
nim -o select -a exclude=datals2 DatalsMacs1
nim -o dtls_init -a group=DatalsMacs1 DatalsRes1
```

4. To initialize the group of dataless machines defined by the machine group `DatalsMacs1`, while allocating the required and optional resources defined by the resource group `DatalsRes1`, when the **dtls_init** operation runs, type:

```
nim -o dtls_init -a group=DatalsMacs1 DatalsRes1
```

fix_query

To list information about fixes installed on client `Standalone1` for 20 APAR numbers, create the file `/tmp/apar.list` with one APAR number per line, as shown:

```
IX123435
IX54321
IX99999
...
```

then type:

```
nim -o define -t fix_bundle -alocation=/tmp/apar.list \
                -aserver=master fix_bun
nim -o allocate -a fix_bundle=fix_bun Standalone1
nim -o fix_query Standalone1
```

lppchk

1. To check fileset version and requisite consistency on the SPOT spot1, type:

   ```
   nim -o lppchk spot1
   ```

2. To verify the file checksums for all packages beginning with the name bos on NIM targets in the group of standalone machines macgrp1, and displaying detailed error information and updating the software database to match the actual file checksum when inconsistencies are found, type:

   ```
   nim -o lppchk -a lppchk_flags='-c -m3 -u' \
   -a filesets='bos*' macgrp1
   ```

   Because the **lppchk** operation runs in the background on group members by default, to view the output from the **lppchk** operation type:

   ```
   nim -o showlog -a log_type=lppchk macgrp1
   ```

lppmgr

1. To list the names of duplicate base level filesets which should be removed from lpp_source1 with space usage information, type:

   ```
   nim -o lppmgr -a lppmgr_flags="-lsb" lpp_source1
   ```

2. To remove duplicate base and update filesets and superseded updates from lpp_source1, type:

   ```
   nim -o lppmgr -a lppmgr_flags="-rbux" lpp_source1
   ```

3. To remove all non-SIMAGES (filesets that are not required for a bos install) from lpp_source1, type:

   ```
   nim -o lppmgr -a lppmgr_flags="-rX" lpp_source1
   ```

4. To remove all language support except 'C' from lpp_source1, type:

   ```
   nim -o lppmgr -a lppmgr_flags="-r -k C" lpp_source1
   ```

lswpar

1. To list the characteristics of the managed wpar1 workload partition, type:

   ```
   nim -o lswpar wpar1
   ```

2. To list the network characteristics of the managed wpar1 workload partition, type:

   ```
   nim -o lswpar -a cmd_flags="-N" wpar1
   ```

3. To list the general characteristics of the workload partitions managed by the global1 standalone system, type:

   ```
   nim -o lswpar -a cmd_flags="-G" global1
   ```

maint

1. To cleanup from an interrupted software installation on a spot, spot1, type:

   ```
   nim -o maint -a installp_flags="-C" spot1
   ```

2. From the master, to clean up from an interrupted software installation on a standalone machine, stand1, type:

```
nim -o maint -a installp_flags="-C" stand1
```

maint_boot

To enable the NIM standalone client, `stand1`, to boot in maintenance mode, type:

```
nim -o maint_boot stand1
```

This sets up the maintenance boot operation, but you must initiate the network boot locally from `stand1`.

remove

To remove a resource named `dump_files`, type:

```
nim -o remove dump_files
```

showlog

To view the boot logs of the machines that are defined by the group `DisklsMacs1`, type:

```
nim -o showlog -a log_type=boot DisklsMacs1
```

showres

1. To show the contents of the configure `script1` script , type:

```
nim -o showres script1
```

2. To show the contents of the bosinst.data resource `bosinst_data1`, type:

```
nim -o showres bosinst_data1
```

3. To list all the filesets in the lpp_source `lpp_source1`, type:

```
nim -o showres lpp_source1
```

4. To list all the filesets in the lpp_source `lpp_source1` relative to what is installed on the machine `machine1`, type:

```
nim -o showres -a reference=machine1 lpp_source1
```

5. To list all the problems that are fixed by software on the lpp_source `lpp_source1`, use:

```
nim -o showres -a instfix_flags="T" lpp_source1
```

6. To show the contents of the secondary adapter configuration file in the **adapter_def** resource, `adapter_def1`, for client, `deadfish`, type:

```
nim -o showres -a client=deadfish adapter_def1
```

7. To show the contents of every secondary adapter configuration file in the **adapter_def** resource, `adapter_def1`, type:

```
nim -o showres adapter_def1
```

8. To show the contents of the **savevg** resource, `savevg1`, type:

```
nim -o showres savevg1
```

syncwpar

1. To synchronize the software of the managed `wpar1` workload partition with its managing system, type:

```
nim -o syncwpar wpar1
```

2. To synchronize the software of all the workload partitions managed by the `global1` standalone system, type:

```
nim -o syncwpar -a cmd_flags="-A" global1
```

update

1. To add all the filesets on `/dev/cd0` to `lpp_source1`, type:

```
nim -o update -a packages=all -a source=/dev/cd0 lpp_source1
```

2. To add the `bos.games 7.1.0.0` and `bos.terminfo` filesets to `lpp_source1`, type:

```
nim -o update -a packages="bos.games 7.1.0.0 bos.terminfo" \
  -a source=/dev/cd0 lpp_source1
```

3. To remove `bos.games` from `lpp_source1`, type:

```
nim -o update -a rm_images=yes -a packages="bos.games" lpp_source1
```

4. To recover the missing SIMAGES for `lpp_source1` from the AIX Installation CD, type:

```
nim -o update -a recover=yes -a source=/dev/cd0 lpp_source1
```

updatios

1. To install fixes or to update VIOS with the *vioserver1* NIM object name to the latest maintenance level, type:

```
nim -o updateios -a lpp_source=lpp_source1 -a preview=no vioserver1
```

   The updates are stored in `lpp_source` and `lpp_source1` files.

   Note: The **updateios** operation runs a preview during installation. Running the **updateios** operation from NIM runs a preview unless the preview flag is set to no. During the installation, you must run a preview when you use the **updateios** operation with **updatios_flags=-install**. With the preview, you can check whether the preview installation is running accurately before you proceed with the VIOS update.

2. To reject fixes for a VIOS with the *vioserver1* NIM object name, type:

```
nim -o updateios -a updateios_flags=-reject vioserver1
```

3. To clean up partially installed updates for a VIOS with the *vioserver1* NIM object name, type:

```
nim -o updateios -a updateios_flags=-cleanup vioserver1
```

4. To commit updates for a VIOS with the *vioserver1* NIM object name, type:

```
nim -o updateios -a updateios_flags=-commit vioserver1
```

5. To remove a specific update such as `update1` for a VIOS with the *vioserver1* NIM object name, type:

```
nim -o updateios -a updateios_flags=-remove-a filesets="update1" vioserver1
```

6. To remove updates for a VIOS with the *vioserver1* NIM object name by using an *installp_bundle bundle1*, where *bundle1* contains the updates to be removed, type:

```
nim -o updateios -a updateios_flags=remove -a installp_bundle=bundle1 vioserver1
```

Files

Item	Description
/etc/niminfo	Contains variables that are used by NIM.

nim_clients_setup Command

Purpose

Define clients and initialize BOS install operation on NIM client objects.

Syntax

`nim_clients_setup` [`-m` *mksysb_resource*] [`-n`] [`-c`] [`-r`] [`-v`] *client_object(s)*

Description

The `nim_clients_setup` command defines new client objects and initializes the BOS install operation for clients in the NIM environment by performing the following tasks:

- Exports the environment variable NIM_LICENSE_ACCEPT=yes.
  - Used for accepting software license agreement during network install.
- Adds variable entry NSORDER=local,bind in `/etc/environment`.
  - Necessary for name resolution when hosts only exist in `/etc/host`.
- Defines client objects using `client.defs` file (if `-c` flag specified).
  - User must edit stanzas in `/export/nim/client.defs` file prior to using `nim_clients_setup`.
- Prepares client objects for install.
  - If `-c` flag is used, defined clients are initialized for install.
  - If client objects are given, specified clients are initialized for install.
  - If `-c` or client objects are omitted, all existing NIM clients are initialized for install.
- Resources contained in the group name `basic_res_grp` are used as resources during the BOS install operation.

Note: The`basic_res_grp` resource group is populated with resources created during `nim_master_setup` command execution. If this group is not present, it must be defined with NIM install resources prior to using the `nim_clients_setup` command.

Flags

Item	Description
`-m` *mksysb_resource*	Specifies an alternate backup image to restore during BOS install. The value for *mksysb_resource* may specify a NIM object name or absolute path location used for defining a new `mksysb` resource. By default, the `mksysb` resource is assigned from the `basic_res_grp` NIM resource group.
`-n`	Enables native (`rte`) install and ignores restoring backup image (`mksysb`) during BOS install. By default, `mksysb restore` is performed during BOS install.
`-c`	Defines client objects from the `client.defs` file. The `/export/nim/client.defs` file must exist and have valid client definition information. The `client.defs` file is created during `nim_master_setup` command execution. If the file is not present, a sample `client.defs` file may be copied from `/usr/samples/nim/client.defs` and edited by the user.
`-r`	Reboots client objects after initiating BOS install operation. By default, clients are not rebooted. Resources are assigned for install and clients may be rebooted when desired.
`-v`	Enables verbose debug output during command execution.

Security

Access Control: You must have root authority to run the `nim_clients_setup` command.

Location

`/usr/sbin/nim_clients_setup`

Examples

1. To define client objects from `/export/nim/client.defs` file, initialize the newly defined clients for BOS install using resources from the `basic_res_grp` resource group, and reboot the clients to begin install, type:

   ```
   nim_clients_setup -c -r
   ```

2. To initialize clients client1 and client2 for BOS install, using the backup file `/export/resource/NIM/530mach.sysb` as the restore image, type:

   ```
   nim_clients_setup -m /export/resource/NIM/530mach.sysb \ client1 client2
   ```

3. To initialize all clients in the NIM environment for native (`rte`) BOS install using resources from the `basic_res_grp` resource group, type:

   ```
   nim_clients_setup -n
   ```

Files

Item	Description
`/etc/niminfo`	Contains variables used by NIM.

nim_master_recover Command

Purpose

Restores a backup of the Network Installation Management (NIM) database to a different machine and updates the database to reflect this change.

Syntax

`nim_master_recover` [`-f` *mstr_fileset_dir*]

[`-n` *nimdef_file*]

[`-r` *nimdb_file*]

[`-i` *mstr_interface*]

[`-D`] [`-R`] [`-S`] [`-p`] [`-s`] [`-u`] [`-v`]

[**`-N`** *mstr_net_info* [**`-t`** *net_def*]]

Description

The `nim_master_recover` command can restore and update the NIM database from a backup tar file. To backup the NIM database on the old master, run the `smit nim_backup_db` command. This creates a tar file named `/usr/objrepos/nimdb.backup` by default. Once the `nimdb.backup` is copied to the new master, pass the `-r` flag with the full path to the file. If the path to the tar file is `/usr/objrepos/nimdb.backup`, then pass `-r /usr/objrepos/nimdb.backup` to the `nim_master_recover` script.

The script updates the master definition in the NIM database based on the master's primary network interface. The `-i` flag specifies the primary interface to use for the master. To use `en0`, pass `-i en0` to the `nim_master_recover` script.

Note: A restored NIM database may be incorrect if you restore from a database that has network definitions containing static routes. The `nim_master_recover` command removes all the interfaces in the old master definition before adding the primary interface for the new master. Check that the routing information is correct after running the `nim_master_recover` command, by running `lsnim -lc networks`. If all the NIM network definitions in the restored database contain dynamic routes, then you should not run into this situation.

Along with restoring and updating the NIM database, the script performs several other optional functions. One is to install the `bos.sysmgt.nim.master` fileset if the `-f` flag is passed with the location of the `bos.sysmgt` package. For instance, if the `bos.sysmgt` package is located in the `/export/latest/installp/ppc` directory, then you would pass `-f /export/latest/installp/ppc` to the `nim_master_recover` script.

The script always resets each client. If the `-u` flag is passed, the script attempts to unexport NIM resources that the database states are allocated to clients. Each client stores the hostname of its NIM master in its `/etc/niminfo` file. To update the `niminfo` file on each client, pass the `-s` flag.

Note: Any NIM client that is not running, does not have a network connection, does not allow the new master `rhost` permissions, or does not have at least the `bos.sysmgt.nim.client 5.1.0.10` package, will not have its `niminfo` updated. The `nim_master_recover` script will report any clients which fail to have their `niminfo` files updated.

New clients can be added to the environment by specifying a `nimdef` file with the `-n` flag. Consult the AIX Installation Guide for more information on `nimdef` files.

Finally, the script will check to see if the resources in the NIM database exist. The script deletes resources that don't exist. For example if the new master is unable to communicate with a NIM server, then the resources defined on that server will be removed from the NIM database. Passing the `-R` flag prevents the script from checking resources.

Note: Resources that were defined on the master where the database was backed up, will not be available once the database is restored unless the resources were copied to the new master before running `nim_master_recover`.

All output will be logged to `/var/adm/ras/nim.recover`. Once the script is complete you should verify that no errors were logged.

The **nim_master_recover** command behaves differently when it is called with the **-N** flag. This allows the master to have its hostname, IP address, and NIM network changed in its if1 attribute. Optionally, a new NIM network may be created if the **-t** flag is provided with the **-N** flag. The command should be run with these flags before the master's network name or address is actually changed so that the NIM environment will work properly once the change actually takes place. When the master's NIM attributes are changed, the command will attempt to update **/.rhosts** and **/etc/niminfo** of each standalone client defined in the environment. Any clients for which this attempt fails must have its NIM master information updated manually. Also, after a standalone client has had its NIM master's network name changed, it will not be able to execute any NIM operations until the master is up and running with its new network name.

Flags

Item	Description
`-D`	Deletes all client definitions from the restored database.
`-f` *directory*	Directory containing the `bos.sysmgt.nim.master` fileset to install.
`-i` *interface*	Primary network interface of the machine where you are running the command.
`-n` *nimdef*	Optional *nimdef* file that will be used to define new machines.

Item	Description
-N *mstr_net_info*	Changes the master's if1 attribute and attempts to update each standalone client defined in the environment with the master's new network information. The *mstr_net_info* variable consists of the following: "`nim_net_name [hostname] [cable_type]`"; where *hostname* and *cable_type* are optional.
`-p`	Print the machine states before the script resets the machines.
`-r` *nimdb.backup*	The NIM database backup tar file that will be restored.
`-R`	Do not check the resources to see if each one exists. The default behavior is for the script to check each resource and if it does not exist, remove its definition from the database.
`-S`	Do not check the SPOT resources. The default behavior is for the script to check every SPOT to ensure it is ready to support an install. For example, the check ensures the boot images are created.
`-s`	Attempt to update the `niminfo` file on each client. Any NIM client that is not running, does not have a network connection, does not allow the new master rhost permissions, or does not have at least the `bos.sysmgt.nim.client 5.1.0.10` package installed, will not have its `niminfo` updated.
-t *net_def*	Creates a new NIM network if the master's IP address changes and there is no existing NIM network that could contain the master. This flag is only valid when the **-N** flag is also specified. The *net_def* variable consists of the following: "`nim_net_name net_type net_addr net_snm default_route`"; where *net_type* can be `ent`, `tok`, `atm`, or `fddi`.
`-u`	Unexport all resources that are listed as allocated in the restored database. The default behavior is for the script to delete the allocation from the NIM database without attempting to deallocate the resource.
`-v`	Enables verbose debug output during command execution.

Location

`/usr/sbin/nim_master_recover`

Exit Status

Returns zero (0) upon success.

Security

Access Control: You must have root authority to run the `nim_master_recover` command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To recover the NIM master using the `/export/nim/nimdb.backup` file and the primary interface en0, type:

```
nim_master_recover -r /usr/objrepos/nimdb.backup -i en0
```

2. To install the `bos.sysmgt.nim.master` fileset from `/export/lpp_source/installp/ppc` before recovering the NIM master, type:

```
nim_master_recover -f /export/lpp_source/installp/ppc \
          -r /usr/objrepos/nimdb.backup -i en0
```

3. To recover the NIM master without checking if each resource exists and without checking the SPOTs to rebuild boot images, type:

```
nim_master_recover -R -S -r /usr/objrepos/nimdb.backup -i en0
```

4. To recover the NIM master while unexporting any resources that are allocated and printing the state of the clients before each one is reset, type:

```
nim_master_recover -u -p -r /usr/objrepos/nimdb.backup -i en0
```

5. To recover the NIM master and update the `/etc/niminfo` file on each client, type:

```
nim_master_recover -s -r /usr/objrepos/nimdb.backup -i en0
```

6. To recover the NIM master, delete each client from the database, and define new clients from the `nimdef` file `/export/nim/nimdef`, type:

```
nim_master_recover -D -n /export/nim/nimdef -r /usr/objrepos/nimdb.backup -i en0
```

7. To change the master's hostname to `newhost.domain.com` and move it to a different existing NIM network, called `net2`, but preserve the value of the current *cable_type* attribute, type:

```
nim_master_recover -N "net2 newhost.domain.com"
```

8. To change the master's hostname to `newhost.domain.com`, change its *cable_type* to `bnc`, and move it to a new NIM ethernet network called `new_nim_net` whose address is `192.168.1.0`, subnet mask is `255.255.255.0`, and default gateway is `192.168.1.1`, type:

```
nim_master_recover -N "new_nim_net newhost.domain.com bnc" \
    -t "new_nim_net ent 192.168.1.0 255.255.255.0 192.168.1.1"
```

Files

Item	Description
`/etc/niminfo`	Contains variables used by NIM.
`/var/adm/ras/nim.recover`	Contains log information from command execution.

nim_master_setup Command

Purpose

Initializes the Network Installation Management (NIM) master fileset, configures the NIM master, and creates the required resources for installation.

Syntax

`nim_master_setup` [-a [mk_resource={yes|no}] [file_system=*fs_name*] [volume_group=*vg_name*] [disk=*disk_name*] [device=*device*]] [-B] [-F] [-L] [-v]

Description

The `nim_master_setup` command initializes the NIM master fileset and configures the NIM environment. Once initialized, the `nim_master_setup` command configures the NIM environment by performing the following tasks:

- Determines which volume group and file system will contain the NIM resources.
- If necessary, creates the volume group and file system.
- Creates a NIM `mksysb` of the master.
  - Backup image.
- Creates a NIM `lpp_source` resource.
  - Source for product images.
- Creates a NIM spot resource.
  - Shared Product Object Tree (SPOT) - equivalent to `/usr` file system.
- Creates a NIM `bosinst_data` resource.
  - `config` file used during BOS installation.
- Creates a NIM `resolv_conf` resource.
  - Name-server configuration file.
- Defines a default resource group for use during install. The default resource group will contain all NIM resources defined during command execution.
- Copies a sample `client.defs` configuration file into the defined NIM file system.
  - Sample file which may be edited for adding clients in the NIM environment.

Flags

Item	Description
`-a`	Assigns the following `attribute=`*value* pairs: **mk_resource={yes|no}** Specifies if NIM resources should be created. If set to no, NIM resources will not be created during command execution. By default, the value is yes. **file_system=*fs_name*** Specifies the absolute path location for creating NIM resources. If *fs_name* does not exist, a logical volume will be created in the volume group defined from *vg_name*. By default, *fs_name* is `/export/nim`. **volume_group=*vg_name*** Specifies the volume group name used for creating new logical volumes. If *vg_name* does not exist, a volume group will be created using the physical volume (disk) defined from *disk_name*. By default, *vg_name* is `rootvg`. **disk=*disk_name*** Specifies the physical volume used when creating the *vg_name* volume group. If *disk_name* is not specified, the next available (empty) physical volume will be used. **device=*device*** Specifies the absolute path location for install images used during NIM master fileset installation and resource creation. By default, `device` is `/dev/cd0`.
`-B`	Disables the creation of the backup image.
`-F`	Disables the creation of the file system.
`-L`	Disables the creation of the `lpp_source` resource.
`-v`	Enables verbose debug output during command execution.

Location

`/usr/sbin/nim_master_setup`

Exit Status

Returns zero (0) upon success.

Security

Access Control: You must have root authority to run the `nim_master_setup` command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To install the NIM master fileset and initialize the NIM environment using install media located in device `/dev/cd1`, type:

   ```
   nim_master_setup -a device=/dev/cd1
   ```

2. To initialize the NIM environment without creating NIM install resources, type:

   ```
   nim_master_setup -a mk_resource=no
   ```

3. To initialize the NIM environment, create NIM install resources without creating a backup image, using install media located under mount point `/cdrom`, type:

   ```
   nim_master_setup -a device=/cdrom -B
   ```

4. To define NIM resources in an existing NIM environment, using install media located in device `/dev/cd0`, and create a new file system named `/export/resources/NIM` under volume group `nimvg`, type:

   ```
   nim_master_setup -a volume_group=nimvg  \
               -a file_system=/export/resources/NIM
   ```

Note: If the file system `/export/resources/NIM` does not currently exist, then it will be created under the volume group `nimvg`. If the `nimvg` volume group does not exist, it will be created using the next empty physical volume (disk) since the disk attribute was not specified.

Files

Item	Description
`/etc/niminfo`	Contains variables used by NIM.
`/var/adm/ras/nim.setup`	Contains log information from command execution.

nim_move_up Command

Purpose

Facilitates the enablement of new hardware in AIX environments.

Syntax

nim_move_up {[-S] | [-K [-h *control_host*]] | [-r [-R] [-u]]} | { [-c *NIM_client*] [-i *target_ip* [*-ending_ip*]] [-s *subnet_mask*] [-g *gateway*] [-h *control_host*] [-m *managed_sys*] [-V *vio_server* [-e] [-D]] [-I *img_src*] [-l *resource_dir*] [-t *seconds*] [-p *loops*] [-j *nimadm_vg*] [-L *lpp_source*] [-U *spot*] [-B *bosinst_data*] [-E *exclude_files*] [-C *script_resource*] [-b *installp_bundle*] [-f *fix_bundle*] {{[-n] [-d]} | -O} [-q] [-T] [-M *manual_configuration_filenames*]}

Description

The nim_move_up command enables users of existing AIX environments to take advantage of the capabilities available on new POWER servers. The command provides an interface that migrates an existing AIX system onto an LPAR residing on a POWER server. The level of AIX on the original machine is raised to a level that supports operation on newer hardware. The original system's hardware resources are closely replicated on the newer hardware. By the end of the migration, the same system is fully running on the new LPAR.

In addition, nim_move_up can use the Virtual I/O capabilities of POWER servers by optionally migrating a client onto virtualized hardware, such as virtual disks and virtual Ethernet.

The nim_move_up command relies on the functionality of NIM and the NIM master's capability of remotely managing and installing NIM clients on the network. The nim_move_up command attempts to use the NIM master and the nimadm command to complete the following actions on an existing NIM client:

1. Create a system backup of the client
2. Migrate the backup's level of AIX
3. Install the backup onto an LPAR that resides on the new POWER server, which is be represented in the NIM environment as a new standalone client.

Before the new hardware is installed, the NIM master (on which the **nim_move_up** command is run) and the NIM clients on the existing hardware must be configured. The clients are the starting point of the migration and eventually turn into the new LPAR.

After a successful migration, the following conditions are true:

- The NIM master remains the same.
- The LPAR on the new POWER server correspond to the original NIM clients and are controlled by the NIM master.
- An HMC controls the LPAR on the new POWER servers by communicating with the NIM master through SSH.
- The original NIM clients remain unaffected and still in control of the NIM master.

The entire migration takes place without any downtime required on the part of the original client. The process can be completed in phases executed sequentially, which allows more control over the process, or can be executed all at once, so that no user interaction is required. The command is delivered as part of the bos.sysmgt.nim.master fileset and requires a functional NIM environment in order to run.

Required Flags

Item	Description
-c *NIM_client*	Specifies either a NIM standalone client (standalone object type) or a NIM machine group (mac_group object type). The client indicated must be reachable using the network from the NIM master and must allow the NIM master to run commands on them. If a NIM machine group is specified in this argument, it must reside in the same NIM network. The client is the target machine that will be migrated onto the new LPAR on a POWER server.
-g *gateway*	Specifies the IP address of the default gateway that the clients will be configured with after the migration to the POWER server.
-h *control_host*	Specifies the host name or IP address of the HMC that is used for hardware control of the POWER server.
-i *target_ip*[-*ending_ip*]	Specifies the IP address that the new migrated client will be configured with after it is installed on the POWER server. If a NIM machine group is supplied to the -c option, a range of IP addresses must be supplied here and there must be enough addresses in the range to enumerate the amount of clients that are to be migrated.
-I *img_src*	Specifies the path to the source of the installation images used to create the NIM resources required for migration and installation. This path can be a device (such as dev/cd0 if using AIX product media) or a path to a location on the file system containing the installation images.
-l *resource_dir*	Specifies the path to a location on the file system that will contain any new NIM resources created through the nim_move_up command. The location must have enough space to accommodate an LPP_Source and a spot unless existing resources were provided through the -L and -U options.
-m *managed_sys*	Specifies the name of the managed system corresponding to the POWER server as tracked by the HMC.
-s *subnet_mask*	Specifies the subnet mask that the clients will be configured with after the migration to the POWER server.

Execution and Control Flags

Item	Description
-d	Executes nim_move_up in the background and returns control of the terminal to the caller. The progress of nim_move_up can be tracked through the -S flag.

Item	Description
-K	Configures SSH keys on the specified HMC. This allows the unattended remote execution of commands from the NIM master without password prompts. This flag cannot be used with any other options except the -h option.
-n	Runs only the next phase of the nim_move_up migration process. The nim_move_up command exits when the phase completes or fails. If this flag is not provided, all the subsequent phases are run and nim_move_up exits when they have all run or one of them has failed.
-O	Saves only supplied values. Save values provided through other options and then exits without executing any phases. This flag cannot be used with any other of the Execution and Control Flags.
-q	Specifies quiet mode. No output is displayed to the terminal (but is instead kept in the logs). This flag has no effect if nim_move_up runs with the -d flag.
-r	Unconfigures nim_move_up. This resets all saved data, including saved options, phase-specific data, and current phase information. This operation must be run if the migration process is to be started over for the migration of a new client or set of clients.
-R	Removes all NIM resources created by nim_move_up in addition to unconfiguring the environment. This flag can only be used with the -r option.
-S	Displays the status of the current phase or the next phase to be run. All saved values are displayed as well. The nim_move_up command exits immediately after displaying the information. This flag cannot be used with any other options.

Optional Flags

Item	Description
-b *installp_bundle*	Specifies an existing installp_bundle NIM resource whose software are installed on each of the newly migrated LPAR in phase 10 (post-installation customization) if the option is provided.
-B *bosinst_data*	Specifies an existing bosinst_data NIM resource used by nim_move_up to install the new clients onto the new LPAR. If this option is not provided, nim_move_up generates a bosinst_data resource with default unattended installation values.
-C *script_resource*	Specifies an existing script NIM resource that, if provided, nim_move_up will execute in phase 10 (post-installation customization) on all of the new migrated LPAR.

Item	Description
`-D`	Forces the use of physical storage controllers instead of virtual SCSI adapters in creating the new LPAR on the POWER server when a Virtual I/O server LPAR is specified. This flag is only valid when used with the `-V` option.
`-e`	Forces the use of physical network adapters instead of shared Ethernet adapters in creating the new LPAR on the POWER server when a Virtual I/O server LPAR is specified. This flag is only valid when used with the `-V` option.
`-E` *exclude_files*	Specifies an existing `exclude_files` NIM resource that `nim_move_up` uses to create a `mksysb` of the original clients. If this option is not provided, `nim_move_up` generates an `exclude_files` resource that excludes the contents of `/tmp` from the backup.
`-f` *fix_bundle*	Specifies an existing `fix_bundle` NIM resource whose APARs are installed on each of the newly migrated LPARin phase 10 (post-installation customization) if the option is provided.
`-j` *nimadm_vg*	Specifies the volume group to be used by the underlying `nimadm` call for data caching. If this option is not provided, the default value is `rootvg`.
`-L` *lpp_source*	Specifies an existing LPP_Source NIM resource to whose AIX level the target clients will be migrated to. If this option is not provided, `nim_move_up` attempts to create a new LPP_Source from the installation image source provided through the `-I` option.
`-M` *manual_configuration_filenames*	Specifies **phase4** to use these manual configuration files to the associated back-level AIX machines. This flag is effective only in **phase4** of the **nim_move_up** command. For more information about this flag, see the Advanced usage section.
`-p` *loops*	Specifies the number of times to execute system analysis tools on the target NIM clients in analyzing resource utilization. The final resource usage data will be the average of the values obtained from each loop. This data will be taken into account when determining the equivalent POWER server resources from which the migrated LPAR will be derived. If this option is not provided, the default is 1 loop.
`-t` *seconds*	Specifies the number of seconds each loop runs for. If this option is not provided, the default is 10 seconds.
`-T`	Transports user-defined volume groups from the original clients to the new migrated LPAR.
`-u`	Enables `nim_move_up` to completely "roll back" entire `nim_move_up` migration. Must be used with the `-r` flag.

Item	Description
-U *spot*	Specifies an existing spot NIM resource that will be used in the migration and installation of the clients. If this option is not provided, a new spot is created from the *lpp_source* NIM resource provided by the -L and -I options.
-V *vio_server*	Specifies the LPAR name of a Virtual I/O server that resides on the POWER server denoted by the -m flag.

Exit Status

Item	Description
0	Successful completion.
nonzero	An error occurred.

Security

Only the root user can run this command.

Examples

1. To run the first phase and configure all the required options (nim_move_up must not be already configured and running), type:

```
nim_move_up -c client1 -i 192.168.1.100 -s 255.255.255.0 -g 192.168.1.1 -h hmc1.mydomain.com
-m \
my-p5 -l /big/dir -I /dev/cd0 -n
```

2. To display the status of the nim_move_up command's environment, including all saved configuration input and which phase is to be executed next, type:

```
nim_move_up -S
```

3. To change the saved host name to a new name and run the next phase while suppressing output, type:

```
nim_move_up -h hmc2.mydomain.com -n -q
```

4. To run all remaining phases in the background, save your agreement to accept all licenses, and have the prompt returned after the phases begin running, type:

```
nim_move_up -Y -d
```

5. To unconfigure nim_move_up, discard all saved input, and reset the command to run phase 1, type:

```
nim_move_up -r
```

   All NIM resources previously created by nim_move_up remain unaffected in the NIM environment and will be used by nim_move_up as necessary to migrate another client.

Restrictions

The following NIM master requirements must be met before running the nim_move_up application:

- Running AIX 5L Version 5.3 with the 5300-03 Recommended Maintenance package, or later.
- Perl 5.6 or later.
- OpenSSH (from the Linux Toolbox CD)
- At least one standalone NIM client running AIX 4.3.3 update or later in the environment

- Product media version AIX 5L Version 5.2 with the 5200-04 Recommended Maintenance package or later, or product media version AIX 5.3 or later (the equivalent LPP_Source and spot NIM resources can also be used).

In addition, the following prerequisites must be available:

- A POWER server with sufficient hardware resources to support the target clients' equivalent POWER server configuration.
- An installed and configured Virtual I/O server is, if virtual resources will be used to migrate the clients.
- An HMC controlling the POWER server, along with sufficient privileges to power-on, power-off, and create LPAR.

The `nim_move_up` command will fail to execute properly if all of the preceding requirements are not met or if the command is executed by a non-root user.

Implementation Specifics

The `nim_move_up` command takes a phased approach to migrating an existing client onto a new LPAR. The following phases make up the process:

1. **Create NIM resources.** The NIM resources required to perform the migration steps are created if they do not already exist.
2. **Assess premigration software.** An assessment of which software is installed and which software cannot be migrated is performed on each target client. Any software missing from the LPP_Source is added from the source of the installation images (such as product media) that is provided to `nim_move_up`.
3. **Collect client hardware and usage data.** Data about each target client's hardware resources are gathered. Also, an attempt to assess the average use of those resources over a given amount of time is made.
4. **Collect POWER server resource availability data and translate client resource data.** The managed system that is provided is searched for available hardware resources. The data gathered in the previous phase is used to derive an equivalent LPAR configuration that uses the managed system's available resources. If a Virtual I/O server LPAR was provided to work with, the derived client LPAR is created with virtual I/O resources instead of physical I/O resources. The appropriate adapters and configuration are created on the Virtual I/O server as needed.
5. **Create system backups of target clients.** After NIM performs a `mksysb` of each target client, the corresponding `mksysb` NIM resources are created.
6. **Migrate each system backup.** Using the NIM resources designated by `nim_move_up`, each `mksysb` resource is migrated to the new level of AIX by the `nimadm` command. The original `mksysb` NIM resources are preserved and new mksysb NIM resources are created for the new migrated `mksysb` resources.
7. **Allocate NIM resources to new LPAR.** NIM standalone client objects are created for each new derived LPAR created in phase 4 using the network information provided to `nim_move_up`. Appropriate NIM resources are allocated and a `bos_inst` pull operation is run on each NIM client (NIM does not attempt to boot the client).
8. **Initiate installation on LPAR.** Each LPAR is rebooted using the control host (HMC) and the installation is initiated. The phase's execution stops after the installation has begun (that is, the progress of the installation is not monitored).
9. **Assess post-migration software.** After each installation has completed, the overall success of the migration is assessed, and a report of software problems encountered during migration is generated. If any filesets failed to migrate, the errors reported for that fileset must be corrected manually.
10. **Customize post-installation.** If an alternate LPP_Source, fileset list, or customization script was provided, a customized NIM operation is performed on each client with the values provided. This allows for the optional installation of additional software applications or for any additional customization.

In order to successfully migrate a NIM client onto a new LPAR, each of these phases (with the exception of phase 10, which is optional) must be executed completely successfully. If all phases completed successfully, a new NIM client object will be present in the NIM environment that represents the migrated LPAR, which will be running the level of AIX supplied through the `nim_move_up` source of installation resources.

After all prerequisites needed to run `nim_move_up` have been satisfied, the `nim_move_up` command runs in two phases: configuration and phase execution.

Configuration

Before the `nim_move_up` command can begin its phases, input must be provided to the application. The required input includes a list of the NIM clients to be migrated, TCP/IP configuration information of the new migrated LPAR, and the POWER server name. For a complete list of required `nim_move_up` configuration options, refer to the Required Flags (they also are denoted by a * (asterisk) in the `nim_move_up_config` SMIT menu). Optional input, such as whether a Virtual I/O server is specified, also affects the behavior of `nim_move_up` and the end result of the migration process (if a Virtual I/O server is specified, virtual I/O resources are used to create the migrated LPAR).

To populate the required and optional input through the SMIT interface, enter one of the following commands:

```
smitty nim_move_up_config
```

or

```
smitty nim_move_up
```

and select the `Configure nim_move_up Input Values` option.

At the menu, fill in the options with values that reflect the requirements of your environment. For further information about the `nim_move_up` command's SMIT interface, see the SMIT usage section below.

After the `nim_move_up` command's environment has been configured with the needed input, those values are remembered through subsequent runs of the `nim_move_up` command until the `nim_move_up` command environment is unconfigured. The values can be changed at any time through the SMIT menu interface or by providing the new values through command line flags. The command line interface can also be used to configure the `nim_move_up` command environment.

Note:

If you use the command line interface, the `nim_move_up` command, by default, also attempts to execute phases whenever configuration values are provided to it. To prevent phases from being executed when calling the command directly, use the `-0` flag.

Phase Execution

After all input is supplied, phase execution begins at phase 1 and continues sequentially. If a phase encounters an error, `nim_move_up` attempts to execute the failed phase the next time it runs. Optionally, you can specify that `nim_move_up` start only the next phase or attempt all remaining phases.

To start `nim_move_up` phases through the SMIT interface, type one of the following commands:

```
smitty nim_move_up_exec
```

or

```
smitty nim_move_up
```

and select the `Execute the nim_move_up Phases` option. Answer the `Execute All Remaining Phases?` option and press Enter. The phases begin executing.

To specify that nim_move_up execute only the next phase using the command line, type the following command:

```
nim_move_up -n
```

To specify that nim_move_up execute all remaining phases, type the following command:

```
nim_move_up
```

In addition to executing phases, this command can also modify saved configuration options if the appropriate flag is supplied.

SMIT Usage

The nim_move_up SMIT menus can be accessed using the nim_move_up fastpath. To invoke the root menu of nim_move_up, type the following command:

```
smitty nim_move_up
```

The following SMIT screens are accessible through the root menu:

Display the Current Status of nim_move_up
Equivalent to running nim_move_up with the -S flag. The next phase to be executed and a listing of all the saved options are displayed.

Configure nim_move_up Input Values
Through this screen, all required and optional input to nim_move_up can be configured. All values entered into the fields are saved and are remembered through subsequent runs of nim_move_up and through subsequent uses of this SMIT screen. This screen can be used at any time to modify saved values after phases have been run.

Execute nim_move_up Phases
Provides a simple interface to execute nim_move_up phases. The phases can be executed one at a time or all at once, depending on how the questions in this phase are answered.

Configure SSH Keys on Target HMC
Provides a simple interface for setting up SSH keys on the remote control host (HMC). This does the equivalent work of passing the -K flag on the command line. Configuring SSH keys on the remote control host enables the unattended remote execution of commands from the NIM master, which is necessary for completing all the phases (some of which remotely execute commands on this system).

Unconfigure nim_move_up
Provides an interface to unconfigure the nim_move_up command's environment. This removes all state information, including which phase to execute next, saved data files generated as a result of the execution of some phases, and all saved input values. Optionally, all NIM resources created through nim_move_up can be removed as well. This screen does the equivalent work of the -r command line option.

Advanced Usage: Understanding the mig2p5 Framework

The mig2p5 framework consists of the /var/mig2p5 directory and serves as a means for nim_move_up to remember its state between subsequent invocations. Its existence and its use by nim_move_up is completely transparent to the user: the directory is created by nim_move_up and its values are initialized if it does not exist. It is removed when nim_move_up is unconfigured. The contents of this directory are easily readable and can be very helpful in troubleshooting problems with nim_move_up; the directory contains all of the logs generated in the phases and contains editable files that affect the behavior of nim_move_up in ways that are not allowed by the command line (such as forcing nim_move_up to run a certain phase out of order).

The following list describes the purpose and contents of each file in the /var/mig2p5 directory:

config_db
Contains all of the saved configuration options passed to `nim_move_up` through the command line arguments or the `nim_move_up_config` SMIT menu. Each line in the file takes the following form:

```
option_name:value
```

current_phase
Contains the number of the phase that will be executed at the next invocation of `nim_move_up`. Before running this phase, `nim_move_up` ensures that all previous phases have run successfully. This information is also maintained elsewhere with the `mig2p5` framework.

global_log
Contains the output of all phases that have been run since the last time the `mig2p5` framework was initialized.

client_data/
Contains files that are generated by `nim_move_up` during phases 3 and 4, in which each of the original clients' system resources and utilization are monitored and quantified into configuration files. The available resources in the POWER server are also quantified into corresponding text files. All the data in these files will be taken into account when determining the hardware profile of the newly derived LPAR on the POWER server. These files are intended to be machine-readable data files for the `nim_move_up` command's internal use. Do not manually modify or create them.

phase#/
Contains data specific to the corresponding phase denoted by the number in its name (#). Every phase has a directory (for example, `phase1/` , `phase2/` , and so on).

phase#/log
Contains all output displayed during a phase's run. If a phase runs multiple times (such as after an error has been corrected), all new output is appended to any text already existing in the file. This log is helpful in investigating failures related to this phase after they have occurred. The `global_log` file is composed of all the phases' log files, and all output in that file is arranged in the order that it was originally displayed.

phase#/status
Indicates whether this phase succeeded or failed when it was last run. This file is used by `nim_move_up` to determine whether a subsequent phase can be run. A phase can run only if all of the previous phases' `status` files contain the string `success`. The `status` file contains the `failure` string if the phase encountered an error that caused it to fail the last time it was run.

pid
Contains the `nim_move_up` process ID number when `nim_move_up` is running in the background. This file and is cleaned up when the process finishes. As long as this file exists and contains a process ID, `nim_move_up` cannot run phases because concurrent runs of `nim_move_up` are not supported.

With the exception of the log files and the contents of the `client_data/` directory, the files in `/var/mig2p5` that comprise the `mig2p5` framework can be read and modified so that `nim_move_up` performs tasks that it would not do through its command line and SMIT interfaces. Users are encouraged to manipulate the `mig2p5` environment to make `nim_move_up` meet any specific need and to aid in the troubleshooting of any problems that might arise during the migration process.

Note: Customizing the `mig2p5` framework is considered advanced usage and can yield unsatisfactory results if done incorrectly. The `mig2p5` environment should only be directly modified by users who understand the changes being performed and their effect on the behavior of the `nim_move_up` application.

What is the manual configuration file and why is it needed?

During **phase4** of the **nim_move_up** command, the tool calculates various resource requirements based on the back-level AIX machine and appropriately creates an LPAR on a target POWER server. When you meet any of the following situations, you can specify what modifications you need in the manual configuration file in a predefined format and run the **nim_move_up** command:

- There is a need for more memory than that determined by the **nim_move_up** command.

- There is a virtual SCSI adapter (vhost#) created on a Virtual I/O server that you want to use for a Volume Group.
- You want to use a different Virtual Local Area Network (VLAN) ID than the one generated by the **nim_move_up** tool.

After the successful completion of the **nim_move_up** command, the manual configuration is applied on the target LPAR.

How do I write a manual configuration file?

Note: You must create the manual configuration file before initiating the **nim_move_up** command. You can create these files for each of the clients to be migrated and specify these files as arguments to the **-M** flag to enable the **nim_move_up** command to use the manual configuration. The file name must be of the form *path*/manual_cfginfo_*client_host_name*. The *path* value is the directory where the manual configuration file is located, and the *client_host_name* value is the host name of the client machine to be migrated.

For each client that is migrated to a POWER platform, the **nim_move_up** command does all of the hardware configuration-related calculations by default. This file enables you to alter or tune the configuration of the target machine as you choose.

You can change the amount of memory, the size of the volume groups and the Virtual I/O server resources to be used. For example, you can change the VSCSI server adapter to be used for the volume groups created for the target LPAR. You can also change the VLAN IDs to be used for the Ethernet adapters created for the target LPAR.

The following is a sample of the manual configuration file:

```
# manual_cfgfile_dennis file
# MEMORY = min_MB desired_MB max_MB
MEMORY = 256 512 1024
# VIO_VG_INFO = vgname_src, size_in_MB, vhost_to_use
#    Where vgname_src is the VG name in source machine, and
#    vhost_to_use is the virtual adapter to be used for
#    the VG specified in the VIO Server.
VIO_VG_INFO = rootvg,15344,vhost4
# VIO_VLAN_INFO = vlan_id, lpar_name, slot_number
VIO_VLAN_INFO = 1,VIO-server,2
```

The file can have any blank lines. You can add comments to the file with a # at the beginning of the line.

All of the *min_MB*, *desired_MB*, and *max_MB* values must be in megabytes (MB). There is no restriction on the number of spaces between these numbers.

min_MB
The minimum memory required for AIX to run.

desired_MB
The amount of memory that you want the logical partition to have when activated.

max_MB
The maximum memory when dynamic logical-partitioning operations are performed on the partition.

The values of the VIO_VG_INFO field must be comma separated. The *vgname_src* value is the Volume Group in the source machine for which the manual data must be given. The *size_in_MB* value is the size of the Volume Group on the target machine and the *vhost_to_use* value is the vhost* (virtual SCSI server adapter) to be used for this Volume Group on the target POWER server.

Similarly, the values of the VIO_VLAN_INFO field must be comma separated. The *vlan_id* value is used instead of the one used by the **nim_move_up** command for the target LPAR's Ethernet adapter. The *lpar_name* value is the LPAR name of the Virtual I/O server having the shared Ethernet adapter (SEA), and the *slot_number* value is the slot number of this SEA on the Virtual I/O server.

It is not necessary to provide all of these values. The **nim_move_up** command receives the specified values from the manual file and generates the rest based on the client configuration.

Files

Item	Description
/usr/sbin/nim_move_up	Contains the nim_move_up command.

nim_update_all Command

Purpose

Updates NIM resources and customizes NIM clients.

Syntax

nim_update_all [-d *device*] [-l *lpp_source resource*] [-s *spot resource*] [-B] [-u] [-v] *client object(s)*

Description

The nim_update_all command updates the install resources and clients in the NIM environment. Flags may be used for specifying which NIM resources need updating and also to disable the updating of NIM clients. The nim_update_all command updates the NIM environment by performing the following tasks:

- Exports the environment variable NIM_LICENSE_ACCEPT=yes.
  - Used for accepting software license agreement during update install.
- Adds variable entry NSORDER=local,bind in /etc/environment.
  - Necessary for name resolution when hosts only exist in /etc/host.
- Obtains the update level information from the media.
  - The default media location is /dev/cd0.
  - The media location may be modified by using the -d flag.
- Updates the lpp_source, spot, and mksysb resources.
  - The lpp_source resource name may be specified by using the -l flag.
  - The spot resource name may be specified by using the -s flag.
  - The mksysb resource name is obtained from the mksysb resource contained in the basic_res_grp resource group. Specify the -B flag to disable updating the mksysb resource.
- Performs an update_all operation on NIM clients.
  - If client objects are given, specified clients are updated.
  - If client objects are omitted, all existing NIM clients are updated.
  - If -u flag is used, no clients are updated.

Flags

Item	Description
-d *device*	Specifies the absolute path location for update images used during command execution. By default, *device* is /dev/cd0.
-l *lpp_source resource*	Specifies the object name for the *lpp_source resource* to update. By default, the resource name is obtained from basic_res_grp.
-s *spot resource*	Specifies the object name for the *spot resource* to update. By default, the resource name is obtained from basic_res_grp.

Item	Description
-B	Disables the updating of the backup image contained in basic_res_grp.
-u	Disables the updating of client objects.
-v	Enables verbose debug output during command execution. Security

Location

/usr/sbin/nim_update_all

Exit Status

Returns zero (0) upon success.

Security

Access Control: You must have root authority to run the nim_update_all command.

Examples

1. To update install resources 520lpp_res (lpp_source), 520spot_res (spot), and master_sysb (mksysb) contained in the basic_res_grp resource group, using update images located in device /dev/cd2, and update all clients in the NIM environment, type:

```
nim_update_all -d /dev/cd2
```

2. To update install resources lpp1 (lpp_source), spot1 (spot), and disable updating the mksysb image, using update images located in device /dev/cd0, and update the client object machine1 in the NIM environment, type:

```
nim_update_all -l lpp1 -s spot1 \
          -B machine1
```

3. To update install resources 520lpp_res (lpp_source), 520spot_res (spot), and disable updating the mksysb image contained in the basic_res_grp resource group, using update images located in device /dev/cd0, and disable updating clients in the NIM environment, type:

```
nim_update_all -B -u
```

Files

Item	Description
/etc/niminfo	Contains variables used by NIM.
/var/adm/ras/nim.update	Contains log information from command execution.

nimadapters Command

Purpose

Defines Network Installation Management (NIM) secondary adapter definitions from a stanza file.

Syntax

nimadapters {-p | -d | -r } -f SecondaryAdapterFileName *adapter_def_name*

or

nimadapters {-p | -d | -r }-a client=*Client* [-a info=*AttributeList*] *adapter_def_name*

Description

The nimadapters command parses a secondary adapters stanza file to build the files required to add NIM secondary adapter definitions to the NIM environment as part of an *adapter_def* resource. The nimadapters command does not configure secondary adapters. The actual configuration takes place during a nim -o bos_inst or nim -o cust operation that references the *adapter_def* resource.

Note: Before using the nimadapters command, you must configure the NIM master. For more information, see **Configuring the NIM Master and Creating Basic Installation Resources** in *Installation and migration* guide.

Secondary Adapters File Rules

The format of the secondary adapters file must comply with the following rules:

- After the stanza header, follow attribute lines of the form: Attribute = *Value*
- If you define the value of an attribute multiple times within the same stanza, only the last definition is used.
- If you use an invalid attribute keyword, that attribute definition is ignored.
- Each line of the file can have only one header or attribute definition.
- More than one stanza can exist in a definition file for each machine host name.
- Each stanza for a machine host name represents a secondary adapter definition on that NIM client. No two secondary adapter definitions for the same machine host name can have the same location or interface_name. There should be only one definition per adapter or interface on a given NIM client.
- If the stanza header entry is the keyword default, this specifies to use that stanza for the purpose of defining default values.
- You can specify a default value for any secondary adapter attribute. However, the netaddr and secondary_hostname attribute must be unique. Also, the location and interface_name must be unique on a NIM client.
- If you do not specify an attribute for a secondary adapter but define a default value, the default value is used.
- You can specify and change default values at any location in the definition file. After a default value is set, it applies to all definitions following it.
- To turn off a default value for all following machine definitions, set the attribute value to nothing in a default stanza.
- To turn off a default value for a single machine definition, set the attribute value to nothing in the machine stanza.
- You can include comments in a client definition file. Comments begin with the # character.
- Tab characters and spaces are ignored when parsing the definition file for header and attribute keywords and values.

Note: During a nim -o bos_inst or nim -o cust operation, if NIM examines the configuration data on the client and determines that a secondary adapter is already configured with precisely the attributes requested in the *adapter_def* resource, this secondary adapter is not reconfigured.

Secondary Adapter File Keywords

The secondary adapter file uses the following keywords to specify machine attributes:

Required Attributes

machine_type = secondary | etherchannel | install
Specifying the machine_type attribute as secondary clearly distinguishes the nimadapters input from nimdef input. If a secondary adapters file is mistakenly passed to the nimdef command, the error can be easily detected. Stanzas with a machine_type of install will be ignored.

netaddr
Specifies the network address for the secondary adapter.

network_type = en | et | sn | ml | vi
Specifies the type of network interface, which can be one of en, et, sn, ml, or vi. This attribute replaces the deprecated network_type attribute.

subnet_mask
Specifies the subnet mask used by the secondary adapter.

Optional Attributes

adapter_attributes
Blank-separated list of physical adapter attributes and values (for example, "Attribute1=Value1 Attribute2=Value2"). To see the list of attributes that can be set for the requested physical adapter, run the command `lsattr -E -l` ***AdapterName***.

interface_attributes
Blank-separated list of interface attributes and values (for example, "Attribute1=Value1 Attribute2=Value2"). To see the list of attributes that can be set for the requested interface, run the command `lsattr -E -l` ***InterfaceName***. This attribute replaces the **attributes** attribute.

cable_type
Specifies the cable type (optional if network_type is en or et).

comments
Specifies a comment to include in the secondary adapter definition. Enclose the comment string in double quotes (").

interface_name
Specifies the name of the network interface for the secondary adapter (for example, en1, sn0, ml0). Do not specify both location and interface_name.

Note: The interface_name must be consistent with the interface_type.

location
Specifies the physical location of the adapter corresponding to this network interface. Do not specify both location and interface_name.

Note: Except for the multilink pseudo-device, use of the location is highly recommended. If the location is not specified and the user adds multiple adapters or adds an adapter at the same time that the operating system is reinstalled, the adapter and network interface names might be reassigned by the operating system in unexpected ways.

multiple_physloc
This attribute can be used with etherchannel or VIPA stanzas to specify the physical adapters to associate with the interface.

media_speed
Specifies the media speed (optional if network_type is en or et).

secondary_hostname
Host name to save in the `/etc/hosts` file with the netaddr attribute. This host name will not be set using the `hostname` command or `uname -S` command.

bos_preconfig
Specifies that the **tunchange** command is to change the value of tuning parameters. With the **bos_preconfig** attribute, you can change tunable parameters that have been set by the **/usr/lpp/bos.sysmgt/nim/methods/c_cfgadptrs** script with default values. The **bos_preconfig** attribute is used for the **nim -o bos_inst** command. For more information about the valid stanza and the respected stanza commands for tunable values, see the **tunchange** command.

The format for the **bos_preconfig** attribute is as follows:

```
bos_preconfig="tunchange -f nextboot -t Stanza [ -o tunable=value ... ]"
```

Requirement: You must restart the system in order for any new setting you made using the **bos_preconfig** attribute to take effect.

cust_preconfig
Specifies that the **vmo** command is to change the value of tuning parameters. With the **cust_preconfig** attribute, you can change tunable parameters that have been set by the **/usr/lpp/bos.sysmgt/nim/methods/c_cfgadptrs** script with default values. The **cust_preconfig** attribute is used for the **nim -o cust** command. For more information about valid tunable parameters, see the **vmo** command.

The format for the **cust_preconfig** attribute is as follows:

```
cust_preconfig="vmo -r [ -o tunable=value ... ]"
```

Note: You must restart the system to use the **cust_preconfig** attribute to set tunable parameters.

route
Specifies the route value to be added into network routing tables. You must specify the following values, or leave a blank space for any value that you do not want to specify:

Destination IP
The host or network for directing the route to. Specify the value as a numeric address.

Destination subnet mask
The mask for determining which network the destination IP belongs to. Specify the value as a numeric address.

Gateway IP
The network to which the packets are sent. Specify the value as a numeric address.

Each value must be separated by a double colon (::), and each additional set of the three values must be separated by a comma (,). The format for the route attribute is as follows:

```
route="DestHostA::MaskHostA::GatewayHostA, DestHostB::MaskHostB::GatewayHostB, ..."
```

For values that do not apply, you can leave it as blank but they still must be separated by a double colon as in the following example:

```
route="1.2.3.4::::5.6.7.8"
```

When you add the route attribute, using the **nimadapters** command with the **-a info** flag, you must separate the value for route with a double colon, and you must separate each additional set of three values with a space.

Secondary Adapter File Stanza Errors

A secondary adapter stanza causes an error under any of the following conditions:

- The host name that was used in the stanza header for the definition cannot be resolved.
- A required attribute is missing.
- An invalid value was specified for an attribute.
- An attribute mismatch occurs. For example, if the interface_type is not en or et, you cannot specify cable_type=bnc or media_speed=1000_Full_Duplex.
- The stanza contains both a location attribute and an interface_name attribute.
- Secondary adapter definitions occur multiple times for the same adapter location and the same host name.
- Secondary adapter definitions occur multiple times for the same interface_name and the same host name.

If a secondary adapter stanza is incorrect, the errors are reported, the stanza is ignored, and the following input is processed without regard to the incorrect stanza.

Example Secondary Adapter File

The following is an example of how a secondary adapter file can look:

```
# Set default values.

 default:

     machine_type  = secondary

     subnet_mask   = 255.255.240.0

     network_type  = en

     media_speed   = 100_Full_Duplex

 # Define the machine "lab1"

 # Take all defaults and specify 2 additional attributes.

 # Unlike the case of the client definitions that are input to the

 # nimdef command, the secondary adapter definition includes at least

 # one required field that cannot be defaulted.

 lab1:

     netaddr = 9.53.153.233

     location = P2-I1/E1

 # Change the default "media_speed" attribute.

 default:

     media_speed   = 100_Half_Duplex


 # define the machine "test1"

 # Take all defaults and include a comment.

 test1:

     comments      = "This machine is a test machine."

# define a machine with a VIPA interface that uses interfaces en2 and en3.
 lab2:
     machine_type         = secondary
     interface_type       = vi
     interface_name       = vi0
     netaddr              = 9.53.153.235
     subnet_mask          = 255.255.255.0
     secondary_hostname   = lab3
     interface_attributes = "interface_names=en2,en3"

 # define a machine with an etherchannel adapter that uses the adapters at
 # the following location codes P1-I4/E1 and P1/E1
 lab4:
     machine_type         = etherchannel
     interface_type       = en
     interface_name       = en2
     netaddr              = 9.53.153.237
     subnet_mask          = 255.255.255.0
     multiple_physloc     = P1-I4/E1,P1/E1

 # define a machine with an etherchannel adapter that uses the
 # ent2 and ent3 adapters and uses mode 8023ad.
 lab6:
     machine_type         = etherchannel
     interface_type       = en
     interface_name       = en2
     netaddr              = 9.53.153.239
     subnet_mask          = 255.255.255.0
     adapter_attributes = "adapter_names=ent2,ent3 mode=8023ad"
```

Flags

Item	Description
`-a`	Assigns the following attribute=value pairs: **client=nim_*client_name*** Specifies the NIM client that will have a secondary adapter definition added or removed. This option allows you to define one secondary adapter for a client. To define multiple secondary adapters, use a stanza file. **info=*AttributeList*** When previewing or defining a secondary adapter, the info attribute must be used when the client attribute is specified. *AttributeList* is a list of attributes separated by commas. The attributes must be specified in the following order:interface_type,location,interface_name,cable_type,media_speed,netaddr,subnet_mask, interface_attributes, secondary_hostname, machine_type, adapter_attributes, multiple_physloc,bos_preconfig, cust_preconfig, route.Use lowercase n/a to specify that a value will not be used.
`-d`	Defines secondary adapters. A `Client.adapter` file is created in the *adapter_def* location for each valid secondary adapter definition. If the `nimadapters` command encounters existing secondary adapter definitions for a NIM client, the existing definitions are replaced.
`-f`	*SecondaryAdapterFileName* Specifies the name of the secondary adapter file.
`-p`	Displays a preview operation to identify any errors. This flag processes the secondary adapter file or info attribute but does not add adapter definitions to the NIM environment. The preview shows the following: • All complete and valid secondary adapter stanzas. • All invalid secondary adapter stanzas and the reason for failure. **Note:** Specify the `-p` flag to verify that all stanzas are correct before using the secondary adapter file for configuring secondary adapters.
`-r`	Removes the secondary adapter definitions of a specific client or all the clients listed in a secondary adapter stanza file. If the client attribute or secondary adapter stanza file are not specified, then all the secondary adapter definitions in the *adapter_def* resource will be removed.

Parameters

Item	Description
adapter_def	This parameter is required to run the `nimadapters` command. Specifies the *adapter_def* NIM resource that is the directory containing secondary adapter definition files. An *adapter_def* resource must be defined using the `nim -o define` operation before the *adapter_def* can be used with the `nimadapters` command.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Security

Access Control: You must have root authority to run the `nimadapters` command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To add the NIM secondary adapters described in the secondary adapters definition file secondary_adapters.defs to the my_adapter_def resource, type:

```
nimadapters -d -f secondary_adapters.defs my_adapter_def
```

2. To preview the client definition file secondary_adapters.defs, type:

```
nimadapters -p -f secondary_adapters.defs my_adapter_def
```

3. To define a NIM secondary adapter for a client called pilsner, type:

```
     nimadapters -d \

     -a info="en,P2-I1/E1,n/
a,bnc,1000_Full_Duplex,9.53.153.233,255.255.254.0,n/a,n/a,n/a,n/a,n/a" \

     -a client=pilsner my_adapter_def
```

4. To remove the NIM secondary adapter definitions for a client called pilsner from the my_adapter_def resource, type:

```
nimadapters -r -a client=pilsner my_adapter_def
```

5. To remove the NIM secondary adapter definitions for clients defined in the file `secondary_adapters.defs`, type:

```
nimadapters -r -f secondary_adapters.defs my_adapter_def
```

6. To remove all the NIM secondary adapter definitions from the my_adapter_def resource, type:

```
nimadapters -r my_adapter_def
```

Files

Item	Description
`/usr/sbin/nimadapters`	Contains the `nimadapters` command.

nimadm Command

Purpose

The **nimadm** (Network Install Manager Alternate Disk Migration) command is a utility that allows the system administrator to do the following actions:

- Create a copy of rootvg to a free disk (or disks) and simultaneously migrate it to a new version or release level of AIX.
- Using a copy of rootvg, create a new NIM mksysb resource that has been migrated to a new version or release level of AIX.

- Using a NIM mksysb resource, create a new NIM mksysb resource that has been migrated to a new version or release level of AIX.
- Using a NIM mksysb resource, restore to a free disk (or disks) and simultaneously migrate to a new version or release level of AIX.

The **nimadm** command uses NIM resources to perform these functions.

Syntax

Perform Alternate Disk Migration:

nimadm -l *lpp_source* **-c** *NIMClient* **-s** *SPOT* **-d** *TargetDisks* [**-a** *PreMigrationScript*] [**-b** *installp_bundle*] [**-z** *PostMigrationScript*] [**-e** *exclude_files*] [**-i** *image_data*] [-j *VGname*] [**-m** *NFSMountOptions*] [**-o** *bosinst_data*] [**-P** *Phase*] [-j *VGname*] [**-Y**] [**-F**] [**-D**] [**-E**] [**-V**] [{ **-B** | **-r** }]

Cleanup Alternate Disk Migration on client:

nimadm -C -c *NIMClient* **-s** *SPOT* [**-F**] [**-D**] [**-E**]

Wake-up Volume Group:

nimadm -W -c *NIMClient* **-s** *SPOT* **-d** *TargetDisks* [**-m** *NFSMountOptions*] [**-z** *PostMigrationScript*] [**-F**] [**-D**] [**-E**]

Put-to-sleep Volume Group:

nimadm -S -c *NIMClient* **-s** *SPOT* [**-F**] [**-D**] [**-E**]

Synchronize Alternate Disk Migration Software:

nimadm -M -s *SPOT* -l *lpp_source* [-d *device*] [-P] [-F]

mksysb to Client Migration:

nimadm -T *NIMmksysb* -c *NIMClient* -s *SPOT* -l *lpp_source* -d *TargetDisks* -j *VGname* -Y [-a *PreMigrationScript*] [-b *installpBundle*] [-z *PostMigrationScript*] [-i *ImageData*] [-m *NFSMountOptions*] [-o *bosinst_data*] [-P *Phase*] [-F] [-D] [-E] [-V] [-B | -r]

mksysb to mksysb Migration:

nimadm -T *NIMmksysb* -O *mksysbfile* -s *SPOT* -l *lpp_source* -j *VGname* -Y [-N *NIMmksysb*] [-a *PreMigrationScript*] [-b *installp_bundle*] [-z *PostMigrationScript*] [-i *image_data*] [-m *NFSMountOptions*] [-o *bosinst_data*] [-P *Phase*] [-F] [-D] [-E] [-V]

Client to mksysb Migration:

nimadm -c *nim_client* -O *mksysbfile* -s *SPOT* -l *lpp_source* -j *VGname* -Y [-N *NIMmksysb*] [-a *PreMigrationScript*] [-b *installp_bundle*] [-z *PostMigrationScript*] [-i *image_data*] [-m *NFSMountOptions*] [-o *bosinst_data*] [-P *Phase*] [-e *exclude_files*] [-F] [-D] [-E] [-V]

Description

The **nimadm** command is a utility that allows the system administrator to create a copy of **rootvg** to a free disk (or disks) and simultaneously migrate it to a new version or release level of AIX. The **nimadm** command uses NIM resources to perform this function.

There are several advantages to using the nimadm command over a conventional migration:

1. Reduced downtime. The migration is performed while the system is up and functioning normally. There is no requirement to boot from install media, and the majority of processing occurs on the NIM master.
2. The **nimadm** command facilitates quick recovery in the event of migration failure. As the **nimadm** command uses **alt_disk_install** to create a copy of **rootvg**, all changes are performed to the copy (altinst_rootvg). In the event of serious migration installation failure, the failed migration is cleaned up and there is no need for the administrator to take further action. In the event of a problem with the new (migrated) level of AIX, the system can be quickly returned to the pre-migration operating system by booting from the original disk.

3. The **nimadm** command allows a high degree of flexibility and customization in the migration process. This is done with the use of optional NIM customization resources: image_data, bosinst_data, exclude_files, pre-migration script, installp_bundle, and post-migration script.

Please note that this document provides information pertaining to the **nimadm** command. For complete coverage of **alt_disk_install**, NIM, migration, and other related install issues, refer to the latest editions of the following publications:

- "*Installation and migration* in guide"
- "*AIX Version 4.3 to 5L Migration Guide*", an IBM Redbooks publication

nimadm Local Disk Caching

Local disk caching allows the NIM master to avoid having to NFS write to the client, which can be useful if the `nimadm` operation is not performing optimally due to an NFS write bottle neck. If this function is invoked with the `-j` *VGname* flag, the `nimadm` command creates file systems on the specified volume group (on the NIM master) and uses streams to cache all of the data from the client to these file systems.

The advantages and disadvantages to this function are as follows:

`Advantages:`

1. Improved performance for **nimadm** operations that are on relatively slow networks.
2. Improved performance for **nimadm** operations that are bottle necked in NFS writes (NFS writes are very expensive).
3. Decreased CPU usage on the client.
4. Client file systems are not exported.

`Disadvantages:`

1. Cache file systems take up space on the NIM master (you must have enough space to host the client's `rootvg` file systems and migration space for each client)
2. Increased CPU usage on the master.
3. Increased I/O on the master (for optimal performance use a volume group (disk) that does not contain the NIM resource being used in the operation).

How to execute disk caching:

1. Make sure you are at the latest level of `bos.alt_disk_install.rte` on the NIM master.
2. Add the `-j` *VGName* flag to any `nimadm` operations. For example:

   ```
   nimadm -j rootvg ...
   ```

   or

   ```
   nimadm -j cachevg
   ```

You can exclude specific file systems (which are not involved in the migration) from being cached over the network (they are still copied locally to `altinst_rootvg` on the client). To specify a list of file systems to be excluded from network caching, you must create a file in the location of the SPOT resource that is used for the migration. To get the exact location of the SPOT path, enter:

```
# lsnim -a location SpotName
```

You must name the file in the following format:

```
Nim_Client.nimadm_cache.excl
```

Note: This file applies to the NIM client specified in *Nim_Client*. The full path should be:

```
Spot_Location/Nim_Client.nimadm_cache.excl
```

For example: `/nim_resources/520spot/usr/myclient.nimadm_cache.excl`.

To exclude a file system from caching, enter one file system (to be excluded) per line in this file. While excluding a file system, ensure that you:

1. Do not exclude any file systems that are involved in the migration process. In other words, these file systems contain software files that are migrated. This can lead to unpredictable results.
2. Do not (cannot) exclude the following AIX file systems: `/, /usr, /var, /opt, /home, and /tmp`.

With disk caching, the `nimadm` command changes the following four phases (all other phases remain the same) :

`Phase 2:` The NIM master creates local cache file system in specified target volume group (on the NIM master).

`Phase 3:` The NIM master populates the cache file systems with the client's data.

`Phase 9:` The NIM master writes all migrated data to the client's alternate `rootvg`.

`Phase 10:` The NIM master cleans up and removes the local cache file systems.

nimadm Requirements

The **nimadm** requirements are:

1. The NIM master must have the same level of **bos.alt_disk_install.rte** installed in its **rootvg** and the SPOT which is used to perform the migration. (Note: it is not necessary to install the **alt_disk_install** utilities on the client).
2. The selected **lpp_source** NIM resource, and selected SPOT NIM resource must match the AIX level to which you are migrating.
3. The NIM master must be at the same or higher AIX level then the level being migrated to.
4. The client (the system to be migrated) must be at AIX 4.3.3 or higher.
5. The client must have a disk (or disks) large enough to clone the **rootvg** and an additional 500 Megs (approximately) of free space for the migration. The total amount of required space depends on original system configuration and **nimadm** customization.
6. The target client must be a registered with the master as a standalone NIM client (see the **niminit** command for more information). The NIM master must be able to execute remote commands on the client using the **rshd** protocol.
7. The NIM master must be able to execute remote commands on the client using the **rshd** protocol.
8. The NIM master and client must both have a minimum of 128 megabytes of RAM.
9. A reliable network, which can facilitate large amounts of NFS traffic, must exist between the NIM master and the client. The NIM master and client must be able to perform NFS mounts and read/write operations.
10. The client's hardware and software must support the AIX level that is being migrated to and meet all other conventional migration requirements.
11. All application and database servers, such as DB2 and LDAP, must be stopped before you run the **nimadm** command to clone the rootvg of a client system. Otherwise, the application servers and the database servers do not start normally after the **nimadm** command operations are complete.

Note: If you cannot meet requirements 1-10, you must perform a conventional migration. If you cannot meet requirement 11, then migration is not possible.

Attention: Before performing a **nimadm** migration you must agree to all software license agreements for software to be installed. You can do this by specifying the **-Y** flag as an argument to the **nimadm** command or setting the **ADM_ACCEPT_LICENSES** environment variable to "yes".

nimadm Limitations

The following limitations apply to the **nimadm** command:

1. If the trusted computing base (TCB) is turned on in the client's **rootvg**, you must disable it (permanently), use the disk caching option (**-j**), or perform a conventional migration. This limitation

exists because TCB must access the file metadata, which is not accessible through network file system (NFS).

2. All NIM resources used by the **nimadm** command must be local to the NIM master.
3. Although there is almost no interference with the client's active **rootvg** volume group during the migration, the client might experience minor reduction in performance due to increased disk I/O, `biod` activity, and CPU usage associated with cloning the **alt_disk_install** command.
4. You might need to tune the NFS to optimize the **nimadm** performance.

NIM resources used by nimadm:

SPOT resource (-s flag)
The NIM spot resource is required for all **nimadm** operations (migration, cleanup, wake-up, sleep). All **nimadm** and **alt_disk_install** utilities that are used by the client are installed in this resource. It is not necessary to install **nimadm** software on the client. The NIM cust operation must be used to install the following file sets into the spot:

- Required: **bos.alt_disk_install.rte** (must match the NIM master's level).
- Optional message catalog: **bos.msg.*$LANG*.alt_disk_install.rte**

lpp_source resource (-l flag)
This NIM resource is the source of install images that are used to migrate the system. It is required for **nimadm** migration operations. The **lpp_source** must contain all system images for the level being migrated to (check the lpp_source images attribute in **lsnim -l lpp_source** output). It must also contain any optional `installp` images that need to be migrated.

pre-migration
This script resource that is run on the NIM master, but in the environment of the client's `alt_inst` file system that is mounted on the master (this is done by using the **chroot** command). This script is run before the migration begins.

post-migration
This script resource is similar to the **pre-migration** script, but it is executed after the migration is complete.

image_data
Specifies an **image_data** resource that is passed to **alt_disk_install** (as arguments to the **-i** flag). NIM allocates and mount this resource on the client before calling **alt_disk_install**.

exclude_files
Specifies an **exclude_files** resource that is passed to **alt_disk_install** (as an argument to the `-e` flag). NIM allocates and mount this resource on the client before calling **alt_disk_install**.

installp_bundle
This NIM resource specifies any additional software that the **nimadm** command installs after completing the migration.

bosinst_data
This NIM resource specifies various install settings that may be used by the **nimadm** command.

The nimadm Migration Process

The **nimadm** command performs migration in 12 phases. Each phase can be executed individually by using the **-P** flag. The **nimadm** phases must be run sequentially. The **nimadm** phases are as follows:

1. The master issues an **alt_disk_install** command to the client that makes a copy of the **rootvg** volume group to the target disks (coincidentally this is Phase 1 of the **alt_disk_install** process). In this phase **altinst_rootvg** (alternate rootvg) is created. If a target **mksysb** is specified, the **mksysb** is used to create a **rootvg** volume group by using local disk caching on the NIM master.
2. The master runs remote client commands to export all the **/alt_inst** file systems to the master. The file systems are exported as read/write with root access to the master. If a target **mksysb** is specified, the cache file systems are created based on the image data from the **mksysb**.
3. The master NFS mounts the file systems exported in Phase 2. If a target **mksysb** is specified, the **mksysb** archive is restored in the cache file systems that was created in phase 2.

4. If the pre-migration script resource is specified, the script is executed at this time.
5. System configuration files are saved. Initial migration space is calculated and appropriate file system expansions are made. "bos" is restored and the device database is merged (similar to a conventional migration). The migration merge methods are executed and some miscellaneous processing takes place.
6. The file sets of the system are migrated by using the **installp**. Any required RPM images are installed during this phase.
7. If the **post-migration** script resource is specified, the script is executed at this time.
8. **bosboot** is executed to create a client boot image that is written to the client's boot logical volume (**hd5**).
9. The mounts that are made on the master in phase 3 are removed.
10. The client exports that are created in phase 2 are removed.
11. The **alt_disk_install** is called again (phase 3 of **alt_disk_install**) to make final adjustments and place **altinst_rootvg** to sleep. The bootlist is set to the target disk (unless the `-B` flag is used). If an output **mksysb** is specified, the cache is archived into a **mksysb** file and made into a NIM **mksysb** resource.
12. Cleanup is executed to end the migration. The client is rebooted, if the `-r` flag is specified.

Note: The **nimadm** command supports migrating several clients simultaneous.

nimadm Cleanup Operation

This operation, indicated with the "**-C**" flag, is designed to clean up after a failed migration that for some reason did not perform a cleanup it self. It can also be used to clear a previous migration in order to perform a new migration.

nimadm Wake-up and Sleep

After a migration completes, the **nimadm** command can be used to "wake-up" the migrated **altinst_rootvg** or the original **rootvg** (if booted from the migrated disk). The **nimadm** wake-up (**-W** flag) performs an **alt_disk_install** wake-up, NFS exports the **/alt_inst** file systems, and mounts them on the NIM master. The **nimadm** sleep function (**-S** flag) reverses the wake-up by unmounting the NIM master mounts, unexporting the **/alt_inst** file systems, and executing the **alt_disk_install** sleep function on the client.

Flags

Item	Description
-a *PreMigrationScript*	Specifies the pre-migration NIM script resource.
-b *installp_bundle*	Specifies the installp_bundle NIM resource.
-B	Specifies not running **bootlist** after the **nimadm** migration. If set, then **-r** flag cannot be used.
-c *ClientDisks*	Specifies the NIM defined client which is the target of this **nimadm** operation. This flag is required for all **nimadm** operations.
-C	Performs **nimadm** cleanup.
-d *TargetDisks*	Specifies the client target disk which is used to create **altinst_rootvg** (the volume group that is migrated).
-D	Sets the **nimadm** command into debug mode. This function must only be used to debug **nimadm** related problems and is not set by default.
-e *exclude_files*	Specifies the **exclude_files** NIM resource. This resource is used by the **alt_disk_install** command during Phase 1.
-E	Enters the **nimadm** debugger if a serious migration error occurs.

Item	Description
-F	Forces a client to unlock. Normally, the **nimadm** command locks a client to perform various operations. While the client is locked, other **nimadm** or NIM operations cannot be performed. This flag must ONLY be used in the unusual condition that a client is incorrectly locked (this can happen if for some reason the **nimadm** command could not call cleanup after a failure).
-i *image_data*	Specifies the **image_data** NIM resource. This resource is used by the **alt_disk_install** command during Phase 1 and 11.
`-j` *VGname*	Creates file systems on the specified volume group (on the NIM master) and uses streams to cache all of the data from the client to these file systems.
-l *lpp_source*	Specifies the lpp_source NIM resource to be used for this **nimadm** operation. This flag is required for migration operations.
-m *NFSMountOptions*	Specifies arguments that are passed to the mount command that **mounts** client resources on the master. This flag can be used to tune **nimadm** related NFS performance.
`-M`	Verifies that the levels of the alt_disk_install software (bos.alt_disk_install) on the NIM master, SPOT, lpp_source, and optional device are synchronized (match). If there is no match, the `nimadm` command installs the highest level found in the lpp_source or optional device.
`-N` *NIMmksysb*	Specifies the unique new NIM mksysb resource to create. If the `-N` flag is specified, the `-0` flag must be specified.
-o *bosinst_data*	Specifies **bosinst_data** NIM resource.
`-0` *mksysbfile*	Specifies the file pathname for the migrated mksysb. If the `-0` flag is specified, the `-j` flag and either the `-c` or `-T` flag must be specified.
-P *Phase*	The phase to execute during this invocation of the **nimadm** command. If there is more then one phase, the phases must be separated by spaces or commas. Valid phases are 1 through 12.
-r	Specifies that the client must reboot after **nimadm** migration is complete.
-s *SPOT*	Specifies the **SPOT** NIM resource to be used for this **nimadm** operation. This flag is required for all **nimadm** operations.
-S	Performs the **nimadm** "sleep" function. This function must be executed to end a **nimadm** "wake-up".
`-T` *NIMmksysb*	Specifies an existing NIM mksysb resource to migrate. If the `-T` flag is specified, the `-j` flag and either the `-0` or `-c` flag must be specified.
-V	Turns on verbose output.
-W	Performs the **nimadm** "wake-up" function.
-Y	Agrees to required software license agreements for software to be installed.
-z *PostMigrationScript*	Specifies the post-migration NIM script resource.

Exit Status

0
All the **nimadm** command related operations completed successfully.

>0
An error occurred.

Security

Access Control: You must have root authority to run the `nimadm` command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To execute **nimadm** migration to target NIM client aix1, using NIM **SPOT** resource spot1, NIM **lpp_source** resource lpp1, and target disks hdisk1 & hdisk2. Note that the **-Y** flag agrees to all required software license agreements for software to be installed, enter the following:

   ```
   nimadm -c aix1 -s spot1 -l lpp1 -d "hdisk1 hdisk2" -Y
   ```

2. To execute the same operation as in the example above to hdisk2, and also run pre-migration script nimscript1 and post-migration script nimscript2, type the following:

   ```
   nimadm -c aix1 -s spot1 -a nimscrip1 -z nimscript2 -l lpp1 -d hdisk1 -Y
   ```

3. To execute **nimadm** cleanup on client aix1, using NIM **SPOT** resource spot1, type the following:

   ```
   nimadm -C -c aix1 -s spot1
   ```

4. To create a migrated new mksysb resource of a client with the filename nim1, type the following:

   ```
   nimadm -c aix1 -s spot1 -l lpp1 -O /export/mksysb/mksysb1 -j vg00 -Y -N nim1
   ```

5. To create a new migrated mksysb resource with the filename nim3 from an existing NIM mksysb resource, type the following:

   ```
   nimadm -s spot1 -l lpp1 -j vg00 -Y -T nim2 -O /export/mksysb/m2 -N nim3
   ```

6. To migrate an existing NIM resource and put it on a client, type the following:

   ```
   nimadm -c aix1 -s spot1 -l lpp1 -d hdisk1 -j vg00 -T nim2 -Y
   ```

   Note: No changes are made to the nim2 NIM mksysb resource.

Files

Item	Description
/usr/sbin/nimadm	Contains the **nimadm** command.

nimclient Command

Purpose

Allows Network Installation Management (NIM) operations to be performed from a NIM client.

Syntax

To Enable or Disable the NIM Master's Push Permissions

nimclient { **-p** } | { **-P** }

To Enable or Disable Cryptographic Authentication for NIM Master Push Operations

nimclient { **-c** } | { **-C** }

To List Information about the NIM Environment

nimclient -l *LsnimParameters*

To Set the Date and Time to That of the NIM Master

nimclient -d

To Perform a NIM Operation

nimclient -o *Operation* [**-a** *Attribute*=*Value*] ...

Description

The **nimclient** command is used by workstations that are NIM clients to pull NIM resources. This command can enable or disable the NIM master server's ability to initiate workstation installation and customization for the workstation. The **nimclient** command can be used to generate a list of available NIM resources or display the NIM resources that have already been allocated to the client. A limited set of NIM operations can also be performed by the **nimclient** command using the **-o** flag.

Flags

Item	Description
-a A*ttribute*=*Value*	Passes information to NIM operations. **From the master** Use the **lsnim -q** *Operation* **-t** *Type* command to get a list of valid attributes for a specific operation. **From the client** Use the **nimclient -l -q** *Operation* **-t** *Type* command to get a list of valid attributes for a specific operation.
-c	Enables SSL authentication during NIM master push operations. **Note:** OpenSSL certificates must be configured on the NIM master using the **nimconfig -c** command. The SSL certificate is copied from the NIM master when **nimclient -c** is executed.
-C	Disables SSL authentication and uses standard nimsh security during NIM master push operations.
-d	Sets the client's date and time to that of the master.
-l *Lsnim parameters*	Executes the **lsnim** command on the master using the **lsnim** parameters that you specify. All the parameters which you use with this option must adhere to the syntax rules of the **lsnim** command. Note that some **lsnim** syntax requires the use of a NIM object name. To find out what the NIM name is for your machine, look in the **/etc/niminfo** file.

Item	Description
-o *Operation*	Performs the specified operation. The possible operations are: **allocate** Allocates a resource for use. **bos_inst** Performs a BOS installation. **change** Changes an object's attributes. **check** Checks the status of a NIM object. **cust** Performs software customization. **deallocate** Deallocates a resource. **diag** Enables a machine to boot a diagnostic image. **maint_boot** Enables a machine to boot in maintenance mode. **reset** Resets an object's NIM state. **showres** Displays the contents of a NIM resource.
-p	Enables the NIM master to push commands.
-P	Removes the NIM master's permissions to push commands. **Note:** The master can override this restriction by using the **-F** flag.

Security

Access Control: You must have root authority to run the **nimclient** command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list all the NIM resources which are available to this machine when its NIM name is `pluto`, enter:

   ```
   nimclient -l -L pluto
   ```

2. To list all the Shared Product Object Trees (SPOTs) which are available to this machine when its NIM name is `pluto`, enter:

   ```
   nimclient -l -L -t spot pluto
   ```

3. To list the operations which may be initiated from this machine, enter:

   ```
   nimclient -l -p -s pull_ops
   ```

4. To prevent the NIM master from running commands locally on the client, enter:

```
nimclient -P
```

5. To allocate a spot resource named `myspot`, an **lpp_source** resource named `images`, and an **installp** bundle file name `dept_bundle`, enter:

```
nimclient -o allocate -a spot=myspot -a lpp_source=images \
-a installp_bundle=dept_bundle
```

6. To perform a base system installation after the required resources have been allocated, enter:

```
nimclient -o bos_inst
```

7. From a standalone client, to allocate an **lpp_source** and install a software product such that the image for the installable option, `adt`, is contained in the **lpp_source**, `images`, enter:

```
nimclient -o allocate -a lpp_source=images
```

Then enter:

```
nimclient -o cust -a filesets="adt"
```

8. From a standalone client, to allocate an **lpp_source** and install a software product such that the image for the installable option, `adt`, is contained in the **lpp_source**, `images`, and the name of the installable option is contained in the **installp_bundle**, `bundle3`, enter:

```
nimclient -o allocate -a lpp_source=images \
-a installp_bundle=bundle3
```

Then enter:

```
nimclient -o cust
```

9. To install all fileset updates associated with APAR `IX12345`, residing in the **lpp_source** `updt_images`, enter:

```
nimclient -o allocate -a lpp_source=updt_images
nimclient -o cust -afixes=IX12345
```

10. From the NIM stand-alone, to run a live update for an APAR `IX12345`, residing in the **lpp_source** `lpp_source1`, enter:

```
nimclient -o allocate -a lpp_source=lpp_source1
nimclient -o cust -a live_update=yes -a filesets=IX12345
```

11. To update all installed software on the client with the latest updates from the `updt_images` **lpp_source**, enter:

```
nimclient -o allocate -a lpp_source=updt_images
nimclient -o cust -afixes=update_all
```

12. To enable the system to boot in maintenance mode using a SPOT resource named `spot1`, enter:

```
nimclient -o maint_boot -a spot=spot1
```

This sets up the maintenance boot operation, but you must initiate the network boot locally.

13. To show the contents of the config script `script1`, enter:

```
nimclient -o showres -a resource=script1
```

14. To show the contents of the bosinst.data resource `bosinst_data1`, enter:

```
nimclient -o showres -a resource=bosinst_data1
```

15. To list all the filesets in the lpp_source `lpp_source1` relative to what is currently installed on the machine `machine1`, from the NIM client machine `machine1`, enter:

```
nimclient -o showres -a resource=lpp_source1
```

    The **reference** attribute is automatically supplied by the **nimclient** command.

16. To list all problems fixed by software on the lpp_source `lpp_source1`, use:

```
nimclient -o showres -a instfix_flags="T" -a resource=lpp_source1
```

17. To install the filesets listed in the NIM **installp_bundle** `client_bundle` using the **lpp_source** `client_images`, while automatically allocating these resources during the installation operation, enter:

```
nimclient -o cust -a installp_bundle=client_bundle \
-a lpp_source=client_images
```

18. To perform a base system installation while automatically allocating all applicable resources from the NIM resource group named `client_grp`, enter:

```
nimclient -o bos_inst -a group=client_grp
```

19. To perform a base system installation while automatically allocating all applicable resources from the NIM group defined as the default resource group on the master, enter:

```
nimclient -o bos_inst
```

20. To copy an SSL certificate and enable SSL authentication, type:

```
nimclient -c
```

    Note: OpenSSL must be installed on the NIM client prior to using this command option.

Files

Item	Description
/etc/niminfo	Contains variables used by NIM.

nimconfig Command

Purpose

Initializes the Network Installation Management (NIM) master package.

Syntax

To Initialize the NIM master package

nimconfig -a pif_name=*Pif* **-a netname=***Objectname* [**-a master_port=***PortNumber*] [**-a platform=***Value*] [**-a registration_port=***PortNumber*] [**-a ring_speed=***Speed* | **-a cable_type=***CableType*]

To Configure SSL for the NIM Environment

nimconfig -c

To Rebuild the /etc/niminfo file:

nimconfig -r

Description

The **nimconfig** command initializes the NIM master package. You must initialize the package before any other NIM commands can be used. When you use the **-a** flag to supply the proper attributes, the **nimconfig** command initializes the NIM environment by performing the following tasks:

- Defines a network object specified by the *ObjectName* parameter to represent the network to which the NIM master's primary interface, specified by the *Pif* parameter, is connected.
- Completes the definition of the NIM master by connecting it to the newly defined network object.
- Defines a resource object to represent the network boot resource, which is managed automatically by NIM.
- Defines a resource object to represent the customization scripts that NIM automatically builds to perform customization.
- Starts the **NIM** communications daemon, **nimesis**.

Flags

Item	Description

-a Assigns the following attribute=value pairs:

pif_name=*Pif*
Designates the primary network interface for the NIM master. This value must be a logical interface name (such as tr0 or en0) is in the available state.

master_port=*PortNumber*
Specifies the port number of the **nimesis** daemon used for NIM client communication.

platform=*Value*
Specifies the platform. The supported platforms are:

rs6K
Micro Channel-based, uniprocessor models for AIX 5.1 and earlier

rs6ksmp
Micro Channeled-based, symmetric multiprocessor models for AIX 5.1 and earlier

rspc
PowerPC PCI bus-based, uniprocessor models for AIX 5.1 and earlier

rspcsmp
PowerPC PCI bus-based, symmetric multiprocessor models for AIX 5.1 and earlier

netname=*ObjectName*
Specifies the name you want the **nimconfig** command to use when creating the network object to represent the network to which the master's primary interface connects.

ring_speed=*Speed*
Speed in Mbps. When the **pif_name** refers to a token ring network, this value must be given. Acceptable values are:

4

16

cable_type=*CableType*
Specifies the ethernet cable type. When the **pif_name** refers to an ethernet network, this value must be given. Acceptable values are:

bnc

dix

N/A

registration_port=*PortNumber*
Specifies the port number used for NIM client registration.

Note: If you do not specify port numbers on the command line, the port numbers in the **/etc/services** file for NIM are used. If the **/etc/services** file does not contain entries for the NIM ports `nim` and `nimreg`, the default values of 1058 for **master_port** and 1059 for **registration_port** are used.

-c When OpenSSL is installed on the NIM master, this option creates SSL keys and certificates for use during NIM client communication. The SSL certificates are later copied to NIM clients using the **nimclient -c** command.

-r Rebuilds the **/etc/niminfo** file on the master using the information already exists in the NIM database. Note that if the **bos.sysmgt.nim.master** package has not been configured on this machine, this option will fail. This option is provided in case the **/etc/niminfo** file is accidentally removed by a user.

Security

Access Control: You must have root authority to run the **nimconfig** command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To initialize the NIM environment using token ring and the default NIM ports for network communications, type:

```
nimconfig -a pif_name=tr0 -a netname=net1 -a ring_speed=16
```

2. To initialize the NIM environment using ethernet and the default NIM ports, type:

```
nimconfig -a pif_name=en0 -a master_port=1058 \
-a netname = net2 -a cable_type=bnc
```

3. To rebuild the **/etc/niminfo** file on the NIM master when that machine has already been correctly configured as a master, type:

```
nimconfig -r
```

4. To initialize the NIM master using an ATM network interface, type:

```
nimconfig -a pif_name=at0 -a master_port=1058 -a netname=ATMnet
```

   Note: Because an interface to an ATM network does not currently support booting over the network, this operation will define a generic network object corresponding to the master's subnet.

5. To initialize the NIM environment using TCP/IP port 1060 for NIM client communications and TCP/IP port 1061 for NIM client registration, type:

```
nimconfig -a pif_name=tr0 -a netname=net2 -a master_port=1060 \
-a registration_port=1061 -a ring_speed=16
```

6. To create SSL keys and certificates for NIM communication, type:

```
nimconfig -c
```

   Note: OpenSSL must be installed on the NIM master prior to using this command option.

Files

Item	Description
/etc/niminfo	Contains variables used by NIM.

nimdef Command

Purpose

Defines Network Installation Management (NIM) clients from a stanza file.

Syntax

nimdef [**-p** | **-d** | **-c**] **-f** *Name*

Description

The **nimdef** command parses a definition stanza file to build the commands required to add NIM client definitions to the NIM environment.

The **nimdef** command can also create NIM networks and NIM machine groups automatically in the NIM environment to support the new client definitions.

> **Note:** Before using the **nimdef** command, you must configure the NIM master. (See **Basic NIM operations and configuration** in *Installation and migration* for more information.)

Client Definition File Rules

The format of the client definition file must comply with the following rules:

- After the stanza header, follow attribute lines of the form *Attribute* = *Value.*
- If you define an attribute value multiple times within the same stanza, only the last definition is used unless the attribute is **machine_group**. If you specify multiple **machine_group** attributes, all are applied to the machine definition.
- If you use an invalid attribute keyword, then that attribute definition is ignored.
- Each line of the file can have only one header or attribute definition.
- Only one stanza may exist in a definition file for each machine hostname.
- If the stanza header entry is the keyword **default**, this specifies to use it for the purpose of defining default values.
- You can specify a default value for any machine attribute except the machine hostname. If you do not specify an attribute for a machine but define a default value, then the default value is used.
- You can specify and change default values at any location in the definition file. After a default value is set, it applies to all definitions following it.
- To turn off a default value for all following machine definitions, set the attribute value to **nothing** in a default stanza.
- To turn off a default value for a single machine definition, set the attribute value to **nothing** in the machine stanza.
- You can include comments in a client definition file. Comments begin with the pound (**#**) character.
- When parsing the definition file for header/attribute keywords and values, tab characters and spaces are ignored.

Client Definition File Keywords

The client definition file uses the following keywords to specify machine attributes:

Required Attributes

Item	Description
cable_type	Specifies the cable type of the machine. Required if **network_type** is **ent**.
gateway	Specifies the hostname or IP address of the default gateway used by the machine. If the machine does not use a gateway, then specify the value **0** (zero) for this attribute.
machine_type	Specifies the type of the machine: **standalone**, **diskless**, or **dataless**.
network_type	Specifies the type of the machine's network adapter: **ent** or **tok**.
ring_speed	Specifies the ring speed of the machine. Required if **network_type** is **tok**.
subnet_mask	Specifies the subnet mask used by the machine.

Optional Attributes

Item	Description
nim_name	Specifies the NIM name to use for a machine. Use this attribute if something other than the hostname is used for the NIM name. By default, the NIM name given to a machine is the hostname of the machine with any domain information stripped off. If you use non-unique hostnames in different domains, a conflict occurs because the same NIM name is used for both machines. In such an environment, define this attribute for the affected machine definitions.
platform	Specifies the machine hardware platform. If you do not specify this attribute, default is **rs6k** through AIX 5.1 only.
net_adptr_name	Specifies the name of the network adapter used by the machine (**tok0**, **ent0**, etc.).
netboot_kernel=*NetbootKernelType*	Specifies the type of kernel to use when booting the client over the network. The **netboot_kernel** values are **up** or **mp**.
ipl_rom_emulation	Specifies the device to use for IPL ROM emulation (**/dev/fd0**, **/dev/rmt0**, etc.).
primary_interface	Specifies the hostname used for the original machine definition. Use this attribute if the current stanza is only to define an additional interface to a machine that is defined in the NIM environment.
master_gateway	Specifies the gateway that the NIM master uses to reach this machine if this machine is on a different network. This attribute is not necessary if this machine is defined on a network that is already defined in the NIM environment, or if the NIM master network has a default gateway specified.
machine_group	Specifies the group or groups to add the machine to when it is defined.
comments	Specifies a comment to include in the machine definition. The comment string should be in double quotes (**"**).

Client Definition File Stanza Errors

A definition stanza is incorrect under any of the following conditions:

- The hostname used in the stanza header for the definition is unresolvable.
- A required attribute is missing.
- You specify an invalid value for an attribute.
- An attribute mismatch occurs. For example, you can not specify **network_type=tok** and **cable_type=bnc** in the same stanza.
- A group-type mismatch occurs. For example, you can not specify a group for a machine if the group includes standalone machines and you specify **machine_type=diskless**.
- Machine definitions occur multiple times for the same hostname.
- A machine definition occurs for a machine that is already defined in the NIM environment.
- The **primary_interface** value in a machine definition does not match the hostname of any defined machine or stanza definition.
- The **primary_interface** value in a machine definition matches the hostname of another machine definition, but that definition is incorrect.

Sample Client Definition File

These default values are for AIX 5.1 and earlier.

```
# Set default values.
default:
    machine_type  = standalone
    subnet_mask   = 255.255.240.0
    gateway       = gateway1
    network_type  = tok
    ring_speed    = 16
    platform      = rs6k
    machine_group = all_machines

# Define the machine "lab1"
# Take all defaults.
lab1:

# Define the machine "lab2"
# Take all defaults and specify 2 additional attributes.
# The machine "lab2" uses IPL ROM emulation, and will be added to
# the machine groups "all_machines" and "lab_machines".
lab2:
    ipl_rom_emulation = /dev/fd0
    machine_group     = lab_machines

# Define the machine "lab3"
# Take all defaults, but do not add the machine to the default
# group.
lab3:
    machine_group=

# Define the machine "lab4"
# Take all defaults, but do not add "lab4" to the default group
# "all_machines".
# Instead add it to the groups "lab_machines" and "new_machines".
lab4:
    machine_group =
    machine_group = lab_machines
    machine_group = new_machines

# Change the default "platform" attribute.
default:
    platform = rspc

# define the machine "test1"
# Take all defaults and include a comment.
test1:
    comments = "This machine is a test machine."
```

Flags

Item	Description
-c	Generates commands from a client definition file. This flag processes the definition file and generates the commands to add the definitions. The commands are not invoked but displayed as a KSH script that you can redirect to a file and invoke at a later time.
-d	Defines machines from a client definition file. This flag processes the definition file and invokes the commands to add the definitions to the NIM environment.
-f *Name*	Specifies the name of the client definition file.

Item	Description
-p	Displays a preview of the client definition file. This flag processes the definition file but does not add machines to the NIM environment. Displays the following: All complete and valid NIM definition stanzas. All additional interfaces that will be defined for machines. All invalid definitions stanzas and the reason for failure. All new machine groups and the members to add. All existing machine groups and the members to add. All network definitions to add to the NIM environment. The commands to invoke to add each new machine. The commands to invoke to add each additional machine interface. The commands to invoke to create new machine groups and add their members. The commands to invoke to add new members to existing machine groups. **Note:** We recommend that you specify the **-p** flag on a client definition file to verify that all stanzas are correct before using it for adding machines.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
!0	An error occurred.

Security

Access Control: You must have root authority to run the **nimdef** command.

Auditing Events: N/A

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To preview the client definition file **client.defs**, enter:

```
nimdef -p -f client.defs
```

2. To add the NIM clients described in the client definition file **client.defs**, enter:

```
nimdef -d -f client.defs
```

3. To create a kshell script called **client.add** to add the NIM clients described in the client definition file **client.defs**, enter:

```
nimdef -c -f client.defs > client.add
```

Files

Item	Description
/usr/sbin/nimdef	Contains the **nimdef** daemon/command.

niminit Command

Purpose

Configures the Network Installation Management (NIM) client package.

Syntax

To Configure the NIM Client Package

niminit{**-a name=***Name* **-a pif_name=***Pif* **-a master=***Hostname*} [**-a master_port=***PortNumber*] [**-a registration_port=***PortNumber*] [**-a cable_type=***Type* | **-a ring_speed=***Speed*] [**-a iplrom_emu=***Device*] [**-a platform=***PlatformType*] [**-a netboot_kernel=***NetbootKernelType*] [**-a adpt_add=***AdapterAddress*] [**-a is_alternate=** yes | no] [**-a connect=***value*] [**-a vlan_tag=***value*] [**-a vlan_pri=***value*]

To Rebuild the /etc/niminfo File

niminit {**-a name=***Name* **-a master=***Hostname* **-a master_port=***PortNumber*}

Description

The **niminit** command configures the NIM client package. This must be done before the **nimclient** command can be used. When the required attributes are supplied to the **niminit** command, a new machine object will be created to represent the machine where the **niminit** command is being executed. When the **niminit** command completes successfully, the machine will be able to participate in the NIM environment.

After the NIM client package has been successfully configured, the **niminit** command can be run again to rebuild the **/etc/niminfo** on the client. The **/etc/niminfo** file is used by the **nimclient** command and must be rebuilt if it is accidentally removed by a user.

This command configures an `alternate_master` when the `is_alternate` attribute is set to `yes`. The `bos.sysmgt.nim.master` fileset must be installed prior to configuring an `alternate_master`. Once the configuration of an `alternate_master` is successful, the master that it registered with will be able to run `alternate_master` operations on this machine.

Flags

Item	Description	Attribute Description
-a	Specifies up to five different attributes for the **niminit** command. All of the following attribute=value pairs are preceded by the **-a** flag:	
	name=*Name*	Specifies the name that NIM will use to identify the workstation. This value is required.
	pif_name=*Pif*	Defines the name of the network interface for all NIM communications. This value is required.

Item	Description	Attribute Description
	master=*Hostname*	Specifies the hostname of the NIM master. The client must have the ability to resolve this hostname to an Internet Protocol (IP) address. This value is required.
	master_port=*PortNumber*	Specifies the port number of the **nimesis** daemon used for NIM communications.
	cable_type=*CableType*	Specifies the ethernet cable type. When the **pif_name** refers to an ethernet network, this value must be given. Acceptable values are: **bnc**, **dix**, and **N/A**.
	ring_speed=*Speed*	Speed in Mbps. When the **pif_name** refers to a token ring network, this value must be given. Acceptable values are: **4** and **16**.
	iplrom_emu=*Device*	Specifies a device that contains a ROM emulation image. This image is required for models that do not have internal support for booting via network interface.
	platform=*PlatformType*	Specifies the platform that corresponds to the client's machine type. If this attribute is not specified, the default, **chrp**, will be used. The supported platforms are: **chrp** PowerPC Common Hardware Reference Platform (CHRP) architecture-based machines **rs6k** Micro Channel-based, uniprocessor models for AIX 5.1 and earlier **rs6ksmp** Micro Channel-based, symmetric multiprocessor models for AIX 5.1 and earlier **rspc** PowerPC PCI bus-based, uniprocessor machines for AIX 5.1 and earlier **rspcsmp** PowerPC PCI bus-based, symmetric multiprocessor machines for AIX 5.1 and earlier
	adpt_add=*AdapterAddress*	Specifies the hardware address that corresponds to the network adapter.
	registration_port=*PortNumber*	Specifies the port number used for NIM client registration. **Note:** 1. If you do not specify port numbers on the command line, the port numbers in the **/etc/services** file for NIM is used. If the **/etc/services** file does not contain entries for the NIM ports `nim` and `nimreg`, the default values of 1058 for **master_port** and 1059 for **registration_port** are used. 2. The values used for **master_port** and **registration_port** should match the values used by the NIM master. To display the values used by the NIM master, run the command **lsnim -l master** on the NIM master.

Item	Description	Attribute Description
	netboot_kernel= *NetbootKernelType*	Specifies the type of kernel to use when booting the client over the network. The **netboot_kernel** values are: **up** Kernel for uniprocessor machines **mp** Kernel for multiprocessor machines The default is **up**.
	`is_alternate=[yes|no]`	Set this to yes if this machine is to be configured as an `alternate_master`.
	connect=*value*	Specifies the communicating service used by the NIM client for remote execution of NIM commands. Value options are `shell` (for rsh) and `nimsh`. The default setting is `connect=shell`. This attribute is optional. If the is_alternate attribute is set to yes then `nimsh` is the default setting, and is the only valid value. Using the is_alternate attribute is optional.
	vlan_tag=*value*	Specifies the virtual logical area network (VLAN) identifier that is used for VLAN tagging. The ID is used to identify the VLAN to which the Ethernet frame must belong. The ID allows the network administrator to organize the client's communication logically rather than assigning the network to the subnet. The VLAN tagging value is used by NIM to perform a network boot of a client. The configuration of the VLAN tag communication must be handled outside of NIM before using the value. Valid values are 0 - 4094.
	vlan_pri=*value*	Specifies the virtual logical area network (VLAN) priority that is used for VLAN tagging. The priority value, along with the VLAN tag, is used to identify the VLAN to which the Ethernet frame must belong. The priority allows the network administrator to organize the client's communication logically rather than assigning the network to the subnet. The VLAN tagging value is used by NIM to perform a network boot of a client. The configuration of the VLAN tag communication must be handled outside of NIM before using the value. Valid values are 0 - 7.

Security

Access Control: You must have root authority to run the **niminit** command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To configure the NIM client package on a machine that has a BOOTP-enabled IPL ROM such that it will be known as `scuba` in the NIM environment, using `en0` as its primary interface and an ethernet cable type of `bnc`, and specifying that it communicates with the NIM master using the master's hostname of `manta` and the default NIM ports located in **/etc/services** for network install communications, type:

```
niminit -a name=scuba -a pif_name=en0 -a cable_type=bnc \
-a master=manta
```

2. To rebuild the **/etc/niminfo** file when it has accidentally been removed by a user, using a hostname of superman for the master's hostname and a port number of 1058, type:

```
niminit -a name=robin -a master=superman -a master_port=1058
```

3. To configure the NIM client package for AIX 5.1 and earlier on a machine that is a PowerPC PCI bus-based, uniprocessor system that has a BOOTP-enabled IPL ROM such that it will be known as starfish in the NIM environment, using en0 as its primary interface and an Ethernet cable type of dix, and specifying that it communicates with the NIM master using the master's host name of whale and a port number of 1058, type:

```
niminit -a name=starfish -a pif_name=en0 -a cable_type=dix \
-a master=whale -a master_port=1058 -a platform=rspc
```

4. To configure the NIM client, on a machine to be known as bluefish in the NIM environment, using at0 as its primary interface and specifying that it communicates with the NIM master using the master's host name redfish and a port number of 1058, type:

```
niminit -a name=bluefish -a pif_name=at0 -a master=redfish \
-a master_port=1058
```

   Note: Because an interface to an ATM network does not currently support booting over the network, this operation will define a machine object on the NIM master if a Generic network object corresponding to the client's subnet is already defined.

5. To configure the NIM client for AIX 5.1 and earlier on a machine that is a PowerPC PCI bus-based, symmetric multiprocessor system that has a BOOTP-enabled IPL ROM such that it will be it will be known as jellyfish in the NIM environment, using en0 as its primary interface and an Ethernet cable type of dix, and specifying that it communicates with the NIM master using the master's host name of whale and a port number of 1058, type:

```
niminit -a name=jellyfish -a pif_name=en0 -a cable_type=dix \
-a master=whale -a master_port=1058 -a platform=rspcsmp
```

6. To configure the NIM client package on a machine that will use an IPL ROM emulation in device /dev/fd0, such that it will be known as octopus in the NIM environment and uses tr0 as its primary interface and a ring speed of 16, and communicates with the NIM master using the master's hostname of dolphin and a port number of1700 for client communications and 1701 for client registration, type:

```
niminit -a iplrom_emu=/dev/fd0 -a name=octopus -a pif_name=tr0 \
-a ring_speed=16 -a master=dolphin -a master_port=1700 \
-a registration_port=1701
```

7. To configure this machine as an alternate_master with the NIM master dolphin and communicate over interface en0, type:

```
niminit -a is_alternate=yes -a name=octopus -a pif_name=en0 \
-a cable_type=bnc -a master=dolphin
```

Files

Item	Description
/etc/niminfo	Contains variables used by NIM.

niminv Command

Purpose

Allows system administrators to gather, conglomerate, compare, and download fixes based on installation inventory of NIM objects.

Syntax

To get installation inventory:

`niminv -o invget -a targets=`*object1,object2,...* [`-a location=`*path*] [`-a colonsep=yes|no`]

To conglomerate installation inventory:

`niminv -o invcon -a targets=`*object1,object2,...* [`-a base=highest|lowest`] [`-a location=`*path*] [`-a colonsep=yes|no`]

To compare installation inventory:

`niminv -o invcmp -a targets=`*object1,object2,...* [`-a base=object|any`] [`-a location=`*path*]

To get fixes based on conglomerate inventory:

`niminv -o fixget -a targets=`*object1,object2,...* [`-a download=yes|no`] [`-a lp_source=`*object*] [`-a location=`*path*] `-a newlppname=`*name*

Description

The `niminv` command (Network Install Manager Inventory) allows system administrators to accomplish the following tasks:

- Gather installation inventory of several NIM objects.
- Conglomerate installation inventory of several NIM objects.
- Compare installation inventory of several NIM objects.
- Download fixes base on the installation inventory of several NIM objects.

The `niminv` command can use any NIM object that contains installation information. Examples of such objects include standalone client objects, SPOT objects, `lpp_source` objects and `mksysb` objects.

Using the `niminv` command has the following advantages:

- Hardware installation inventory is gathered alongside the software installation inventory.
- Data files are saved with a naming convention that is easily recognizable.
- All NIM objects that have installation inventory can be used.
- The command provides a holistic view of all managed NIM objects.

The information displayed by `niminv` can be limited by any of the following factors:

- Only software installation inventory is provided for objects that do not actually have physical devices (such as SPOT objects, `lpp_source` objects, and `mksysb` objects).
- Software and hardware installation inventory on client objects are limited to what commands on the remote system can provide.
- The recognition of fixes to download is based on the fix backend server. For more details, see **Using the Software Service Management menu (including SUMA)**.

Flags

Item	Description
-a *attribute=value*	Specifies the attribute and value. The supported attributes and values are based on the operation.
-o *operation*	Specifies the operation. The following operations are currently supported: **fixget** Gathers the latest fixes based on the installation inventory. This operation supports the following attributes: **targets** (required) A comma-separated list of NIM objects to base the gathering of fixes. **lpp_source** (optional) The NIM `lpp_source` object to use as a filter for downloading fixes. If the location and `newlppname` attributes are not used, this `lpp_source` object will also be where any fixes are downloaded to. **location** (optional) A directory to store the fixes. Use this attribute only if the fixes should not be downloaded to the object supplied to the `lpp_source` attribute. This attribute can only be used with the `newlppname` attribute. **newlppname** (optional) The NIM object name of the `lpp_source` to create at location. This attribute can only be used with the `location` attribute. The value supplied must be distinct and currently unused in the NIM environment. **download** (optional) Instructs the command whether or not to download the fixes. If no `lpp_source` or `location` field is specified and the value of this attribute is `yes`, fixes will be downloaded to the default location through the `suma` command. **Note:** The `suma` command will increase the file system space according to the `MaxFSSize` field in the `suma` configuration.

Item	Description
-o *operation (Continued)*	**invcmp** Compares installation inventory. This operation supports the following attributes: **targets** (required) A comma-separated list of NIM objects to compare installation inventory. **base** (required) The NIM object to use as the comparison base, or the keyword `any`. If the NIM object is supplied, the installation inventory in the object is the sole determinate of the data displayed, and only inventory in the base object is compared against inventory in the target objects. The keyword `any` forces the command to use any installation inventory of the targets. **location** (optional) A directory to store the data files. If this option is used, each inventory is saved with the format `conglomerate`*base_object*.*target_object_list*.*timestamp*, where *base_object* is the NIM name of the base object used for comparison (or the keyword `any`), *target_object_list* is a colon-separated and sorted list of the NIM name of the objects, and *timestamp* is the time the command was run (*year month day hour minute second*). If the directory does not exist, it will be created. The default is to display the data to the screen.

Item	Description
-o *operation (Continued)*	**invcon** Conglomerates installation inventory. This operation supports the following attributes: **targets** (required) A comma-separated list of NIM objects to conglomerate installation inventory. **base** (optional) Specifies whether the conglomerate inventory is based on the highest or lowest software levels. **location** (optional) A directory to store the data files. If this option is used, each inventory is saved with the format `base`.*target_object_list*.*timestamp*, where `base` indicates wther the conglomerate is based on the highest or lowest levels, *target_object_list* is a colon-separated and sorted list of the NIM name of the objects, and *timestamp* is the time that the command was run (*year month day hour minute second*). If the directory does not exist, it will be created. The default is to display the data to the screen. **colonsep** (optional) Instructs the command whether or not to produce colon-separated output. The default is no. **invget** Gathers installation inventory. This operation supports the following attributes: **targets** (required) A comma-separated list of NIM objects to gather installation inventory. **location** (optional) A directory to store the data files. If this option is used, each inventory is saved with the format `conglomerate`.*target_object_name*.*timestamp*, where *target_object_name* is the NIM name of the object, and *timestamp* is the time that the command was run (*year month day hour minute second*). If the directory does not exist, it will be created. The default is to display the data to the screen. **colonsep** (optional) Instructs the command whether or not to produce colon-separated output. The default is no.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

Access Control: You must have root authority to run the `niminv` command.

Attention RBAC users and Trusted AIX users: The **rbacqry** command grants execute (x) access to the root user. The **rbacqry** command is a privileged command that is used to run privilege operations. You must activate a role that has the authorization to run the command successfully.

Examples

1. To gather installation inventory of a two clients and save the output to /tmp/inventory, enter:

   ```
   niminv -o invget -a targets=client1,client2 -a location=/tmp/inventory
   ```

   Output similar to the following is displayed:

   ```
   Installation Inventory for client1 saved to
       /tmp/inventory/inventory.client1.060406140453.
   Installation Inventory for client2 saved to
       /tmp/inventory/inventory.client2.060406140453.
   ```

   The information in the files is similar to the output of lslpp -L
2. To conglomerate installation inventory of two clients and save the output to /tmp/inventory, enter:

   ```
   niminv -o invcon -a targets=client1,client2 -a location=/tmp/inventory
   ```

   Output similar to the following is displayed:

   ```
   Installation Inventory for client1 saved to
       /tmp/inventory/conglomerate.client1:client2.060406140500.
   ```

   The information in the files is similar to the output of lslpp -L.
3. To compare installation inventory of a mksysb, SPOT, and lpp_source to what's currently installed on the master, and save the output to /tmp/inventory, enter:

   ```
   niminv -o invcon -a targets=mksysb1,spot1,lpp_source1 -a base=master -a \
   location=/tmp/inventory
   ```

   Output similar to the following is displayed:

   ```
   Installation Inventory for client1 saved to
       /tmp/inventory/comparison.master.mksysb1:spot1:lpp_source1.060406140610.
   ```

   The information in the file is listed in column format. The comparison only includes installation inventory on the master.
4. To do the same comparison as in the preceding example but also include software on any of the objects, enter:

   ```
   niminv -o invcon -a targets=mksysb1,spot1,lpp_source1,master -a base=any -a \
   location=/tmp/inventory
   ```

   Output similar to the following is displayed:

   ```
   Installation Inventory for client1 saved to
       /tmp/inventory/comparison.any.mksysb1:spot1:lpp_source1.060406140733.
   ```

   The information in the file is listed in column format. The comparison includes any installation inventory in any of the target objects.
5. To see the fixes that can be downloaded based on the lowest installations in a mksysb, SPOT and lpp_source, enter:

   ```
   niminv -o fixget -a targets=mksysb1,spot1,lpp_source1
   ```

   Output similar to the following is displayed:

   ```
   ****************************************
   Performing preview download.
   ****************************************
   ```

```
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/Java14.debug.1.4.1.0.bff
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/Java14.debug.1.4.1.7.bff
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/devices.pci.00100100.com.5.2.0.50.bff
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/devices.pci.00100300.diag.5.2.0.75.bff
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/devices.pci.00100f00.rte.5.2.0.85.bff
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/devices.pci.13100560.rte.5.2.0.85.bff

Summary:
       6 downloaded
       0 failed
       0 skipped
```

6. To download the latest fixes based on the lowest installations in a mksysb, SPOT and lpp_source, enter:

```
niminv -o fixget -a targets=mksysb1,spot1,lpp_source1 -a download=yes
```

   Output similar to the following is displayed:

```
Extending the /usr filesystem by 30 blocks.
File System size changed to 8126464

Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/Java14.debug.1.4.1.0.bff
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/Java14.debug.1.4.1.7.bff
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/devices.pci.00100100.com.5.2.0.50.bff
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/devices.pci.00100300.diag.5.2.0.75.bff
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/devices.pci.00100f00.rte.5.2.0.85.bff
Download SUCCEEDED: /usr/sys/inst.images/installp/ppc/devices.pci.13100560.rte.5.2.0.85.bff

Summary:
       6 downloaded
       0 failed
       0 skipped
```

   Note: Any installations already contained in the default download path (as specified by the suma command) will not be downloaded again. The default download path in this example was /usr/sys/inst.images. Refer to the suma command for specifics on where the default download path will be.

7. To download the latest fixes based on the lowest installations in a mksysb, SPOT and lpp_source to an existing lpp_source, enter:

```
niminv -o fixget -a targets=mksysb1,spot1,lpp_source1 -a download=yes -a \
lpp_source=lpp_source2
```

   Output similar to the following is displayed:

```
Download SUCCEEDED: /nim/lpps/lpp_source2/installp/ppc/Java14.debug.1.4.1.0.bff
Download SUCCEEDED: /nim/lpps/lpp_source2/installp/ppc/Java14.debug.1.4.1.7.bff

Summary:
       2 downloaded
       0 failed
       0 skipped
```

   Note: Any installations already contained in lpp_source2 will not be downloaded again. In this example, the filesets device already existed in the lpp_source2.

8. To download the latest fixes based on the lowest installations in a mksysb, SPOT and lpp_source to a newlpp_source while filtering filesets in an existing lpp_source, enter:

```
niminv -o fixget -a targets=mksysb1,spot1,lpp_source1 -a download=yes -a \
location=/nim/lpps/newlpp1 -a newlppname=newlpp1
```

   Output similar to the following is displayed:

```
Download SUCCEEDED: /nim/lpps/newlpp1/installp/ppc/Java14.debug.1.4.1.0.bff
Download SUCCEEDED: /nim/lpps/newlpp1/installp/ppc/Java14.debug.1.4.1.7.bff

Summary:
       2 downloaded
       0 failed
       0 skipped
```

Note: Any installations already contained in `lpp_source2` will not be downloaded again. In this example, the `filesets` device already existed in the `lpp_source2`.

9. To download the latest fixes based on the lowest installations in a `mksysb`, SPOT and `lpp_source` to a new`lpp_source`, enter:

```
niminv -o fixget -a targets=mksysb1,spot1,lpp_source1 -a download=yes -a \
location=/nim/lpps/newlpp2 -a newlppname=newlpp2
```

Output similar to the following is displayed:

```
Download SUCCEEDED: /nim/lpps/newlpp2/installp/ppc/Java14.debug.1.4.1.0.bff
Download SUCCEEDED: /nim/lpps/newlpp2/installp/ppc/Java14.debug.1.4.1.7.bff
Download SUCCEEDED: /nim/lpps/newlpp2/installp/ppc/devices.pci.00100100.com.5.2.0.50.bff
Download SUCCEEDED: /nim/lpps/newlpp2/installp/ppc/devices.pci.00100300.diag.5.2.0.75.bff
Download SUCCEEDED: /nim/lpps/newlpp2/installp/ppc/devices.pci.00100f00.rte.5.2.0.85.bff
Download SUCCEEDED: /nim/lpps/newlpp2/installp/ppc/devices.pci.13100560.rte.5.2.0.85.bff

Summary:
       6 downloaded
       0 failed
       0 skipped
```

Location

`/usr/sbin/niminv`

nimol_backup Command

Purpose

Creates NIMOL install resources from an AIX client.

Note: This command is used only for Virtual I/O Server (VIOS) or Integrated Virtualization Management (IVM) from the Hardware Management Console (HMC).

Syntax

nimol_backup **-c** *client_hostname* [**-t** *directory*] [**-m** *remote_access_method*] [**-L** *label*] [**-D**]

Description

The **nimol_backup** command creates NIMOL install resources from a configured NIMOL client using the specified remote access method, which is **/usr/bin/rsh** by default, to call the nimol_mk_resources method on the client. When configuring a NIMOL server using the **nimol_config** command, the user can set the default remote access method to something other than **/usr/bin/rsh**, such as **/usr/bin/ssh**. A machine is considered a NIMOL client when it has been installed using the **nimol_install** command without the **-n** flag.

The command creates the target directory and label on the NIMOL server. The directory is then exported. The default label is `default`. For example, if the command is passed **-t /export/aix -L aix530**, then the command creates the **/export/aix/aix530** directory on the NIMOL server.

The command then uses the remote access method to run the **nimol_mk_resources** command. The **nimol_mk_resources** command creates the necessary install resources in the target directory.

Flags

Item	Description
-c *client_hostname*	Specifies the NIMOL client hostname on which to execute the **geninstall** command.

Item	Description
-D	Runs the command in debug mode.
-L *label*	Specifies the label or name to create for the created resources.
-m *remote_access_method*	Specifies the remote access method to use to run the **geninstall** command. The default **/usr/bin/rsh**. Another option is **/usr/bin/ssh**.
-t *directory*	Specifies the target directory where the AIX install resources will be created from the NIMOL client. The default directory is **/export/aix**.

Exit Status

Item	Description
0	The command completed successfully.
> 0	Error returned.

Security

To run the **nimol_backup** command on a NIMOL client, the client must provide remote access permissions to the NIMOL server. Using **/usr/bin/ssh** is a more secure remote acces method than **/usr/bin/rsh**.

Examples

1. To create install resources from client `myclient` in the **/export/aix** directory and named 530, type:

```
nimol_backup -c myclient -L 530 -t /export/aix
```

2. To execute `nimol_mk_resources` using `ssh`, type:

```
nimol_backup -c myclient -m ssh
```

Location

/usr/sbin/nimol_backup

Files

Item	Description
/etc/nimol.conf	Stores configuration information for the command.

nimol_config Command

Purpose

Configures a Linux server to network install a machine with AIX by configuring services and copying install resources.

Note: This command is used only for Virtual I/O Server (VIOS) or Integrated Virtualization Management (IVM) from the Hardware Management Console (HMC).

Syntax

nimol_config [**-d** *DirectoryContainingAIXResources*] [**-t** *TargetDirectoryToCopyResources*] [**-L** *InstallResourcesLabel*] [**-s** *NIMOLServerHostname*] [**-m** *RemoteAccessMethod*] [**-C**] [**-e**] [**-l**] [**-r**] [**-S**] [**-U**] [**-D**]

Description

The **nimol_config** command configures a Linux server to network install a machine with AIX. The command performs the following configuration.

1. First, the command obtains the hostname and IP address of the Linux server. If no hostname is specified with the **-s** flag, the command uses the hostname of the local machine and the IP address associated with the hostname. If a hostname and IP address are specified, then the pair is added to the **/etc/hosts** file, if it does not already exist.
2. The command then starts the portmap service and nfs server.
3. The command stores the remote access method in the **/etc/nimol.conf** file if specified with the **-m** flag. The default remote access command is **/usr/bin/rsh**, which is used to communicate with NIMOL clients that have been installed without specifying the **-n** flag to the **nimol_install** command.
4. Next, tftpboot is configured. The **/tftpboot** directory is created if it does not exist and the **/etc/xinetd.d/tftp** file is created if it does not exist. Then the command sets disable equal to no in the **/etc/xinetd.d/tftp** file and restarts xinetd so that the tftp server can handle incoming requests.
5. The **nimol_config** command also sets up syslog to accept incoming messages from other machines. Clients pass back status while installing to the syslog server. The **/etc/sysconfig/syslog** file is modified to include the **-r** flag in the SYSLOGD_OPTIONS or SYSLOGD_PARAMS variable. Then the command searches **/etc/syslog.conf** for the first available local log and sets it to write messages to **/var/log/nimol.log**. Clients write status to this log file, which can be monitored during a client installation. After the changes are made to the syslog configuration files, the service is restarted.
6. Next, the command sets up the DHCP server to receive bootp requests from AIX clients. The subnet of the NIMOL server is determined and added to the **dhcpd.conf** file. The options `allow bootp`, `not authoritative`, and `ddns-update-style none` are added if they do not already exist. Existing settings for these options will be overwritten.
7. Once the services have been configured, the **nimol_config** command attempts to copy AIX install resources locally, if the **-C** flag was not passed to the command. The command copys resources from the source directory specified with the **-d** flag (**/mnt/cdrom** by default) to the target directory (**/export/aix** by default). A directory is created (name that matches the LABEL name specified with the **-L** flag 'default' by default). The command looks in the source directory for the following resources:
   - a SPOT (Source Product Object Tree) directory named **/SPOT** and a SPOT directory named **ispot.tar.Z**
   - an lpp_source directory named **/lpp_source**
   - a mksysb named **mksysb** or **mksysb.bff**
   - a boot image named **booti.chrp.mp.ent**
   - a bosinst.data file named **bosinst.data**
   - an image.data file named **image.data**
   - a customization script named **cust.script**
   - a resolv.conf file named **resolv.conf**

   A SPOT, boot image, and either mksysb or lpp_source are required.
8. The target directory is then globally exported unless the **-e** flag is specified.
9. If a target directory and label are specified that contain resources, then these resources will be used and no resources will be copied. For example, if the command is passed **-t /export/aix -L aix530** and the directory **/export/aix/aix530** contains resource, then the command will not attempt to copy resources from the source directory.

10. After the NIMOL server has been configured, the **nimol_config** command will not attempt to reconfigure services on the NIMOL server when defining new resource labels.
11. The command also lists defined resource labels with the **-l** flag.
12. Resource labels can be removed by specifying the **-r** flag with a resource label. The command unexports the directory, if exported, and deletes the directory of the resource label.
13. When the **-U** flag is passed, the command attempts to undo any configuration that it has done, such as unconfiguring services.

Flags

Item	Description
-C	Specifies that the server should only configure services without copying install resources.
-d *directory*	Specifies the source directory that contains the AIX install resources. The default directory is **/mnt/cdrom**.
-D	Runs the command in debug mode.
-e	Instructs the command not to globally export the directory of newly created resource label.
-l	Lists the defined resource labels available to install a client.
-L *label*	Specifies the label or name to create for the copied resources.
-m *method*	Specifies the remote access method to use when running commands on clients that have been installed without specifying the **-n** flag to the **nimol_install** command.
-r	Instructs the command to remove the specified resource label.
-s *hostname*	The hostname to use for the NIMOL server. The default is to determine the hostname by running the **hostname** command.
-S	Instructs the command to not configure the syslog service. No status will be logged when clients are installing.
-t *directory*	Specifies the target directory where the AIX install resources will be copied from the source directory. The default directory is **/export/aix**.
-U	Instructs the command to unconfigure the NIMOL server. The command will attempt to undo any configuration that it performed.

Exit Status

Item	Description
0	The command completed successfully.
> 0	Error returned.

Security

Configuring the syslog service to accept messages from remote clients can be a security issue. Configure your firewall to only accept syslog messages from known clients.

Examples

1. To configure the NIMOL server without copying resources, type:

```
nimol_config -C
```

2. To configure the NIMOL server, copy resources from **/mnt/aix** to **/export/aix**, and label the resource `aix530`, type:

```
nimol_config -d /mnt/aix -t /export/aix -L aix530
```

3. To configure the NIMOL server and copy resources without configuring syslog and without globally exporting the resource label directory, type:

```
nimol_config -S -e
```

4. To list defined resource labels, type:

```
nimol_config -l
```

5. To remove the `aix530` resource label, type:

```
nimol_config -L aix530 -r
```

Location

/usr/sbin/nimol_config

Files

Item	Description
/etc/nimol.conf	Stores configuration information for the command.

nimol_install Command

Purpose

Sets up a configured NIMOL server to install AIX to a specific client machine.

Note: This command is used only for Virtual I/O Server (VIOS) or Integrated Virtualization Management (IVM) from the Hardware Management Console (HMC).

Syntax

nimol_install -c *client_hostname* [**-g** *gateway*] [**-m** *mac_address*] [**-p** *ip_address*] [**-s** *subnet_mask*] [**-L** *label*] [**-n**] [**-r**] [**-D**]

Description

The **nimol_install** command sets up a configured NIMOL server to network install a machine with AIX. The command peforms the following configuration.

1. The command determines the IP address of the client hostname if the client IP address isn't specified. If the client hostname isn't resolvable and a client IP address is specified, then the pair will be added to the **/etc/hosts** file if it does not exist.
2. The client is added to the **/etc/nimol.conf** file.
3. The directory of the resource label is exported to the client if it is not already globally exported.
4. A stanza for the client is added to the **/etc/dhcpd.conf** file. The client's subnet will also be added to the **/etc/dhcpd.conf** file if it does not exist. If the client or its subnet already exist in the **/etc/dhcpd.conf** file, an error is displayed.
5. A symbolic link to the boot image is created in the **/tftpboot** directory for the client.
6. A static `arp` entry is added if the client is on the same subnet as the NIMOL server.

7. The command will turn off the firewall rules to a client that is installing if the **iptables** command exists by running:

   ```
   iptables -I INPUT 1 -s client_hostname -j ACCEPT
   ```

   This allows the various services used by NIMOL to succeed. When a client is removed, the **nimol_install** command will run the following command to delete the rule: `iptables -D INPUT -s client_hostname`.
8. The command ensures that the required resources exist in the resource label's directory.
9. A nim_script is created in the scripts subdirectory of the resource label's directory if a **resolv.conf** or customization script was specified or if the client will remain a client of the NIMOL server after the installation. The **nimol_install** command will look for a general customization script in the resource label's directory named `cust.script` or a specific customization script for the client named *client_name*`.script`.
10. An information file is created in the **/tftpboot** directory that will be used during the installation of the operating system.
11. If the **-l** flag is specified, the command will list clients set up for an installation. A client will be removed if the **-r** flag is specified with a client name.
12. Once the client has been set up to install, the client must be told to perform a network install. If the client has AIX installed and is running, then use the **bootlist** command. For example, if the NIMOL server is `192.168.1.20` and the AIX client is `192.168.1.30`, then to boot off `ent0` run:

    ```
    bootlist -m normal -ent0 bserver=192.168.1.20 \\
      gateway=0.0.0.0 client=192.168.1.30
    ```

    then reboot by running:

    ```
    shutdown -Fr
    ```
13. If the client is not running, then boot into the SMS menus and specify the network boot parameters and the network boot device. If the client is on the same subnet as the NIMOL server, then the client will be able to do a broadcast **bootp** install. A broadcast **bootp** does not require the IP parameters to be set; the bserver, gateway and client would be `0.0.0.0` on a broadcast bootp install.

Flags

Item	Description
-c *client_hostname*	Specifies the client hostname that will be set up for an install or will be removed.
-D	Runs the command in debug mode.
-g *gateway*	Specifies the gateway that will be configured after the client has installed AIX. This is required when installing a client.
-l	Lists the clients set up to install.
-L *label*	Specifies the label or name of resources with which to install the client. The default is `default`.
-m *mac_address*	Specifies the MAC address of the network interface the client will install over. This is required when installing a client. The MAC address must contain colons (for example `00:60:08:3F:E8:DF`).
-n	Specifies not to configure the machine to remain a client of the NIMOL server after the installation has completed. If this option is specified, the client will not have its network configured after the installation.
-p *ip_address*	Specifies the IP address of the client. Use this flag if the client's hostname is not resolvable.

Item	Description
-r	Removes the client. The client will not be able to install AIX until it is reconfigured. This flag requires a client hostname.
-s *subnet_mask*	Specifies the subnet mask of the client interface. This flag is required when installing a client.

Exit Status

Item	Description
0	The command completed successfully.
> 0	Error returned.

Security

If the machine remains a client of the NIMOL server (the **-n** flag is not specified), then it will give the NIMOL server **/usr/bin/rsh** permissions so it can run commands on the client.

Examples

1. To setup client `myclient` to install the `aix530` resource label with gateway `192.168.1.1`, MAC address `00:60:08:3F:E8:DF`, and subnet mask `255.255.255.0`, type:

   ```
   nimol_install -c myclient -g 192.168.1.1 \\
     -m 00:60:08:3F:E8:DF -s 255.255.255.0 -L aix530
   ```

2. To setup client `myclient` and not have it remain a client to the NIMOL server after the installation, type:

   ```
   nimol_install -n -c myclient -g 192.168.1.1 \\
     -m 00:60:08:3F:E8:DF -s 255.255.255.0 -L aix530
   ```

3. To list the clients configured to be installed, type:

   ```
   nimol_install -l
   ```

4. To remove client `myclient`, type:

   ```
   nimol_config -c myclient -r
   ```

Location

/usr/sbin/nimol_install

Files

Item	Description
/etc/nimol.conf	Stores configuration information for the command.

nimol_lslpp Command

Purpose

Runs the **lslpp** command on a NIMOL client.

Note: This command is used only for Virtual I/O Server (VIOS) or Integrated Virtualization Management (IVM) from the Hardware Management Console (HMC).

Syntax

nimol_lslpp -c *client_hostname* [**-m** *remote_access_method*] [**-f** *lslpp_flags*] [**-D**]

Description

The **nimol_lslpp** command executes the **lslpp** command on a configured NIMOL client using the specified remote access method, which is **/usr/bin/rsh** by default. When configuring a NIMOL server using the **nimol_config** command, the user can set the default remote access method to something other than **/usr/bin/rsh**, such as **/usr/bin/ssh**. A machine is considered a NIMOL client when it has been installed using the **nimol_install** command without the **-n** flag.

The command runs the **lslpp** command with **-L -c** as the default flags. The **lslpp** command flags can be specified with the **-f** flag.

Flags

Item	Description
-c *client_hostname*	Specifies the NIMOL client hostname on which to execute the **lslpp** command.
-D	Runs the command in debug mode.
-f *lslpp_flags*	Specifies the **lslpp** command flags to pass to the **lslpp** command.
-m *remote_access_method*	Specifies the remote access method to use to run the **lslpp** command. The default is **/usr/bin/rsh**. Another option is **/usr/bin/ssh**.

Exit Status

Item	Description
0	The command completed successfully.
> 0	Error returned.

Security

To run the **nimol_lslpp** command on a NIMOL client, the client must provide remote access permissions to the NIMOL server. Using **/usr/bin/ssh** is a more secure remote access method than **/usr/bin/rsh**.

Examples

1. To run the **lslpp** command on client `myclient`, with the default flags **-Lc**, type:

   ```
   nimol_lslpp -c myclient
   ```

2. To run the **lslpp** command on client `myclient`, with the flags **-i bos.rte**, type:

   ```
   nimol_lslpp -c myclient -f "-i bos.rte"
   ```

3. To run the **lslpp** command on client `myclient`, using `ssh` as the remote access method, type:

   ```
   nimol_lslpp -c myclient -m ssh
   ```

Location

/usr/sbin/nimol_lslpp

Files

Item	Description
/etc/nimol.conf	Stores configuration information for the command.

nimol_update Command

Purpose

Runs geninstall on a NIMOL client to perform software maintenance.

Note: This command is used only for Virtual I/O Server (VIOS) or Integrated Virtualization Management (IVM) from the Hardware Management Console (HMC).

Syntax

nimol_update -c *client_hostname* [**-L** *label*] [**-f** *geninstall_flags*] [**-m** *remote_access_method*] [**-p** *package_list*] [**-D**]

Description

The **nimol_update** command executes the **geninstall** command on a configured NIMOL client using the specified remote access method, which is **/usr/bin/rsh** by default. When configuring a NIMOL server using the **nimol_config** command, the user can set the default remote access method to something other than **/usr/bin/rsh**, such as **/usr/bin/ssh**. A machine is considered a NIMOL client when it has been installed using the **nimol_install** command without the **-n** flag.

The command runs the **geninstall** command with **-acgX** as the default flags. Use the **-f** flag to specify **geninstall** command flags. The software packages to pass the **geninstall** command are specified with the **-p** flag.

When installing filesets using the **nimol_update** command, you must specify a resource label that has an lpp_source. Run **nimol_config -l -L *label*** to determine if a resource label contains an lpp_source. The command will export the resource label directory if it is not already globally exported. The client will mount the directory and use it as the source directory during an installation.

Flags

Item	Description
-c *client_hostname*	Specifies the NIMOL client hostname on which to execute the **geninstall** command.
-D	Runs the command in debug mode.
-f *geninstall_flags*	Specifies the flags to pass to the **geninstall** command. The default flags are **-acgX**.
-L *label*	Specifies the name of the resource label that will be used as the source for install images.
-m *remote_access_method*	Specifies the remote access method to use to run the **geninstall** command. The default is **/usr/bin/rsh**. Another option is **/usr/bin/ssh**.
-p *package_list*	Specifies the name of software packages to pass to the **geninstall** command. The default is `all`.

Exit Status

Item	Description
0	The command completed successfully.
> 0	Error returned.

Security

To run the **nimol_update** command on a NIMOL client, the client must provide remote access permissions to the NIMOL server. Using **/usr/bin/ssh** is a more secure remote access method than **/usr/bin/rsh**.

Examples

1. To install all packages in resource label 530 to client `myclient`, type:

```
nimol_update -c myclient -L 530
```

2. To apply an update for `bos.games` on client `myclient`, type:

```
nimol_update -c myclient -L 530 -f "-a" -p "bos.games"
```

3. To remove `bos.games` from client `myclient`, type:

```
nimol_update -c myclient -f "-u" -p "bos.games"
```

4. To execute the **geninstall** command using `ssh`, type:

```
nimol_update -c myclient -L 530 -m ssh
```

Location

/usr/sbin/nimol_update

Files

Item	Description
/etc/nimol.conf	Stores configuration information for the command.

nimquery Command

Purpose

Query a system in the Network installation management (NIM) environment for system information and create client objects in the environment.

Syntax

`nimquery`{{ `-a` host=*hostname* [`-a` name=*client obj*[`-d`] [`-a` hmc=*obj name*[`-d`] [`-a` cec=*obj name* [`-a` bcmm=*obj name* [`-a` ivm=*obj name*} [`-p`] [`-q`] [`-v`]

Description

The `nimquery` command queries a machine for system information when using the `-a` host parameter. The information is used for defining a new client object in the NIM environment. System information is provided from systems that use the NIM service handler (nimsh).

The `nimquery` command can also be used to query logical partitions (LPARs) , central electronics complex (CEC) and blades information when pointing to a Hardware Management Console (HMC) , CEC, Integrated Virtualization Manager (IVM) or Blade Center Management Module (BCMM) object. To do so, the **openssh.base.client** must be installed on the NIM master.

Flags

Item	Description
`-a`	Assigns the following parameter attribute=value pairs.
`-d`	Defines a new client object (requires the name attribute when `-a` host is used).
`-p`	Enables print format.
`-q`	Shows an attribute list for `nimquery` command.
`-v`	Enables verbose debug output during command execution.

Parameters

Item	Description
`host=`*hostname*	Specifies the host name of the system to query. This attribute is required.
`name=`*client_obj*	Specifies the name to assign the client object when creating a new definition in the NIM database.
`hmc=`*objname*	Specifies the object name of the HMC system to query. This attribute is required.
`cec=`*objname*	Specifies the object name of the CEC system to query. This attribute is required.
`ivm=`*objname*	Specifies the object name of the IVM system to query. This attribute is required.
`bcmm=`*objname*	Specifies the object name of the BCMM system to query. This attribute is required.

Exit Status

0
Returns zero on success.

Security

You must have root authority to run the `nimquery` command.

Examples

1. To query machine buckey for system information, type:

   ```
   nimquery -a host=buckey
   ```

2. To query machine buckey for system information and to provide detailed output information, type:

   ```
    nimquery -a host=buckey -p
   ```

3. To define machine buckey.austin.ibm.com using name client6 as the NIM object name, type:

   ```
   nimquery -a name=client6 -a host=buckey -d
   ```

4. To query Management Module bcmm2 for blade system information, type:

```
nimquery -a bcmm=bcmm2
```

5. To define CEC objects managed by HMC hmc1, type:

```
nimquery -a hmc=hmc1 -d
```

6. To query LPARs attached to cec1 buckey for system information, type:

```
nimquery -a cec=cec1
```

Files

Item	Description
`/usr/sbin/nimquery`	The location of `nimquery` command.

nistoldif Command

Purpose

Exports user, group, name resolution, and rpc data to rfc 2307-compliant form.

Syntax

nistoldif **-d** *Suffix* [**-a** *BindDN* **-h** *Host* **-p** *Password* [**-n** *Port*]] [**-f** *Directory*] [**-y** *domain*] [-S *Schema*] [**-k** *KeyPath* **-w** *SSLPassword*] [**-s** *Maps*] [**-m** *ldap_mapname*]

Description

The **nistoldif** command converts the data from **passwd**, **group**, **hosts**, **services**, **protocols**, **rpc**, **networks**, **netgroup**, and **automount** into forms compliant with rfc2307. It will first attempt to read data from NIS, and if it cannot find a NIS map it will fall back to the flat files.

If the server information (the **-a**, **-h**, and **-p** flags) is given on the command line, data will be written directly to the server. If any data conflicts with an entry already on the server, either because the entry already exists, or because the **uid** or **gid** already exists, a warning will be printed. If the server information is not given, the data will be written to **stdout** in LDIF. In either case, **nistoldif** does not add an entry for the suffix itself; if that entry does not exist, attempts to add data to the server will fail. This entry will be added during server setup, usually by the **mksecldap** command.

Translation is not exact. Because of the limitations of the rfc2307 definitions, some attributes are defined in a case-insensitive way; for example, TCP, Tcp, and tcp are all the same protcol name to the LDAP server. Uids and gids greater than 2^31-1 will be translated to their negative twos complement equivalent for storage.

The **nistoldif** command reads the **/etc/security/ldap/sectoldif.cfg** file to determine what to name the sub-trees that the passwd, group, hosts, services, protocols, rpc, networks and netgroup data will be exported to. The names specified in the file will be used to create sub-trees under the base DN specified with the **-d** flag. For more information, see the **/etc/security/ldap/sectoldif.cfg** file documentation.

Flags

Item	Description
-a	Specifies the administrative bind DN used to connect to the LDAP server. If this flag is used, **-h** and **-p** must also be used, and data will be written directly to the LDAP server.
-d	Specifies the suffix that the data should be added under.

Item	Description
-f	Specifies the directory to look for flat files in, or the name of the automount map file. If this flag is not used, **nistoldif** will look for files in **/etc**. This flag is required for automount maps.
-h	Specifies the host name which is running the LDAP server. If this flag is used, **-a** and **-p** must also be used, and data will be written directly to the LDAP server. This flag will be ignored for automount data.
-k	Specifies the SSL key path. If this flag is used, **-w** must also be used.
-m	Specifies the automount map on the LDAP server.
-n	Specifies the port to connect to the LDAP server on. If this flag is used, **-a**, **-h** and **-p** must also be used; if it is not used, the default LDAP port is used.
-p	Specifies the password used to connect to the LDAP server. If this flag is used, **-a** and **-h** must also be used, and data will be written directly to the LDAP server.
-s	Specifies a set of maps to be written to the server. This flag should be followed by a list of letters representing the maps that should be migrated. If this flag is not used, all maps will be migrated. The letters are: **a** for automount, **e** for netgroup, **g** for group, **h** for hosts, **n** for networks, **p** for protocols, **r** for rpc, **s** for services, and **u** for passwd.
-S	Specifies the LDAP schema to use for users and groups. This can be either RFC2307 or RFC2307AIX; RFC2307AIX gives extended AIX schema support. If this flag is not used, RFC2307 is the default.
-w	Specifies the SSL password. If this flag is used, **-k** must also be used.
-y	Specifies the NIS domain to read maps from. If this flag is not used, the default domain will be used.

Exit Status

This command returns the following exit values:

0
No errors occurred. Note that failure to find a map is not considered an error.

>0
An error occurred.

Security

Access Control: Only the root user can run this command.

Examples

1. To export the NIS maps from the domain **austin.ibm.com** (falling back to the flat files in **/tmp/etc**) to LDIF under the suffix **cn=aixdata**, type:

```
nistoldif -d cn=aixdata -y austin.ibm.com -f /tmp/etc > ldif.out
```

2. To export the hosts and services maps from the default domain (falling back to the flat files in **/etc**) to the LDAP server **ldap.austin.ibm.com** with administrator bind DN**cn=root** and password `secret` under the suffix **cn=aixdata**, type:

```
nistoldif -d cn=aixdata -h ldap.austin.ibm.com -a cn=root -p secret -s hs
```

3. To convert the **/etc/auto_master** automount map file into LDIF, type:

```
nistoldif -s a -f /etc/auto_master > ldif.out
```

4. In order to remove automount data, the LDIF file must be created manually. For example, suppose the user `user1` was erroneously added to the `auto_home` automount map in the `dc=austin,dc=ibm,dc=com` suffix, and needs to be deleted. Create the following LDIF:

```
# cat /tmp/del_user1.ldif
dn: automountKey=user1,automountMapName=auto_home,dc=austin,dc=ibm,dc=com
changetype: delete
```

   Then run the following command:

```
ldapmodify -f /tmp/del_user1.ldif
```

5. In order to edit automount data, the LDIF file must be created manually. For example, suppose the user `user2` was given the wrong mount point in the `auto_home` automount map in the `dc=austin,dc=ibm,dc=com` suffix, and needs to be changed to the correct location of `/home/user2`. Create the following LDIF:

```
# cat /tmp/ch_user2.ldif
dn: automountKey=user2,automountMapName=auto_home,dc=austin,dc=ibm,dc=com
changetype: modify
replace: automountInformation
automountInformation: /home/user2
```

   The run the following command:

```
ldapmodify -f /tmp/ch_user2.ldif
```

Files

Item	Description
/usr/sbin/nistoldif	Contains the **nistoldif** command.

nl Command

Purpose

Numbers lines in a file.

Syntax

nl [**-b** *Type*] [**-f** *Type*] [**-h** *Type*] [**-l** *Number*] [**-d** *Delimiter*] [**-i** *Number*] [**-n** *Format*] [**-v** *Number*] [**-w** *Number*] [**-p**] [**-s** *Separator*] [*File*]

Description

The **nl** command reads the *File* parameter (standard input by default), numbers the lines in the input, and writes the numbered lines to standard output. In the output, the **nl** command numbers the lines on the left according to the flags you specify on the command line.

The input text must be written in logical pages. Each logical page has a header, a body, and a footer section (you can have empty sections). Unless you use the **-p** flag, the **nl** command resets the line numbers at the start of each logical page. You can set line-numbering flags independently for the header, body, and footer sections (for example, the header and footer lines can be numbered while the text lines are not).

Signal the start of logical-page sections with lines in the file that contain only the following delimiter characters:

Line Contents	Start Of
\:\:\:	Header

Line Contents	Start Of
\:\:	Body
\:	Footer

You can name only one file on the command line. You can list the flags and the file name in any order.

Flags

All the parameters are set by default. Use the following flags to change these default settings. Except for the **-s** flag, enter a **-n** flag without a variable to see its default value.

Item	Description
-b *Type*	Chooses which body section lines to number. Recognized values for the *Type* variable are: **a** Numbers all lines **t** Does not number lines that are blank or lines that contain any non-graphic character such as a tab within them. (default) **n** Does not number any lines **p*Pattern*** Numbers only those lines specified by the *Pattern* variable.
-d *Delimiter*	Uses the two characters specified by the *Delimiter* variable as the delimiters for the start of a logical page section. The default characters are \: (backslash, colon). You may specify two ASCII characters, two 1-byte extended characters, or one extended character. If you enter only one 1-byte character after the **-d** flag, the second character remains the default (a colon). If you want to use a backslash as a delimiter, enter two backslashes (\\).
-f *Type*	Chooses which logical-page footer lines to number. The possible values for the *Type* variable are the same as the **-b** flag. The default value of the *Type* variable is **n** (no lines numbered).
-h *Type*	Chooses which logical-page header lines to number. The possible values for the *Type* variable are the same as the **-b** flag. The default value of the *Type* variables **n** (no lines numbered).
-i *Number*	Increments logical-page line numbers by the number specified in the *Number* variable. The default value of the *Number* variable is 1. The range of the *Number* variable is from 1 to 250.
-l *Number*	(Lowercase L) Uses the value specified in the *Number* parameter as the number of blank lines to count as one. For example, `-l3` numbers every third blank line in a series. The default value of the *Number* variable is 1. This flag works when the **-ha**, **-ba**, or **-fa** option is set. The range of the *Number* variable is from 1 to 250.
-n *Format*	Uses the value of the *Format* variable as the line numbering format. Recognized formats are: **ln** Left-justified, leading zeros suppressed **rn** Right-justified, leading zeros suppressed (default) **rz** Right-justified, leading zeros kept

Item	Description
-p	Does not restart numbering at logical page delimiters.
-s *Separator*	Separates the text from its line number with the character specified in the *Separator* variable. The default value of the *Separator* variable is a tab character.
-v *Number*	Sets the initial logical-page line number to the value specified by the *Number* variable. The default value of the *Number* variable is 1. The range of the *Number* variable is from 0 to 32767.
-w *Number*	Uses the value specified by the *Number* variable as the number of characters in the line number. The default value of the *Number* variable is 6. The range of the *Number* variable is from 1 to 20.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To number only the non-blank lines, enter:

   ```
   nl  chap1
   ```

   This displays a numbered listing of `chap1`, numbering only the non-blank lines in the body sections. If `chap1` contains no \:\:\+:, or \: delimiters, then the entire file is considered the body.

2. To number all lines:

   ```
   nl  -ba  chap1
   ```

   This numbers all the lines in the body sections, including blank lines. This form of the **nl** command is adequate for most uses.

3. To specify a different line number format, enter:

   ```
   nl  -i10  -nrz  -s::  -v10  -w4  chap1
   ```

   This numbers the lines of `chap1` starting with ten (`-v10`) and counting by tens (`-i10`). It displays four digits for each number (`-w4`), including leading zeros (`-nrz`). The line numbers are separated from the text by two colons (`-s : :`).

   For example, if `chap1` contains the text:

   ```
   A  not-so-important
   note  to  remember:

   You  can't  kill  time
   without  injuring  eternity.
   ```

   then the numbered listing is:

   ```
   0010::A  not-so-important
   0020::note  to  remember

   0030::You  can't  kill  time
   0040::without  injuring  eternity.
   ```

Note that the blank line was not numbered. To do this, use the **-ba** flag as shown in example 2.

Files

Item	Description
/usr/bin/nl	Contains the **nl** command.

nlssrc Command

Purpose

Gets the status of a subsystem or a group of subsystems in canonical form.

Syntax

nlssrc [**-h** *host*] **-a**

nlssrc [**-h** *host*] **-g** *group_name*

nlssrc [**-h** *host*] [**-l**] [**-c**] **-s** *subsystem_name*

nlssrc [**-h** *host*] [**-l**] [**-c**] **-p** *subsystem_pid*

The syntax for the first two usages of `nlssrc` will generate the exact same output as `lssrc`. The syntax for the last two usages will generate the output in the canonical form as `lssrc`.

Description

Use the `nlssrc` command to get the status of a subsystem or a group of subsystems in canonical form. For the AIX platform, use the `nlssrc -c` command to get language-independent output for supported subsystems from the `lssrc` command. The status is displayed in English regardless of the installed language locale. If the `-c` flag is not present, the `nlssrc` command will invoke the `lssrc` command that uses the daemon's locale.

Flags

Item	Description
-a	Lists the current status of all defined subsystems
-c	Requests the canonical `lssrc` output of the supported subsystems.
-g *group_name*	Specifies a group of subsystems to get status for. The command is unsuccessful if the *group_name* parameter is not contained in the subsystem object class.
-h *host*	Specifies the foreign host on which this status action is requested. The local user must be running as root. The remote system must be configured to accept remote System Resource Controller (SRC) requests. That is, the `srcmstr` daemon (see `/etc/inittab`) must be started with the `-r` flag and the `/etc/hosts.equiv` file or the `.rhosts` file must be configured to allow remote requests.

Item	Description
-l	Requests that a subsystem send its current status in long form. Long status requires that a status request be sent to the subsystem; it is the responsibility of the subsystem to return the status.
-p *subsystem_pid*	Specifies a particular instance of the *subsystem_pid* parameter to get status for, or a particular instance of the subsystem to which the status subserver request is to be taken.
-s *subsystem_name*	Specifies a subsystem to get status for. The *subsystem_name* parameter can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the *subsystem_name* parameter is not contained in the subsystem object class.

Security

You do *not* need `root` authority to run this command.

Exit Status

0
Command has run successfully.

1
Command was not successful.

Restrictions

This command applies to the `cthags` and `cthats` subsystems only.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output.

Standard Error

Error messages are written to standard error (and to the `ctsnap.`*host_name*`.nnnnnnnn.log` file).

Examples

1. To get `nlssrc` output in English from a subsystem called `ctsubsys`, enter:

```
nlssrc -c -ls ctsubsys
```

2. The following example shows the same information in different formats:

```
nlssrc -ls ctsubsys (locale-dependent)

Subsystem  Group     PID   Status
ctsubsys   ctsubsys  6334  active
2 locally-connected clients.  Their PIDs:
15614 23248
HA Subsystem domain information:
Domain established by node 5
Number of groups known locally: 1
                    Number of          Number of local
Group Name          providers          providers/subscribers
ha_filesys             7                  1        0
```

```
nlssrc -ls ctsubsys -c (canonical form)

Number of local clients: 2
PIDs: 15614 23248
HA Subsystem domain information:
Domain established by node 5.
Number of known local groups: 1
Group Name: ha_filesys
     Providers: 7
     Local Providers: 1
     Local Subscribers: 0
```

Location

/opt/rsct/bin/nlssrc
Contains the `nlssrc` command

Files

/tmp/ctsupt
Location of the default directory that contains the output files.

/tmp/ctsupt/ctsnap.*host_name.nnnnnnnn*.log
Location of the log file of the command execution, where *nnnnnnnn* is a timestamp and *host_name* is the name of the host on which the command is running.

tmp/ctsupt/ctsnap.*host_name.nnnnnnnn*.tar.Z
Location of the compressed tar file that contains the collected data, where *nnnnnnnn* is a timestamp and *host_name* is the name of the host on which the command is running.

nm Command

Purpose

Displays information about symbols in object files, executable files, and object-file libraries.

Syntax

nm [**-A**] [**-C**] [**-X** {**32**|**64**|**32_64**|**d64**| **any**}] [**-f**] [**-h**] [**-l**] [**-p**] [**-r**] [**-T**] [**-v**] [**-B** | **-P**] [**-e** | **-g** | **-u**] [**-d** | **-o** | **-x** | **-t** *Format*] *File* ...

Description

The **nm** command displays information about symbols in the specified *File*, which can be an object file, an executable file, or an object-file library. If the file contains no symbol information, the **nm** command reports the fact, but does not interpret it as an error condition. The **nm** command reports numerical values in decimal notation by default.

The **nm** command writes the following symbol information to standard output:

- **Library** or **Object Name**

  The **nm** command reports either the library or the object name associated with the file only if you specify the **-A** option.
- **Symbol Name**
- **Symbol Type**

  The **nm** command represents the file's symbol type with one of the following characters (with weak symbols represented by the same characters as global symbols):

Item	Description
A	Global absolute symbol.
a	Local absolute symbol.
B	Global bss symbol.
b	Local bss symbol.
D	Global data symbol.
d	Local data symbol.
f	Source file name symbol.
L	Global thread-local symbol (TLS).
l	Static thread-local symbol (TLS).
T	Global text symbol.
t	Local text symbol.
U	Undefined symbol.

- **Value**
- **Size**

  The **nm** command reports the size associated with the symbol, if applicable.

Flags

Item	Description
-A	Displays either the full path name or library name of an object on each line.
-B	Displays output in the Berkeley Software Distribution (BSD) format: `value type name`
-C	Suppresses the demangling of C++ names. The default is to demangle all C++ symbol names. **Note:** Symbols from C++ object files have their names demangled before they are used.
-d	Displays a symbol's value and size as a decimal. This is the default.
-e	Displays only static and external (global) symbols.
-f	Displays full output, including redundant .text, .data, and .bss symbols, which are normally suppressed.
-g	Displays only external (global) symbols.
-h	Suppresses the display of output header data.

Item	Description
-l	Distinguishes between WEAK and GLOBAL symbols by appending a * to the key letter for WEAK symbols. If used with the **-P** option, the symbol type for weak symbols is represented as follows: **V** Weak Data Symbol **W** Weak Text Symbol **w** Weak Undefined Symbol **Z** Weak bss Symbol
-o	Displays a symbol's value and size as an octal rather than a decimal number.
-P	Displays information in a standard portable output format: `library/object name name type value size` This format displays numerical values in hexadecimal notation, unless you specify a different format with the **-t**, **-d**, or **-o** flags. The **-P** flag displays the **library/object name** field only if you specify the **-A** flag. Also, the **-P** flag displays the **size** field only for symbols for which size is applicable.
-p	Does not sort. The ouput is printed in symbol-table order.
-r	Sorts in reverse order.
-t *Format*	Displays numerical values in the specified format, where the *Format* parameter is one of the following notations: **d** Decimal notation. This is the default format for the **nm** command. **o** Octal notation. **x** Hexadecimal notation.
-T	Truncates every name that would otherwise overflow its column, making the last character displayed in the name an asterisk. By default, **nm** displays the entire name of the symbols listed, and a name that is longer than the width of the column set aside for it causes every column after the name to be misaligned.
-u	Displays only undefined symbols.
-v	Sorts output by value instead of alphabetically.
-x	Displays a symbol's value and size as a hexadecimal rather than a decimal number.

Item	Description
-X *mode*	Specifies the type of object file **nm** should examine. The *mode* must be one of the following: **32** Processes only 32-bit object files **64** Processes only 64-bit object files **32_64** Processes both 32-bit and 64-bit object files **d64** Examines discontinued 64-bit XCOFF files (magic number == U803XTOCMAGIC). **any** Processes all of the supported object files. The default is to process 32-bit object files (ignore 64-bit objects). The *mode* can also be set with the **OBJECT_MODE** environment variable. For example, **OBJECT_MODE=64** causes **nm** to process any 64-bit objects and ignore 32-bit objects. The **-X** flag overrides the **OBJECT_MODE** variable.

Note: The **nm** command supports the **—** (double hyphen) flag. This flag distinguishes a *File* operand if the file name can be misinterpreted as an option. For example, to specify a file name that begins with a hyphen, use the **—** flag.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To list the static and external symbols of the object file a`.out`, enter:

```
nm -e a.out
```

2. To display symbol sizes and values as hexadecimal and sort the symbols by value, enter:

```
nm -xv a.out
```

3. To display symbol of all 64-bit objects in **libc.a**, ignoring all 32-bit objects:

```
nm -X64 /usr/lib/libc.a
```

Files

Item	Description
/usr/ccs/bin/nm	Contains the **nm** command.

nmon Command

Purpose

Displays local system statistics in interactive mode and records system statistics in recording mode.

Syntax

Interactive mode

nmon [**-h**]

nmon [**-s** < *seconds* >] [**-c** < *count* >] [**-b**] [**-B**] [**-g** < *filename* >] [**-k** *disklist*] [**-C** < process1:process2:...:processN >] [**-i**]

Recording mode

nmon [**-f** | **-F** *filename* | **-x** | **-X** | **-z**] [**-r** < *runname* >] [**-t** | **-T** | **-Y**] [**-s** *seconds*] [**-c** *number*] [**-w** *number*] [**-l** *dpl*] [**-d**] [**-g** *filename*] [**-k** *disklist*] [**-C** <process1:process2:...:processN >] [**-G**] [**-K**] [**-o** *outputpath*] [**-D**] [**-E**] [**-J**] [**-V**] [**-P**] [**-M**] [**-N**] [**-W**] [**-S**] [**-^**] [**-O**] [**-L**] [**-I** *percent*] [**-A**] [**-m** < *dir* >] [**-Z** *priority*] [**-i**] [**-y** *options*]

Note: In recording mode, specify only one of the **-f**, **-F**, **-z**, **-x**, or **-X** flags as the first argument.

Description

The **nmon** command displays and records local system information. The command can run either in interactive or recording mode. If you specify any of the **-F**, **-f**, **-X**, **-x**, and **-Z** flags, the **nmon** command is in recording mode. Otherwise, the **nmon** command is in interactive mode.

The **nmon** command provides the following views in interactive mode:

- Adapter I/O statistics (pressing the **a** key)
- AIO processes view (pressing the **A** key)
- Detailed Page Statistics (pressing the **M** key)
- Disk busy map (pressing the **o** key)
- Disk groups (pressing the **g** key)
- Disk statistics (pressing the **D** key)
- Disk statistics with graph (pressing the **d** key)
- ESS vpath statistics view (pressing the **e** key)
- Fibre Channel adapter statistics (pressing the **^** key)
- JFS view (pressing the **j** key)
- Kernel statistics (pressing the **k** key)
- Long term processor averages view (pressing the **l** key)
- Large page analysis (pressing the **L** key)
- Memory and paging statistics (pressing the **m** key)
- Network interface view (pressing the **n** key)
- NFS panel (pressing the **N** key)
- Paging space (pressing the **P** key)
- Process view (pressing the **t** and **u** keys)
- Processor usage small view (pressing the **c** key)
- Processor usage large view (pressing the **C** key)
- Shared Ethernet adapter statistics (pressing the **O** key)

- Shared-processor logical partition view (pressing the **p** key)
- System resource view (pressing the **r** key)
- Thread level statistics (pressing the **i** key)
- Verbose checks OK/Warn/Danger view (pressing the **v** key)
- Volume group statistics (pressing the **V** key)
- WLM view (pressing the **W** key)

In the recording mode, the command generates the **nmon** files. You can view these files directly by opening them or with post-processing tools such as nmon analyzer. The **nmon** tool disconnects from the shell during the recording so that the command continues running even if you log out.

If you use the same set of keys every time you start the **nmon** command, you can place the keys in the NMON shell variable. For example, you can run the following command:

```
export NMON=mcd
```

Then run the **nmon** command.

To stop the **nmon** command from the command line, use the **kill -USR2** with the nmon process ID.

To print the background process IDs of the nmon recording, run the **nmon** command with the **-p** flag.

To limit the processes that the **nmon** command lists (online and to a file), you can utilize the following options:

- Set the program names in environment variables from **NMONCMD0** to **NMONCMD63**
- Use the **-C** flag with *cmd:cmd:cmd* parameter. For example, you can enter the following command:

```
nmon -C ksh:vi:syncd
```

To limit the disks that the **nmon** lists to a maximum of 64 (online only), use the **-k** flag with the *diskname* parameter. For example, you can enter the following command:

```
nmon -k hdisk2,hdisk0,hdisk3
```

The **nmon** tool disconnects from the shell during the recording, ensuring that the command continues running even if you log out. This function is not true in the case of recordings triggered using the on-demand recording facility.

Recording or monitoring journaled file system (JFS) statistics in nmon can prevent unloading a file system because the file system is in use while collecting statistics.

Inside workload partitions (WPAR), the **nmon** command shows global values for processors and memory statistics. The rest of the values are WPAR specific. The following statistics cannot be retrieved inside a WPAR, and the **nmon** screen does not support them inside a WPAR:

- Disks, disk I/O graphs, disk busy map, disk groups
- Disk adapters
- Paging space
- Volume group
- ESS/vpaths
- Fibre Channel adapters
- VIOS Shared Ethernet adapters

Note: The dynamic configuration changes applied to the system are not reflected in the current **nmon** recording. The nmon tool must be restarted for the new configuration changes to take effect.

Flags in Interactive Mode

You can use the following flags in the interactive mode.

Item	Description
-s < *seconds* >	Time interval between refreshing the screen. The default value is 2 seconds.
-c < *count* >	Number of times the screen must be refreshed.
-g < *filename* >	A file that contains user-defined disk groups that can be specified using the *filename* parameter. Each line in the file begins with a group name. The list of hard disks follows the group name and is separated by spaces. The file can contain a maximum of 64 disk groups. A hard disk can belong to various disk groups.
-b	Displays the view in black and white mode.
-B	Does not include boxes in the view. By default, the command displays boxes.
-h	Displays help information.
-k < *disklist* >	Reports only the disks in the disk list.
-i	Reports top thread level CPU utilization.

Flags in Recording Mode

Item	Description
-A	Includes the Asynchronous I/O section in the view.
-c	Specifies the number snapshots that must be taken by the command. The default value is 10000000.
-d	Includes the Disk Service Time section in the view.
-D	Skips the Disk Configuration section.
-E	Skips the ESS Configuration section.
-f	Specifies that the output is in spreadsheet format. By default, the command takes 288 snapshots of system data with an interval of 300 seconds between each snapshot. The name of the output file is in the format of *hostname_YYMMDD_HHMM***.nmon**.
-F	Specifies that the output is in spreadsheet format and the name of the output file is *filename*. The *filename* parameter specifies the name of the output file.
-g	Specifies the file that contains the user-defined disk groups, using the *filename* parameter. Each line in the file begins with a group name. The list of disks follows the group name and is separated with spaces. The file can contain a maximum of 64 disk groups. A disk can belong to various disk groups.
-G	Uses Greenwich mean time (GMT) instead of local time. This method is helpful when you compare nmon files from many LPARof 1 system for processor view but the LPARare in different time zones.
-i	Reports thread level statistics.
-I	Specifies the percentage of process threshold at which the command ignores the TOP processes statistics. The default percentage is zero. The command does not save the TOP processes statistics if the process is using less processor power than the specified percentage.
-J	Skips the JFS section.
-k	Specifies a list of disks to be recorded.

Item	Description
-K	Includes the RAW Kernel section and the LPAR section in the recording file. The **-K** flag dumps the raw numbers of the corresponding data structure. The memory dump is readable and can be used when the command is recording the data.
-l	Specifies the number of disks to be listed on each line. By default, 150 disks are listed per line. For EMC disks, specify a value of 64.
-L	Includes the large page analysis section.
-m	Changes the directory before the command saves the data to a file.
-M	Includes the MEMPAGES section in the recording file. The MEMPAGES section displays detailed memory statistics per page size.
-N	Includes the NFS section in the recording file. To collect the NFSv4 statistics, specify **-NN**.
-o	Specifies the file name or directory to which the recorded file is to be stored.
-O	Includes the Shared Ethernet adapter (SEA) VIOS sections in the recording file.
-P	Includes the Paging Space section in the recording file.
-r	Specifies the value for the *runname* field written to the spreadsheet file. By default, the value is the hostname.
-s	Specifies the interval in seconds between 2 consecutive recording snapshots.
-S	Includes WLM sections with subclasses in the recording file.
-t	Includes the top processes in the output. You cannot specify the **-t**, **-T**, or **-Y** flags with each other.
-T	Includes the top processes in the output and saves the command-line arguments into the UARG section. You cannot specify the **-t**, **-T**, or **-Y** flags with each other.
-V	Includes disk volume group section.
-w	Specifies the size of timestamp (Tnnnn) to be recorded. The timestamp is recorded in the **.csv** file. The value of the *number* parameter ranges from 4 through 16. For NMON analyzer, use the values 4 or 8.
-W	Includes the WLM sections into the recording file.
-x	Specifies the sensible spreadsheet recording for duration of 1 day for capacity planning. By default, the recording is done every 900 seconds for 96 times. This flag is equivalent to **-ft -s 900 -c 96**.
-X	Specifies the sensible spreadsheet recording for duration of 1 hour for capacity planning. By default, the recording is done every 30 seconds for 120 times. This flag is equivalent to **-ft -s 30 -c 120**.

Item	Description
-y options	Controls the nmon recording sections. The values of the **options** parameter must be separated by commas. The following values are valid for the **options** parameter: **sub=sea** Records the SEA Bridged adapters statistics. **sub=ssp** Records the shared storage pool (SSP) statistics. **PCPU=[on | off]** Enables or disables recording of Physical CPU (PCPU) sections, which are nothing but metrics that start with PCPU. These metrics are based on Processor Utilization of Resources Register (PURR). The default value is off. You can specify the following values for the PCPU section: **on** Enables recording of the PCPU section. **off** Disables recording of the PCPU section. **SCPU=[on | off]** Enables or disables recording of Scaled CPU (SCPU) sections, which are metrics that start with SCPU. These metrics are based on Scaled Processor Utilization of Resources Register (SPURR). Default value is off. You can specify the following values for the SCPU section: **on** Enables recording of the SCPU section. **off** Disables recording of the SCPU section. **dfreq=[on | off]** Enables or disables frequency sections, which are metrics that start with CPUMHZ. These metrics are based on the POWER options that are set. Default value is off. You can specify the following values for the dfreq section: **on** Enables recording of the CPUMHZ section. **off** Disables recording of the CPUMHZ section. **Note:** The value reported in the currentMHz column in CPUMHZ can vary slightly from the actual processor speed depending on conditions such as system load. The latest values of the **options** parameter override the previous values if the same value is used more than once in the command line. **Example:** If you run the command "`nmon -f -y PCPU=on -y PCPU=off`", the value *off* is used for the PCPU option.
-Y	Includes the top process in the recording with all of the commands of the same name added and recorded. You cannot specify the **-t**, **-T**, or **-Y** flags together.
-z	Specifies the sensible spreadsheet recording for duration of 1day for capacity planning. By default, the recording is done every 900 seconds for 96 times. This flag is equivalent to **-f -s 900 -c 96**.

Item	Description
-Z	Specifies the priority of the **nmon** command that is running. A value of -20 means important. A value of 20 means not important. Only root user can specify negative value.
-^	Includes the Fibre Channel (FC) sections.

Parameters

Item	Description
disklist	Specifies a list of disks.
dir	Specifies a directory.
dpl	Specifies the number of disks to list on each line.
filename	Specifies a file that contains the disk group you select.
number	Specifies the number of refreshes.
count	Specified the number of times to record.
percent	Specifies the percentage of processor usage.
priority	Specifies the priority of processes to be run.
runname	Specifies the value for the *runname* field in the spreadsheet file to be run.
seconds	Specifies the interval, in seconds, of refreshing the snapshot.
outputpath	Specifies the path for the output file.

Subcommands

Item	Description
space	Refreshes the screen immediately.
.	Displays only busy disks and processes.
~	Switches to the **topas** screen.
^	Displays the Fibre Channel adapter statistics
+	Doubles the screen refresh time.
-	Decreases the screen refresh time by half.
0	Resets the peak values of statistics (displayed on the screen) to zero. Applicable only for panels that display peak values.
a	Displays the I/O statistics of the adapters.
A	Summarizes the Async I/O (AIO server) processes.
b	Displays the view in black and white mode.
c	Displays processor statistics with bar graphs.
C	Displays processor statistics. It is useful for comparison when the number of processors ranges from 15 to 128.
d	Displays the I/O information of disks. To display specific disks only, specify the **-k** flag.
D	Displays the I/O statistics of disks. To get additional statistics of the disks, press the **D** key more than once.

Item	Description
e	Displays the I/O statistics of the ESS virtual path logical disks.
g	Displays the I/O statistics of the Disk Group. You must specify the **-g** flag with this key.
h	Displays the online help information.
j	Displays the JFS statistics.
k	Displays the internal statistics of the kernel.
l	Displays the processor statistics in long format. More than 75 snapshots are displayed with bar graphs.
m	Displays the memory and paging statistics.
M	Displays multiple page size statistics in pages. If you press the **M** key twice, the statistics are displayed in megabytes.
n	Displays the network statistics.
N	Displays the statistics of the NFS Network file system. If you press the **N** key twice, you see the NFSv4 statistics.
o	Displays the map of Disk I/O.
O	Displays only the Shared Ethernet adapter VIOS.
p	Displays the statistics of the partitions.
P	Displays the statistics of the paging space.
q	Quits. You can also use the **x**, or Ctrl+C key sequence.
r	Displays the resource type, system name, cache details, AIX version, and the LPAR information.
S	Displays the WLM with subclasses.
t	Displays the statistics of top processes. You can press the following keys with this subcommand: • 1: Displays basic details. • 2: Displays accumulated process information. • 3: Sorts the view by processor. • 4: Sorts the view by size. • 5: Sorts the view by I/O information.
u	Displays the top processes with the command arguments. To refresh the arguments for new processes, press the **u** key twice.
U	Displays the top processes with the command arguments, and the workload class or workload partitionworkload partition information.
v	Highlights status of pre-defined system resources and categorizes them as either danger, warnings, or normal.
V	Displays the statistics of the Disk Volume Group.
w	Displays the wait processes when used with the top processes.
W	Displays the statistics of the Workload Manager (WLM).
[	Triggers a custom on-demand recording. The recording initiated exits along with the interactive **nmon** if not stopped earlier.
]	Stops a custom recording triggered by **]** .

Output Details

This section provides explanations to the metrics that are displayed on nmon screen.

System resources view

This view provides general information about the system resources. To display this view, press the **r** key. It contains information about the following resources:

- The number of processors in the system.
- The number of online processors that are active in the system.
- The frequency of the processors.
- The version of AIX and its technical level.
- The type of the running kernel.
- The logical partition.
- The power savings mode of the logical partition.
- The model of the hardware.
- The processor architecture of the system.
- The type of the platform bus.
- The cache information of processors.
- The number of active events.
- The old serial number. This number is the system ID of the partition before the dynamic configuration event.
- The current serial number. This number is the current system ID or the system ID of the partition after the dynamic configuration event.
- The local time of the last dynamic reconfiguration event. This information is labeled with the "When" keyword.
- The sub processor mode of the logical partition.

AIO Processes View

The AIO processes view provides information about the asynchronous I/O (AIO) processes. To display this view, press the **A** key. The following columns are displayed on the screen:

Item	Description
Total AIO Processes	The total number of AIO processes.
Actually in use	The number of AIO processes that uses more than 0.1% of the processor.
CPU Used	The percentage of the processor that is used by all of the kernel processes.
All time peak	The maximum number of kernel processes that are running since the system starts.
Recent peak	The recent maximum number of kernel processes that use more than 0.1% of the processor.
Peak	The maximum percentage of the processor that is used by all of the kernel processes.

Process View

The **Process View** provides details of the processes in the system. To display this view, press the **t** key or the **v** key. It contains the following columns are displayed on the screen:

Item	Description
pid	The ID of the process.

Item	Description
ppid	The ID of the parent process.
User	The user ID of the process.
Proc Group	The ID of the process group.
Nice	The initial priority of a process. This value is set by the **nice** command.
Priority	The base schedule priority of a process.
Status	The status of a program.
Proc_Flag	The flag of a process.
Thrds	The number of threads.
Files	The maximum file index that is in use.
Foreground	Foreground process or background process.
Command	The name of the command.
Time Start	The time when the command started.
CPU-Total	The total time that the process takes since it starts.
Child Total	The total time that the child process takes since it starts.
Delta-Total	The total time taken by the process in the interval.
%CPU Used	The percentage of the processor that is used in the last interval.
Size KB	The size of the pages in kilobytes.
Res Size	The sum of real-memory data (resident set) and real-memory (resident set) text size of the process.
Res Set	The sum of real-memory data (resident set) and real-memory (resident set) text size of the process.
Res Text	The real-memory text size of the process.
Res Data	The real-memory data size of the process.
Char I/O	The number of I/O characters per second from the last interval.
RAM Use	The percentage of the RAM that is used.
Paging I/O	The I/O page faults per second in the last interval.
Paging Other	The non-I/O page faults per second in the last interval.
Paging Repages	The number of repage faults per second in the last interval.
Class	The Workload Manager class name of the process.

Processor Usage Small View

The Processor Usage Small View provides a brief summary of the user, system, idle, and wait time of logical processors, the corresponding entitlement, and the virtual processor used. You can generate the Processor Usage Small View by pressing the **c** key.

Processor Usage Large View

The Processor Usage Large View displays the use of logical processor in a graph. To display this view, you can press the **C** key.

The following labels are used to identify time that is spent in different modes:

- **s**: Labels the percentage of time that is spent in system mode
- **u**: Labels the percentage of time that is spent in user mode

Shared-Processor Logical Partition View

The Shared-Processor Logical Partition View includes flags that indicate the following information of a partition:

- Whether the partition is an LPAR or not
- Whether the partition can be an LPAR or not
- Whether the partition is shared or dedicated
- Whether the SMT is turned on or off
- Whether the shared-partition is capped or uncapped
- Whether LPAR is SMT bound or enabled
- Whether the LPAR flags are set, and whether they are set to display a value greater than AVG=lp

If the flags are set, the **nmon+C** graph contains information about the Cpu_user and the Avg_user, respectively. You can view the graph in the right column.

To display this view, you can press the **p** key.

Processors:

The following metrics of the processor status are displayed in this view:

Item	Description
Max Phys in Sys	Maximum number of physical processors in the system
Phys CPU in system	Number of physical processors in the system
Virtual Online	Number of online virtual processors
Logical online	Number of online logical processors
Physical pool	Number of shared physical processors in the shared pool ID that this partition is assigned to
SMT threads/CPU	Number of SMT threads per processor

Capacity:

The following information displays the processor capacity:

Item	Description
Cap. Processor Min	Minimum number of processing units that are defined for this LPAR
Cap. Processor Max	Maximum number of processing units that are defined for this LPAR
Cap. Increment	Granularity at which changes to the entitled capacity can be made
Cap. Unallocated	Sum of the number of processor units that are unallocated from shared LPARin an LPAR group
Cap. Entitled	Entitled capacity
MinReqVirtualCPU	Minimum required virtual processors for the LPAR

ID Memory:

The following metrics of the ID memory are displayed:

Item	Description
LPAR ID Group:Pool	ID of an LPAR group and its pool ID
Memory (MB/GB) Min:Max	Minimum and maximum memory that is defined for this LPAR in megabytes or gigabytes
Memory(MB/GB) Online	Online real memory in megabytes or gigabytes

Item	Description
Memory Region LMB	Size in bytes of one logical memory block (LMB)

Time (in seconds):

Item	Description
Time Dispatch Wheel	Interval during which each virtual processor receives its entitlement
MaxDispatch Latency	Maximum latency in seconds between the dispatch of the LPAR on the physical processors
Time Pool Idle	Time in seconds that the shared processor pool is idle
Time Total Dispatch	Total time in seconds that the LPAR dispatches

Minimum and Maximum Values of Processors

The following minimum and maximum values of processors are displayed:

Item	Description
Virtual CPU (Min - Max)	Minimum number and maximum number of virtual processors in the LPAR definition
Logical CPU (Min - Max)	Minimum number and maximum number of logical processors

Weight

The following information about the weight of the processor is displayed:

Item	Description
Weight Variable	Variable weight of the processor capacity
Weight Unallocated	Unallocated variable weight available for this partition

NFS Panel

The NFS Panel provides information about the Network File System (NFS). To display this view, press the **N** key. The following metrics are included in the view:

Item	Description
Root	NFS V2 server and client root requests
Wrcache	NFS server and client write cache requests
Null	NFS server and client write cache requests
Getattr	NFS server and client get attributes requests
Setattr	NFS server and client set attributes requests
Lookup	NFS server and client filename lookup requests
Readlink	NFS server and client read link requests
Read	NFS server and client read requests
Write	NFS server and client write requests
Create	NFS server and client file creation requests
Mkdir	NFS server and client directory creation requests
Symlink	NFS server and client symbolic link creation requests
Remove	NFS server and client file removal requests

Item	Description
Rmdir	NFS server and client directory removal requests
Rename	NFS server and client file renaming requests
Link	NFS server and client link creation requests
Readdir	NFS server and client read-directory requests
Fsstat	NFS server and client file-status requests
Access	NFS V3 server and client access requests
Mknod	NFS V3 server and client **mknod** creation requests
readdir+	NFS V3 server and client read-directory plus requests
Fsinfo	NFS V3 server and client file information requests
Pathconf	NFS V3 server and client path configuration requests
Commit	NFS server and client commit requests
Bad calls	NFS server and client failed calls
Calls	NFS server and client requests

The following NFS V4 client/server statistics are printed when you press the **N** key twice.

Item	Description
Access	NFS V4 server and client access requests
acl_read	NFS V4 client reading access control list (ACL)
acl_stat_l	NFS V4 client that is retrieving long ACL information
acl_write	NFS V4 client write access control list (ACL)
Clntconfirm	NFS V4 client confirm operations
Close	NFS V4 client closing files
Commit	NFS V4 server and client committed
Compound	NFS V4 server compound calls
Create	NFS V4 server and client that is creating a non-regular object
Delegpurge	NFS V4 server purge delegations that is awaiting recovery
Delegreturn	NFS V4 server and client that is returning delegation
Finfo	NFS V4 client that is obtaining file information
getattr	NFS V4 server and client retrieving attributes
getfh	NFS V4 server retrieving file handles
Link	NFS V4 server and client that is linking operations
Lock	NFS V4 server and client that is locking operations
lockt/test	NFS V4 server that is testing the specified lock or NFS V4 client lock test
locku/unlock	NFS V4 server or NFS V4 client unlock operations
lookup	NFS V4 server and client that is looking up filenames
lookupp	NFS V4 server that is looking up parent directories
mkdir	NFS V4 client that is creating directories
mknod	NFS V4 client that is creating special files

Item	Description
Null	NFS V4 server null calls or NFS V4 client null calls
nverify	NFS V4 server verifying difference in attributes
openattr	NFS V4 server opening named attribute directories
openconfirm	NFS V4 server and client that is confirming the open for usage
opendowngrade	NFS V4 server and client that is downgrading the access for a specified file
Open	NFS V4 server and client open operations
operations	NFS V4 server and client operations
pcl_read	NFS V4 client extracting numeric data from printer control language (PCL) files
pcl_readstat_l	NFS V4 client **pcl_stat** long operations
pcl_stat	NFS V4 client **pcl_stat** operations
pcl_write	NFS V4 client **pcl_write** operations
putfh	NFS V4 server setting current file handles
putpubfh	NFS V4 server setting public file handles
putrootfh	NFS V4 server setting root file handles
readdir	NFS V4 server and client reading directories
readlink	NFS V4 server and client reading symbolic links
Read	NFS V4 server and client reading data from files
release	NFS V4 server and client **release_lock** operations
remove	NFS V4 server and client removing file system object
rename	NFS V4 server and client renaming object names
renew	NFS V4 server and client renewing leases
replicate	NFS V4 client replicate operations
restorefh	NFS V4 server restoring file handles
rmdir	NFS V4 client removing directories
savefh	NFS V4 server saving file handles
secinfo	NFS V4 server and client obtaining security information
setattr	NFS V4 server and client setting object attributes
setclient	NFS V4 server and client **setclient** operations
statfs	NFS V4 client file statistics requests
symlink	NFS V4 client symbolic link operations
verify	NFS V4 client verifying same attributes
write	NFS V4 server and client writing to files

Network Interface View

The Network Interface View shows the statistics errors for the network. You can view this information by pressing the **n** key.

If the screen is updated three times with no network errors, the Network Interface View does not contain the network error statistics.

The following metrics are displayed in this view:

Item	Description
I/F Name	Interface name
Recv-KB/s	Data that are received in kilobytes per second in the interval
Trans-KB/s	Data that are transmitted in kilobytes per second in the interval
Packin	Number of packets that are received in the interval
Packout	Number of packets that are sent in the interval
Insize	Average size of packet that is received in the interval
Outsize	Average size of packet that is sent in last interval
Peak->Recv	Peak value of received data in kilobytes per second
Peak->Trans	Peak value of sent data in kilobytes per second
Total Recv	Total received data in megabytes per second
Total Sent	Total sent data in megabytes per second
MTU	Maximum size of transport unit in bytes
Ierror	Number of input errors
Oerror	Number of output errors
Collision	Number of collisions
Mbits/s	Adapter bit rate in megabits (Mbits) per second. If the network adapter is larger than 10Gb, the adapter bit rate is shown as 10240 Mbits per second.
Description	Description of the interface

WLM View

The WLM View shows the information about workload management. You can open this view by pressing the **W** key. To turn on the subclasses section, press the **S** key from WLM View. To turn off the subclasses section, press the **S** key again.

The following metrics are displayed in this view:

Item	Description
CPU	Percentage of processor use of the class.
MEM	Percentage of physical memory use of the class.
BIO	Percentage of disk I/O bandwidth use for the class.
Process (Procs)	Number of processes in the class.
Tier (T)	Tier number. The value ranges from zero through nine.
Inheritance (I)	Values of the inheritance attribute. A value of zero means no. A value of one means yes.
Location	Values of location. A value of one means avoiding transfer of segments to shared classes. Otherwise, a value of zero is displayed.

Disk Busy Map

The Disk Busy Map shows the use statistics of disks. To display this map, press the **o** key. A maximum of 100 disks is shown per screen. Only the disks with the names that range from hdisk0 through hdisk100 are displayed. The following table shows the symbols for the ranges of names.

Symbols	Names
_	Less than 5

Symbols	Names
.	Less than 10
-	Less than 20
+	Less than 30
o	Less than 40
0	Less than 50
O	Less than 60
8	Less than 70
X	Less than 80
#	Less than 90
@	Less than 100 and equal to 100

Disk Groups

Multiple disks can be monitored by placing them in groups. To display this view, press the **g** key.

You must create a group configuration file that contains the lines as shown in the following example:

```
<Group_name1> <disk_name1> <disk_name2> ....
<Group_name2> <disk_nameA> <disk_nameB> ...
```

In the example, <Group_name1> is the name of the first disk in the group; <disk_name1> and <disk_name2> are the first and second disks in the group.

To see the Disk Group I/O, run the **nmon** command with the **-g** flag and a group file, and then press the **g** key. The following metrics are shown in this view:

Item	Description
Name	Disk Group name. You can specify a maximum of 64 groups. A disk can be in multiple groups.
Disks	Number of disks in the group.
Read/Write-KB/s	Data transfer rate of read and written data in kilobytes per second in the interval.
TotalMB/s	Sum of read and written data in megabytes per second in the interval.
Xfers/s	Number of read and written data transfers per second in the interval.
BlockSizeKB	Block size in kilobytes read or written per transfer operation.

ESS Vpath Statistics View

This view provides the ESS Vpath Statistics. To display this view, press the **e** key. The following metrics are included in this view:

Item	Description
Name	Name of the virtual path.
Size	Size of the ESS path.
AvgBusy	Average busy use of the disk.
Write-KB/s	Transfer rate of written data in kilobytes per second in the interval.
Read-KB/s	Transfer rate of read data in kilobytes per second in the interval.
Xfers/s	Number of read and write transfers per second.

Item	Description
Total vpaths	Number of virtual paths.

JFS View

This view provides the Journaled File System (JFS) statistics. To display this view, press the **j** key. The following statistics are recorded in this view:

Item	Description
FileSystem	Name of the file system.
Size (MB)	Size in megabytes for the file system.
Free (MB)	Available free space in megabytes in the file system.
%Used	Percent of file system used.
%Inodes	Percent of file system that is used by i-nodes.
Mount point	Local mount point.

Kernel Statistics

This view contains the statistics of the kernel. To display this view, press the **k** key. The following statistics are displayed in this view:

Item	Description
runqueue	Average number of threads that are ready to run but are waiting for an available processor.
pswitch	Number of processor switches per second in the interval.
fork	Number of forks per second in the interval.
exec	Number of execs per second in the interval.
msg	Number of interprocess communication (IPC) messages that are sent and received per second in the interval.
sem	Number of semaphore operation system calls per second in the interval.
hw intrp	Number of device interrupts per second in the interval.
sw intrp	Number of off-level handlers that are called per second in the interval.
Swapin	Number of processes in swap queue per second in the interval.
Syscall	Number of system calls per second in the interval.
read	Number of read calls per second in the interval.
write	Number of write calls per second in the interval.
readch	Number of characters that are transferred through read system call per second in the interval.
Writech	Number of characters that are transferred through write system call per second in the interval.
R + W (MB/s)	Number of read and write characters in megabytes per second in the interval.
Uptime	Time duration for which the system is up.
iget	Number of inode lookups per second in the interval.
dirblk	Number of 512-byte block reads by the directory search routine to locate an entry for a file per second in the interval.
namei	Number of vnode lookup from a path name per second in the interval.

Item	Description
ksched	Number of kernel processes that are created per second in the interval.
koverf	Number of kernel process creation attempts where the user forked to the maximum limit or the configuration limit of processes that are reached per second in the interval.
kexit	Number of kernel processes that become zombies per second in the interval.

Long Term Processor Averages View

This view provides information about the instantaneous system. To display this view, press the **l** key. You can use the following labels to identify the time that is spent in different modes:

- **s**: Labels the percentage of the time that is spent in system mode.
- **u**: Labels the percentage of the time that is spent in user mode.
- **w**: Labels the percentage of the time that is spent in wait mode.

The following metrics are displayed on this view:

Item	Description
EntitledCPU	Entitled capacity of the partition.
UsedCPU	Number of physical processors that are used by the partition.

Large Page Analysis

This view provides analysis of the large page. To display this view, press the **L** key. The following information is displayed:

Item	Description
Count	Number of large pages and their total size.
Free	Percentage of free large pages and their size.
In Use	Percentage of large pages in use and their size.
Size	Size of a large page.
High water mark	Large page high watermark.

Paging Space

This view prints the paging-space statistics. To display this view, press the **p** key. The following metrics are displayed in the view:

Item	Description
PagingSpace	Number of paging spaces.
Volume-Group	Number of volume groups.
Type	Type of logical volumes. The types can be NFS or LV.
LPs	Size of logical partitions.
MB	Size in megabytes.
Used	Percentage of use for volume groups.
IOpending	Number of pending I/O in the paging space.
Active/Inactive	Active or inactive paging space.
Auto/NotAuto	Indicates whether the paging space is auto that is loaded or not.

Volume Group Statistics

This view provides statistics for the volume group. To display this view, press the **V** key. The following information is displayed in the view:

Item	Description
Name	Volume group name.
Disks	Number of disks in the group.
AvgBusy	Average busy of the disks in the volume group.
Read/Write-KB/s	Data transfer rate of read and written data in kilobytes per second in the interval.
TotalMB/s	Sum of read and written data in megabytes per second in the interval.
Xfers/s	Number of read and written transfers per second in the interval.
BlockSizeKB	Block size that is read or written per transfer in kilobytes per second in the interval.

Disk Statistics

This view provides statistics for disks. To display this view, press the **D** key. You can press the **D** key for the following times to view various metrics:

- Once: Shows disk numbers
- Twice: Shows disk descriptions
- Three times: Shows service times
- Four times: Shows disk statistics with graphs similar to the graph shown on pressing the **d** key

Disk Numbers (pressing the **D** key once)

The following metrics are shown in this view:

Item	Description
Name	Name of the disks.
Busy	Average busy of the disks.
Read-KB/s	Data transfer rate of read data in kilobytes per second in the interval.
Write-KB/s	Data transfer rate of written data in kilobytes per second in the interval.
Transfers/sec	Number of read and written transfer per second in the interval.
SizeKB	Block size that is read or written per transfer in kilobytes per second in the interval.
Peak	Peak percentage of average busy.
Peak KB/s	Peak that is read and written data in kilobytes per second.
qDepth	Number of requests that are sent to disk and are not completed.
Totals Size (GB)	Total size of disks in gigabytes.
Totals Free (GB)	Total free space that is left in disks in gigabytes.
Totals Read (MB/s)	Total data transfer rate of read data from all disks in megabytes per second.
Totals Write (MB/s)	Total data transfer rate of written data to all disks in megabytes per second.

Disk Descriptions (Pressing the **D** key twice)

The following metrics are shown in this view:

Item	Description
Name	Disk names.
Size (GB)	Size of disks in gigabytes.
Free (GB)	Free space that is left in disk in gigabytes.
Disk Paths	Number of paths that are defined to the disk.
Disk Adapter	Name of disk adapters.
Volume Group	Volume group that the disk belongs to.
Disk Description	Description of the disk.
Totals Size (GB)	Total size of disks in gigabytes.
Totals Free (GB)	Total free space that is left in disks in gigabytes.
Totals Read (MB/s)	Total data transfer rate of read data from all disks in megabytes per second.
Totals Write (MB/s)	Total data transfer rate of written data to all disks in megabytes per second.

Service Times (Pressing the **D** key three times)

The following metrics are displayed in the view:

Item	Description
Disk	Name of the disk.
Service (in msecs)	Average service time per request in milliseconds.
Wait (in msecs)	Average waiting time per request in milliseconds.
ServQ size	Average number of requests in service queue.
WaitQ size	Average number of requests that is waiting to be accomplished.
ServQ Full	Number of times the disk is not accepting any coming requests.
Totals Size (GB)	Total size of disks in gigabytes.
Totals Free (GB)	Total free space that is left in disks in gigabytes.
Totals Read (MB/s)	Total data transfer rate of read data from all disks in megabytes per second.
Totals Write (MB/s)	Total data transfer rate of written data to all disks in megabytes per second.

Disk Statistics With Graphs (Pressing the **D** key four times)

This view displays disk statistics with graphs. To display this view, press the **d** key. The following metrics are displayed in this view:

Item	Description
Name	Name of the disk.
Busy	Average percentage of busy for the disk.
Read-KB/s	Data transfer rate of read data in kilobytes per second.
Write-KB/s	Data transfer rate of written data in kilobytes per second.

Memory and Paging Statistics

The view provides information about the memory and paging statistics. To display this view, press the **m** key. The following metrics are included in this view:

Item	Description
%Used	Percentage of used space in physical memory and paging space.
%Free	Percentage of free space in physical memory and paging space.
MB Used	Physical memory and paging space that are used in megabytes.
MB Free	Physical memory and paging space that are free in megabytes.
Pages/sec to Paging Space	Number of I/O pages that are transferred to or from the paging space per second.
Pages/sec to file system	Number of I/O pages that are transferred to or from the file system per second.
Page Scans	Number of page scans by clock.
Page Faults	Number of page faults.
Page Cycles	Number of page replacement cycles.
Page Steals	Number of pages steals.
Numperm	Number of frames that are used for files (in 4-KB pages).
Process	Percentage of real memory that is used by process segments.
System	Percentage of real memory that is used by system segments.
Free	Percentage of real memory that is free.
Total	Percentage of total real memory used.
Min/Maxperm	The **minperm** and **maxperm** values for page steals.
Min/Maxfree	The **minfree** and **maxfree** pages free list.
Min/Maxpgahead	Minimum and maximum number of page ahead pages.
Total Virtual	Total virtual memory.
Accessed Virtual	Active virtual memory.
Numclient	Number of client frames.
Maxclient	Maximum number of client frames.
User	Real memory that is used by non-system segments.
Pinned	Real memory that is pinned.

The AMS statistics are displayed in the **topas_nmon** memory panel. To display this view, press the **m** key. The following metrics are included in this view:

Item	Description
Pool	AMS pool ID of the pool that the logical partition (LPAR) belongs to.
Weight	Weight of the variable memory.
pMem	Physical memory currently backing up the logical memory partition (in MB).
hpi	Number of hypervisor page-ins.
hpit	Time that is spent in hypervisor page-ins (in seconds).

Logical unit information:

Item	Description
Size (MB)	Total size that is allocated for the logical unit.

Item	Description
Lu Udid	Logical unit identifier.

Adapter I/O Statistics View

This view provides the adapter I/O statistics. To display this view, press the **a** key. The following metrics are displayed in this view:

Item	Description
Adapter	Name of the adapter.
Busy%	Bandwidth use of the adapter. This is the aggregate **Busy%** of the disks connected to this adapter. The value might exceed 100% if more than one disk is connected to the adapter.
Read-KB/s	Data transfer rate of read data in kilobytes per second.
Write-KB/s	Data transfer rate of written data in kilobytes per second.
Transfers	Number of read and write transfers.
Disks	Number of disks.
Adapter-Type	Type of the adapter.

Shared Ethernet adapter

This view provides shared Ethernet adapter statistics in a Virtual I/O Server (VIOS). To display this view, press the **O** key. The following metrics are displayed in this view:

Item	Description
Number	Serial number.
Name	Name of the shared Ethernet adapter.
Recv-KB/s	Data transfer rate of received data in kilobytes per second.
Trans-KB/s	Data transfer rate of sent data in kilobytes per second.
Packin	Number of packets that are received per second in the interval.
Packout	Number of packets that are sent per second in the interval.
Insize	Average size per second for received packet in the interval.
Outsize	Average size per second for outgoing packet in the interval.

Verbose Checks OK/Warn/Danger

This view prints the statistics for processor, memory, and disks. It also prints the status message, such as OK, Warn, or Danger, which is based on the system metrics that exceed the pre-defined threshold values. To display this view, press the **v** key.

Detailed Page Statistics

This view provides page statistics. To display this view, press the **M** key.

If you press the **M** key once, the view contains the statistics in pages. If you press the **M** key twice, the page statistics are shown in megabytes.

The following metrics are shown in this view:

Item	Description
Numframes	Number of real memory frames of the page size.
Numfrb	Number of pages on free list.

Item	Description
Numclient	Number of client frames.
Numcompress	Number of frames in compressed segments.
Numperm	Number of frames in non-working segments.
Numvpages	Number of accessed virtual pages.
Minfree	Minimum free list.
Maxfree	Maximum free list.
Numpout	Number of page-outs.
Numremote	Number of remote page-outs.
Numwseguse	Number of pages in use for working segments.
Numpseguse	Number of pages in use for persistent segments.
Numclseguse	Number of pages in use for client segments.
Numwsegpin	Number of pages that are pinned for working segments.
Numpsegpin	Number of pages that are pinned for persistent segments.
Numclsegpin	Number of pages that are pinned for client segments.
numpgsp_pgs	Number of allocated page spaces.
numralloc	Number of remote allocations.
pfrsvdblks	Number of system reserved blocks.
Pfavail	Number of pages available for pinning.
Pfpinavail	Application level number pages available for pinning.
system_pgs	Number of pages on segment control blocks (SCB) that are marked with **V_SYSTEM.**
nonsys_pgs	Number of pages on SCBs not marked with **V_SYSTEM.**
Numpermio	Number of pageouts in non-working storage.
Pgexct	Number of page faults.
Pgrclm	Number of pages reclaims.
Pageins	Number of paged-in pages.
Pageouts	Number of paged-out pages.
Pgspgins	Number of paged-in pages from page space.
Pgspgouts	Number of paged-out pages from page space.
Numsios	Number of I/O started.
Numiodone	Number of I/O completed.
Zerofills	Number of zero-filled pages.
Exfills	Number of exec-filled pages.
Scans	Number of page scans by clock.
Cycles	Number of clock hand cycles.
pgsteals	Number of pages steals.

Fibre Channel Adapter Statistics

This view contains information about the Fibre Channel adapter. You can see this view by pressing the caret (**-^**) key. The following metrics are included in this view:

Note: If the N_Port Virtualization (NPIV) is configured on the VIOS, use the -^ option in the **nmon** command to record the NPIV related statistics.

Item	Description
Number	Serial number.
Name	Name of the Fibre Channel adapter.
Receive-KB/s	Data transfer rate of received data in kilobytes per second.
Transmit-KB/s	Data transfer rate of sent data in kilobytes per second.
Requests In	Number of requests that are received per second in the interval.
Requests Out	Number of requests that are sent per second in the interval.
Outsize	Average outgoing packet size per second in the interval.

Thread level statistics

This view contains information about thread level statistics. To display this view, press the **-i** key. The following metrics are included in this view:

Item	Description
PID	Process ID to which the thread belongs.
TID	Top thread ID that uses higher CPU. Sorting is based on CPU utilization in descending order.
%CPU	Percentage of CPU used by the specific thread.
BOUND CPU ID	Bounded CPU ID if the thread has been bound to any processor.

Environment Variables

Environment variables **NMON_START**, **NMON_END**, **NMON_SNAP**, and **NMON_ONE_IN** are used for collecting external data while recording in nmon format.

Item	Description
NMONCMD0, NMONCMD1, ..., NMONCMD63	You can monitor only the processes that are set in these variables when these environment variables are set. Alternatively, you can use the **-C** flag to restrict the commands in the process listing of the **nmon** command. For example, you can run the **nmon -C db2:nmon:topas** command.
NMON	Contains the set of key strokes corresponding to the initial set of panels to be displayed when the **nmon** command is started.

Item	Description
NMON_TIMESTAMP	You can specify the **NMON_TIMESTAMP** variable to the following values: **NMON_TIMESTAMP = 0** The recorded lines contain the nmon Tnnnn timestamps at the beginning of the line and work with the nmon data file. **NMON_TIMESTAMP = 1** The lines contains timestamps that have the hours, minute, seconds, day, month, and year. This value can be used if you do not want to merge the data with the nmon file for analysis.
NMON_START	External command to be started when the **nmon** recording begins.
NMON_END	External command to be started when the **nmon** recording ends.
NMON_SNAP	External command to be started periodically to record metrics.
NMON_ONE_IN	You can specify the **NMON_ONE_IN** variable to the following values: **NMON_ONE_IN=1** Runs the **snap** command every time the recording is done. **NMON_ONE_IN=*n*** Runs the **snap** command after the number of recordings that are specified by the *n* parameter is done.

Examples

1. To generate the **nmon** recording in the current directory for two hours, capturing data every 30 seconds, enter the following command:

   ```
   nmon -f -s 30 -c 240
   ```

2. To display the memory and processor statistics immediately after the **nmon** command is started, do the following steps:

   a. Enter the following command:

   ```
   export NMON=mc
   ```

   b. Run the **nmon** command.

3. To run the **nmon** command for 20 seconds with the screen that is refreshed at 10 seconds, enter the following command:

   ```
   nmon -c 10 -s 2
   ```

4. To run nmon in black and white mode, enter the following command:

   ```
   nmon -b
   ```

5. To view the process information, do the following steps:

   a. Run the **nmon** command.

b. Press the **t** key.

6. To view the list of views that **nmon** command provides, press the key **h**.
7. The following sample explains the steps to collect external data. In the sample, the `mystart` file, the `mysnap` file, and the `myend` file are executable and are in the path that the $PATH defines.

   a. Set the environment variables as indicated in the following example:

   ```
   $export NMON_TIMESTAMP=0
   $export NMON_START="mystart"
   $export NMON_SNAP="mysnap"
   $export NMON_END="myend"
   $export NMON_ONE_IN=1
   ```

   In the previous example, the value of one is the default value for the NMON_ONE_IN environment variable. It generates one set of external recorded data for every snapshot of nmon recording.

   b. Modify the content of the `mystart` file as the following:

   ```
   ps -ef >start_ps.xt
   echo "PROCCOUNT,Process Count, Procs" >ps.csv
   ```

   c. Modify the content of the `mysnap` file as the following:

   ```
   echo PROCCOUNT,$1,`ps -ef | wc -l` >>ps.csv
   ```

   d. Modify the content of the `myend` file as the following:

   ```
   echo PROCCOUNT,$1,`ps -ef | wc -l` >>ps.csv
   ```

   e. Run the **nmon** command as follows:

   ```
   nmon -f -s 2 -c 10
   ```

   The recording finishes in 20 seconds.

   The output of the `ps.csv` file is similar to the following sample:

   ```
   PROCCOUNT,Process Count, Procs
   PROCCOUNT,T0001, 43
   PROCCOUNT,T0002, 43
   PROCCOUNT,T0003, 43
   PROCCOUNT,T0004, 43
   PROCCOUNT,T0005, 43
   PROCCOUNT,T0006, 43
   PROCCOUNT,T0007, 43
   PROCCOUNT,T0008, 43
   PROCCOUNT,T0009, 44
   PROCCOUNT,T0010, 44
   PROCCOUNT,T0010, 44
   ```

   To concatenate the generated nmon file with the `ps.csv` file that is generated by external recording, enter the following command:

   ```
   cat  filename.nmon ps.csv > c.csv
   ```

   To get the graph, open the `c.csv` file in **nmon** analyzer.

8. To view the hdisk details, enter the **nmon** command with `-k` flag:

   ```
   nmon -k hdisk1,hdisk2
   ```

   The previous command shows the disk details for hdisk1 and hdisk2. For hdiskpower devices, enter the following command:

   ```
   nmon -k hdiskpower or
   nmon -k power
   ```

   Note: The `nmon -k hdisk` matches all the hdisk devices on the LPAR and does not match the hdiskpower devices.

All hdiskpower devices display as power in interactive and recording modes. For example, `nmon -k hdiskpower1` matches the device hdiskpower1 and `nmon -k hdiskpower` matches all hdiskpower devices on the LPAR.

Note: The output of the **lsconf** and **lspv** commands in the **nmon** recording file is not affected by the changes to the **nmon -k** command.

Location

`/usr/bin/nmon`

`/usr/bin/topasrec`

no Command

Purpose

Manages the tuning parameters of the network.

Syntax

no [**-p** | **-r**] { **-o** *Tunable*[**=***NewValue*] }

no [**-p** | **-r**] {**-d** *Tunable* }

no [**-p** | **-r**] { **-D** }

no [**-p** | **-r**] [**-F**] **-a**

no -h [*Tunable*]

no [**-F**] **-L** [*Tunable*]

no [**-F**] **-x** [*Tunable*]

Note: Multiple flags **-o**, **-d**, **-x**, and **-L** are allowed.

Description

Use the **no** command to configure parameters that used to tune the network. The **no** command sets or displays current or next system boot values for network tuning parameters. This command can also make permanent changes or defer changes until the next system reboot. Whether the command sets or displays a parameter, is determined by the accompanying flag. The **-o** flag does both these actions. It can either display the value of a parameter or set a new value for a parameter. When the **no** command is used to modify a network option, it logs a message to the syslog by using the LOG_KERN facility.

Understanding the Effect of Changing Tunable Parameters

Be careful when you use this command. If used incorrectly, the **no** command can cause your system to become inoperable.

Before you modify any tunable parameter, you must read about all its characteristics in the Tunable Parameters section, and follow the Refer To pointer instructions to understand the purpose. Ensure that the Diagnosis and Tuning sections for this parameter apply to the situation, and changing the value of this parameter helps to improve the performance of your system.

If the Diagnosis and Tuning sections both contain N/A, you must not change this parameter unless directed by AIX development.

Flags

Item	Description
-a	Displays current, reboot (when used with **-r**) or permanent (when used with **-p**) value for all tunable parameters, one per line in pairs *Tunable* **=** *Value*. For the permanent options, a value displays for a parameter if its reboot and current values are equal. Otherwise NONE displays as the value.
-d *Tunable*	Resets *Tunable* to its default value. If *Tunable* must be changed when, it is set to one of the following values: • The tunable is not set to its default value and it is of type Bosboot or reboot • The tunable is of type Incremental and must be changed from its default value. and **-r** is not used in combination. The tunable parameter is not changed but a warning message is displayed.
-D	Resets all tunable parameters to their default value. If a tunable parameter that must be changed, is of one of the following types: • Bosboot or Reboot • Incremental type and is changed from its default value and if either **-p** nor **-r** flag are used in combination, the parameter is not changed but a warning message is displayed.
-F	Forces restricted tunable parameters to be displayed when the options **-a**, **-L** or **-x** are specified on the command line. If you do not specify the **-F** flag, restricted tunables are not included, unless they are named in association with a display option.
-h [*Tunable*]	Displays help about *Tunable* parameter if one is specified. Otherwise, displays the **no** command usage statement.

Item	Description
-L [*Tunable*]	Lists the characteristics of one or all *Tunable*s, one per line, by using the following format:

```
NAME              CUR    DEF    BOOT   MIN
MAX    UNIT       TYPE
 DEPENDENCIES
------------------------------------------------------
-----------------
General Network
Parameters
------------------------------------------------------
-----------------
sockthresh        85     85     85     0
100    %_of_thewall  D
------------------------------------------------------
-----------------
fasttimo          200    200    200    50
200    millisecond   D
------------------------------------------------------
-----------------
inet_stack_size   16     16     16
1             kbyte         R
------------------------------------------------------
-----------------
...
where:
    CUR = current value
    DEF = default value
    BOOT = reboot value
    MIN = minimal value
    MAX = maximum value
    UNIT = tunable unit of measure
    TYPE = parameter type: D (for Dynamic),
           S (for Static), R (for Reboot),B (for
Bosboot), M (for Mount),
           I (for Incremental), C (for Connect),
and d (for Deprecated)
    DEPENDENCIES = list of dependent tunable
parameters, one per line
```

-o *Tunable* [**=***NewValue*]	Displays the value or sets the *Tunable* to *NewValue*. If a tunable must be changed, that is the specified value is different from current value, and is one of the following types: • Bosboot or Reboot • Incremental and its current value is more than the specified value and **-r** is not used in combination, it is not changed but a warning message is displayed. When **-r** is used in combination without a new value, the nextboot value for *Tunable* is displayed. When **-p** is used in combination without a new value, a value displays only if the current and next boot values for tunable are the same Otherwise NONE displays as the value.

Item	Description
-p	Changes are applied to both current and reboot values when used in combination with **-o**, **-d** or **-D**, that is turns on updating of the /etc/tunables/nextboot file in addition to updating of the current value. These combinations cannot be used on Reboot and Bosboot type parameters because their current value cannot be changed. When used with **-a** or **-o** without specifying a new value, the values are displayed when the current and next boot values for a parameter are the same. Otherwise NONE displays as the value.
-r	Changes are applied to reboot values when used in combination with **-o**, **-d**, or **-D** flags, that is it turns on updating the /etc/tunables/nextboot file. If any parameter of type Bosboot is changed, the user is prompted to run bosboot. When used with **-a** or **-o** without specifying a new value, next boot values for tunables display instead of the current values.
-x [*Tunable*]	Lists characteristics of one or all tunables, one per line, by using the following (spreadsheet) format: `tunable,current,default,reboot,min,max,unit,type,` `{dtunable }` `where:` `  current = current value` `  default = default value` `  reboot = reboot value` `  min = minimal value` `  max = maximum value` `  unit = tunable unit of measure` `  TYPE = parameter type: D (for Dynamic),` `  S (for Static), R (for Reboot),B` `(for Bosboot), M (for Mount),` `  I (for Incremental), C (for Connect),` `and d (for Deprecated)` `  dtunable = space separated list of` `dependent tunable parameters`

If you change by using the **-o**, **-d** or **-D** flag to a restricted tunable parameter, it results in a warning message that a tunable parameter of the restricted-use type is modified. If you also specify the **-r** or **-p** options on the command line, you are prompted for confirmation of the change. During system reboot, the presence of restricted tunables in the /etc/tunables/nextboot file that were modified to a value different from their default value by using a command line and by specifying the **-r** or **-p** options, results in an error log entry that identifies the list of these modified tunables.

If you change by using the **-o**, **-d**, or **-D** flag to a parameter of type Mount, it results in a warning message that the change is effective for future mountings.

If you change to a parameter of type Connect by using the **-o**, **-d** or **-D** flag, it results in starting the **inetd** and displays a warning message that the change is effective for future socket connections.

If you change to a parameter of type Bosboot or Reboot by using the **-o**, **-d**, or **-D** flag and without using the **-r** flag, it results in an error message.

If you change the current value of a parameter of type Incremental with a new value that is smaller than the current value by using the **-o**, **-d**, or **-D** flag and without using the **-r** flag, it results in an error message.

Tunable Parameters Type

All the tunable parameters that are manipulated by the tuning commands such as **no**, **nfso**, **vmo**, **ioo**, **schedo**, and **raso** commands are classified into the following categories:

Item	Description
Dynamic	If the parameter can be changed at any time
Static	If the parameter can never be changed
Reboot	If the parameter can be changed during reboot
Bosboot	If the parameter can be changed by running bosboot and rebooting the machine
Mount	If changes to the parameter are only effective for future file systems or directory mounts
Incremental	If the parameter can be incremented, except at boot time
Connect	If changes to the parameter are only effective for future socket connections
Deprecated	If this parameter cannot be changed and is no longer supported by the current release of AIX.

For parameters of type Bosboot, whenever there is a change, the tuning commands automatically prompt the user to ask if they want to run the **bosboot** command. For parameters of type Connect, the tuning commands automatically restart the **inetd** daemon if pre520tune is disabled.

Note: The current set of parameters that are managed by the **no** command includes Reboot, Static, Dynamic, Incremental, and Connect types.

Tunable Parameters

For default values and range of values for tunables, refer the **no** command help (**-h** <*tunable_parameter_name*>).

Item	Description
arpqsize	**Purpose:** Specifies the maximum number of packets to queue while waiting for Address Resolution Protocol (ARP) responses. **Tuning:** This attribute is supported by Ethernet, 802.3, Token Ring and FDDI interfaces.
arpt_killc	**Purpose:** Specifies the time in minutes before a complete ARP entry will be deleted. **Tuning:** To reduce ARP activity in a stable network, you can increase **arpt_killc**.
arptab_bsiz	**Purpose:** Specifies Address Resolution Protocol (ARP) table bucket size. **Tuning:** **netstat -p arp** will show the number of ARP packets sent and the number of ARP entries purged from the ARP table. If large number of entries are being purged, the ARP table size should be increased. Use arp -a to show the ARP table hashing distribution.

Item	Description
arptab_nb	**Purpose:** Specifies the number of ARP table buckets. **Tuning:** **netstat -p arp** will show the number of ARP packets sent and the number of ARP entries purged from the ARP table. If large number of entries are being purged, the ARP table size should be increased. Use arp -a to show the ARP table hashing distribution. Increase this value for systems that have a large number of clients or servers. The default provides for 149 x 7 = 1043 ARP entries, but assumes an even hash distribution.
bcastping	**Purpose:** Allows response to ICMP echo packets to the broadcast address. **Tuning:** A value of **0** disables it; while a value on **1** enables it. The default is to not respond to echo packets to a broadcast address. This prevents so called 'broadcast storms' on the network that can result when multiple machines respond to a broadcast address.
clean_partial_conns	**Purpose:** Specifies whether or not we are avoiding SYN attacks. If non-zero, **clean_partial_conns** specifies how many partial connections to be removed randomly to make room for new non-attack connections. **Tuning:** A value of **0** disables this option. This option should be turned on for servers that need to protect against network attacks.
delayack	**Purpose:** Delays ACKs for certain TCP packets and attempts to piggyback them with the next packet sent instead. **Tuning:** This action will only be performed for connections whose destination port is specified in the list of the **delayackports** attribute. This can be used to increase the performance when communicating with an HTTP server by reducing the total number of packets sent. The parameter can have one of following four values: **0** No delays, normal operation **1** Delays the ACK for the server's SYN **2** Delays the ACK for the server's FIN **3** Delay both the ACKs for the SYN and FIN

Item	Description
delayackports	**Purpose:** Specifies the list of destination ports for which the operation defined by the `delayack` port option is performed. **Tuning:** The attribute takes a maximum of 10 ports, which are separated by commas and enclosed in curly braces. For example: `no -o delayackports={80,30080}.` To clear the list, set the option to {}.
dgd_flush_cached_route	**Purpose:** Flushes the cached routes of sockets when Dead Gateway Detection detects a previous dead gateway back online. The connections are forced to reacquire the route before the data is sent. **Tuning:** A value of 1 enables the DGD to flush the cached routes. A value of 0 disables it.
dgd_packets_lost	**Purpose:** Specifies how many consecutive packets must be lost before Dead Gateway Detection decides that a gateway is down.
dgd_ping_time	**Purpose:** Specifies the seconds that must pass between pings of a gateway by Active Dead Gateway Detection.
dgd_retry_time	**Purpose:** Specifies the minutes a route's cost must remain raised when it is raised by Passive Dead Gateway Detection. After this many minutes pass, the route's cost is restored to its user-configured value. The unit specified is in numeric.
directed_broadcast	**Purpose:** Specifies whether a directed broadcast to a gateway must be allowed or not. **Tuning:** The value of 1 allows packets to be directed to a gateway that must be broadcast on a network on the other side of the gateway.
fasttimo	**Purpose:** Allows to set the millisecond delay for the TCP fast timeout timer. This timeout controls how often the system scans the TCP control blocks to send delayed acknowledgments. **Tuning:** Reducing this timer value can improve performance with some non-IBM systems. However, this parameter can result in slightly increased system utilization.

Item	Description
hstcp	**Purpose:** Enables the HighSpeed TCP as specified in RFC 3649. This parameter modifies the congestion control mechanism for use with TCP connections with large congestion windows to improve the average throughput. **Tuning:** A value of 1 enables the HighSpeed TCP enhancements on a system-wide scale. A value of 0 disables it.
icmp6_errmsg_rate	**Purpose:** Specifies the upper limit for the number of ICMP v6 error messages that can be sent per second. This parameter prevents excessive bandwidth from being used by ICMP v6 error messages.
icmpaddressmask	**Purpose:** Specifies whether the system responds to an ICMP address mask request. **Tuning:** If the value 0 is set, the network silently ignores any ICMP address mask request that it receives.
icmptimestamp	**Purpose:** Specifies whether the system responds to an ICMP timestamp request. **Tuning:** If the value of 0 is set, the network ignores any ICMP timestamp request that it receives.
ie5_old_multicast_mapping	**Purpose:** Specifies IP multicasts on token ring that must be mapped to the broadcast address rather than a functional address when value 1 is used.
ifsize	**Purpose:** Specifies the maximum number of network interface structures per interface of a single type. This limit does not apply to ethernet interface structures for which the infrastructure expands dynamically to handle any number of ethernet interface structures. **Tuning:** The **ifsize** parameter must be large on systems that supports hotplug adapters and on DLPAR configurations because adapters can be added as required. The static interface tables must be large enough to accept the large number of adapters that is added for this system or partition. If the system detects at the start, that more adapters of a type are present than that is allowed by the current value of **ifsize**, it automatically increases the value to support the number of adapters present.
ip6_defttl	**Purpose:** Specifies the default hop count that is used for IP version 6 packets if no other hop count is specified.
ip6_prune	**Purpose:** Specifies how often to check the IP version 6 routing table for expired routes, in seconds.

Item	Description
ip6forwarding	**Purpose:** Specifies whether the kernel must forward the IP version 6 packets. **Tuning:** The default value of 0 prevents forwarding of ipv6 packets when they are not for the local systems. A value of 1 enables forwarding.
ip6srcrouteforward	**Purpose:** Specifies whether the system forwards source-routed IP version 6 packets. **Tuning:** A value of 1 allows the forwarding of source-routed packets. A value of 0 causes all source-routed packets that are not at their destinations to be discarded.
ip_ifdelete_notify	**Purpose:** Specifies when an interface address is deleted. All the existing TCP connections that were bound locally to the interface address and were deleted must be notified with error **ENETDOWN**. **Tuning:** Existing FTP/Telnet connections are disconnected when the ENETDOWN error is returned.
ip_ifdelete_no_retrans	**Purpose:** Specifies that when an interface address is deleted, the existing TCP connections that were bound locally to the interface address must not retransmit data. **Tuning** No further retransmission of data occurs over the existing SSH connections.
ip_nfrag	**Purpose:** Specifies the maximum number of fragments of an IP packet that can be kept on IP reassembly queue at a time.
ipforwarding	**Purpose:** Specifies whether the kernel must forward packets. **Tuning:** Set this parameter to 1, if the system is acting as an IP router.
ipfragttl	**Purpose:** Specifies the time to live for IP fragments in half-seconds. **Tuning:** Check for fragments that dropped after timeout (netstat -p ip). If the value of IP, that is the fragments dropped after timeout is nonzero, increases the **ipfragttl** parameter, it can reduce retransmissions.
ipignoreredirects	**Purpose:** Specifies whether to process redirects that are received. **Tuning:** A value of 0 processes redirects as usual. A value of 1 ignores redirects.

Item	Description
ipqmaxlen	**Purpose:** Specifies the number of received packets that can be queued on the IP protocol input queue. **Tuning:** Examine if `ipintrq` overflows (netstat -s) or use crash to access IP input queue overflow counter. Increase size if system is using many loopback sessions. Most operating system network drivers call IP directly and do not use the IP queue. Increasing the **ipqmaxlen** parameter on these devices has no effect.
ipoutqueues	**Purpose** Specifies whether to queue User Datagram Protocol (UDP) packets that are sent over IPv4. These UDP packets are handled by a separate kernel thread. **Tunning** The default value is 0 and it specifies the UDP to transmit the packet immediately without queuing. A non-zero value specifies the number of queues to be created and used. For example, to create a single queue that is used by the UDP, enter the following command: `no -o ipoutqueues=1`
ipsendredirects	**Purpose:** Specifies whether the kernel must send redirect signals. **Tuning:** This parameter is a configuration decision with performance consequences.
ipsrcrouteforward	**Purpose:** Specifies whether the system forwards source routed packets. **Tuning:** The default value of 1 allows the forwarding of source-routed packets. A value of 0 causes all source-routed packets that are not at their destinations to be discarded.
ipsrcrouterecv	**Purpose:** Specifies whether the system accepts source routed packets. **Tuning:** The default value of 0 causes all source-routed packets that are destined for this system to be discarded. A value of 1 allows source-routed packets to be received.
ipsrcroutesend	**Purpose:** Specifies whether applications can send source routed packets. **Tuning:** The default value of 1 allows source-routed packets to be sent. A value of 0 causes **setsockopt()** to return an error if an application attempts to set the source routing option, and removes any source routing options from the outgoing packets.

Item	Description
limited_ss	**Purpose:** Enables the Limited SlowStart as specified in RFC 3742. This limits the number of segments by which the congestion window is increased for one window during slow-start. This enhancement improves the performance for TCP connections with large congestion windows. **Tuning:** A value from 1 to 100 enables the Limited SlowStart enhancements on a system-wide scale and sets it as the number of segments to the value of the maximum SlowStart threshold. A value of 0 disables it. The default value is 0.
llsleep_timeout	**Purpose:** Specifies timeout value in seconds for link local timeouts (used when multi_homed=1).
lo_perf	**Purpose:** Specifies whether you want to utilize a separate queue per CPU to improve loopback performance. **Tuning:** A value of 1 enables a separate queue per CPU. A value of 0 disables this option.
lowthresh	**Purpose:** Specifies the maximum number of bytes that can be allocated by using the **allocb** call for the **BPRI_LO** priority. **Tuning:** When the total amount of memory that is allocated by the **net_malloc** call reaches this threshold, then the **allocb** request for the **BPRI_LO** priority returns 0. The lowthresh attribute represents a percentage of the **thewall** attribute and you can set its value from 0 to 100.
main_if6	**Purpose:** Specifies the interface to use for link local addresses.
main_site6	**Purpose:** Specifies the interface to use for site local address routing.
maxnip6q	**Purpose:** Specifies the maximum number of IP version 6 packet reassembly queues.
maxttl	**Purpose:** Specifies the time to live (in seconds) for RIP packets.
medthresh	**Purpose:** Specifies the maximum number of bytes that can be allocated by using the **allocb** call for the **BPRI_MED** priority. **Tuning:** When the total amount of memory that is allocated by the **net_malloc** call reaches this threshold, then the **allocb** request for the **BPRI_MED** priority returns 0. The **medthresh** attribute represents a percentage of the **thewall** attribute. A typical setting of 95 represents 95% of **thewall** attribute.

Item	Description
mpr_policy	**Purpose:** Specifies the policy to be used for Multipath Routing. **Tuning:** The following are the available routing policies: **Weighted Round-Robin (1)** Based on user-configured weights that are assigned to the multiple routes (through the route command) round-robin is applied. If no weights are configured then, it behaves identical to plain round-robin. **Random (2)** Chooses a route at random. **Weighted Random (3)** Chooses a route that is based on user-configured weights and a randomization routine. The policy adds up the weights of all the routes and picks a random number between 0 and total weight. Each of the individual weights is removed from the total weight until this number is zero. This picks a route in the range of the total number of routes available. **Lowest Utilization (4)** Chooses a route with the minimum number of current connections going through it. **Hash-based (5)** Hash-based algorithm chooses a route by hashing based on the destination IP address.
multi_homed	**Purpose:** Specifies the level of multi-homed IP version 6 host support. **Tuning:** Tuning is performed for connections whose destination port is specified in the list of the `delayackports` parameter. This parameter can be used to increase performance when communicating with an HTTP server. The parameter can have one of four values: **0** Indicates the original functionality in AIX 4.3. **1** Indicates that link local addresses is resolved by querying each interface for the link local address. **2** Indicates that link local addresses is examined for the interface that is defined by `main_if6`. **3** Indicates that link local addresses is examined for the interface that is defined by `main_if6` and site local addresses are routed to the `main_site6` interface.

Item	Description
nbc_limit	**Purpose:** Specifies the total maximum amount of memory that can be used for the Network Buffer Cache. **Tuning:** This attribute is in number of Kilobytes. When the cache grows to this limit, the rarely used cache objects are flushed out of the cache to make room for the new ones.
nbc_max_cache	**Purpose:** Specifies the maximum size of the cache object that is allowed in the Network Buffer Cache without using the private segments. **Tuning:** This parameter is in number of bytes. A data object bigger than this size is either cached in a private segment or is not cached at all.
nbc_min_cache	**Purpose:** Specifies the minimum size of the cache object that is allowed in the Network Buffer Cache. **Tuning:** This attribute is in number of bytes. A data object smaller than this size is not put into the NBC. This attribute applies for **send_file()** API and some web servers that use the get engine in the kernel.
nbc_ofile_hashsz	**Purpose:** Specifies the size of the hash table that is used for hashing cache objects in the Network Buffer Cache. **Tuning:** This hash table size applies to only opened file entries that is, entries that cache files from the file system. Since this attribute resizes the hash table size and affects the hashing of all existing entries, this attribute can be modified when the Network Buffer Cache is empty.
nbc_pseg	**Purpose:** Specifies the maximum number of private segments that can be created for the Network Buffer Cache. **Tuning:** When this option is set at nonzero0, a data object between the size that is specified in **nbc_max_cache** and the segment size (256MB) is cached in a private segment. A data object bigger than the segment size is not cached. When the maximum number of private segments exist, cache data in private segments can be flushed for new cache data so that the number of private segments do not exceed the limit. When **nbc_pseg** is set to 0, all cache in private segments is flushed.

Item	Description
nbc_pseg_limit	**Purpose:** Specifies the maximum amount of cached data size allowed in private segments in the Network Buffer Cache. **Tuning:** This value is expressed in Kilobytes. Since data cached in private segments are pinned by the Network Buffer Cache, **nbc_pseg_limit** controls the amount of pinned memory that is used for the Network Buffer Cache in addition to the network buffers in global segments. When the amount of cached data reaches this limit, cache data in private segments can be flushed for new cache data so that the total pinned memory size does not exceed the limit. When **nbc_pseg_limit** is set to 0, all cache in private segments is flushed.
ndd_event_name	**Purpose:** Specifies the list of interface names for **ns_alloc** and **ns_free** events to be captured, when the trace of **ns_alloc/ns_free** events is enabled by setting the `ndd_event_tracing` option.
ndd_event_tracing	**Purpose:** Specifies the size of the `ns_alloc/ns_free` trace buffer. **Tuning:** If the value of this option is non-zero all **ns_alloc/ns_free** events are traced in a kernel buffer. A value of zero disables this event tracing. If the values of `ndd_event_tracing` are larger than 1024 it allocates as many items in the kernel buffer for tracing.
ndp_mmaxtries	**Purpose:** Specifies the maximum number of Multicast NDP Neighbor Discovery Protocol (NDP) packets to send.
ndp_umaxtries	**Purpose:** Specifies the maximum number of Unicast Neighbor Discovery Protocol (NDP) packets to send.
ndpqsize	**Purpose:** Specifies the number of packets to hold waiting on completion of a Neighbor Discovery Protocol (NDP) entry that is used by IP version 6.
ndpt_down	**Purpose:** Specifies the time, in half seconds, to hold down an NDP entry.
ndpt_keep	**Purpose:** Specifies the time, in half seconds, to keep a Neighbor Discovery Protocol (NDP) entry.
ndpt_probe	**Purpose:** Specifies the time in half seconds, to delay before the first Neighbor Discovery Protocol (NDP) probe is sent .
ndpt_reachable	**Purpose:** Specifies the time, in half seconds, to test if a Neighbor Discovery Protocol (NDP) entry is still valid.
ndpt_retrans	**Purpose:** Specifies the time, in half seconds, to wait before an NDP request is retransmitted.

Item	Description
net_buf_size	**Purpose:** Specifies a list of buffer sizes for `net_malloc/net_free` events to be captured. **Tuning:** The **net_buf_size** strings represent a list of sizes. If this attribute is not of value all, only `net_malloc/net_free` events of those sizes are captured. A value of all means that the events of any size are captured.
net_buf_type	**Purpose:** Specifies a list of buffer types for `net_malloc/net_free` events to be captured. **Tuning:** The **net_buf_type** string represents a list of types. If the string is not empty and different from all, only `net_malloc/net_free` events of that type is captured.
net_malloc_frag_mask	**Purpose:** It is used as boolean attribute for mask with each bucket that requests similar fragments to be promoted to full pages. **Tuning:** Allows promotion of allocations smaller than 1 page to full pages for better detection of memory overwrite problems. It is a mask for each bucket size that requests such fragments to be promoted to full pages. Enabling this option for memory fragments results in lower performance.
netm_page_promote	**Purpose:** Specifies whether to allow promotion of a fragment to page size. **Tuning:** This option allows promotion of fragment sizes that are specified in `net_malloc_frag_mask` to page size. Setting this option to 0, disables the page promotion irrespective of the sizes that are set in `net_malloc_frag_mask`.
nonlocsrcroute	**Purpose:** Tells the Internet Protocol that strictly source-routed packets can be addressed to hosts outside the local network. **Tuning:** A value of 0 disallows addressing to outside hosts. A value of 1 allows packets to be addressed to outside hosts. Loosely source routed packets are not affected by this attribute.
nstrpush	**Purpose:** Specifies the maximum number of modules that you can push onto a single stream. The minimum value is 8. **Tuning:** This parameter is read-only. This attribute can be set when loading the operating system in the `/etc/pse_tune.conf` file.

Item	Description
passive_dgd	**Purpose:** Specifies whether Passive Dead Gateway Detection is enabled. **Tuning:** A value of 0 disables **passive_dgd**, and a value of 1 enables it for all gateways in use.
pmtu_default_age	**Purpose:** This option is now unused because UDP applications are now required to always set **IP_DONTFRAG** socket option to be able to detect decreases in Path MTU. **Tuning:** A value of zero allows no aging. The default value is 10 minutes. The **pmtu_default_age** value can be overridden by UDP applications. **pmtu_default_age** is a runtime attribute.
pmtu_expire	**Purpose:** Specifies the default amount of time (in minutes) before which the path MTU entries with reference count of zero are deleted. **Tuning:** A value of 0 suggests that the **pmtu** entries does not expire.
pmtu_rediscover_interval	**Purpose:** Specifies the default amount of time (in minutes) before the path MTU value for UDP and TCP paths are checked for a higher value. **Tuning:** A value of 0 allows no path MTU rediscovery.
psebufcalls	**Purpose:** Specifies the maximum number of **bufcalls** to allocate by Streams. **Tuning:** The Stream subsystem allocates certain number of **bufcall** structures at initialization, so that when the **allocb** call fails, the user can register their requests for the **bufcall**. You are not allowed to lower this value until the system is restarted. During restart, the parameter returns to its default value.
psecache	**Purpose:** Controls the number of stream buffers.
psetimers	**Purpose:** Specifies the maximum number of timers to allocate by Streams. **Tuning:** The Stream subsystem allocates certain a number of timer structures at initialization so that the streams driver or module can register their timeout calls. You are not allowed to lower this value until the system is restarted. During restart, the parameter returns to its default value.
rfc1122addrchk	**Purpose:** Performs address validation as specified by RFC1122, Requirements for Internet Hosts-Communication Layers. **Tuning:** A value of 0 does not perform address validation. A value of 1 performs address validation.

Item	Description
rfc1323	**Purpose:** Enables TCP enhancements as specified by RFC 1323, TCP Extensions for High Performance. **Tuning:** A value of 0 disables the RFC enhancements on a system-wide scale. A value of 1 specifies that all TCP connections attempts to negotiate the RFC enhancements. The SOCKETS application can override the default behavior on individual TCP connections, by using the **setsockopt** subroutine. The **rfc1323** network option can also be set on a per interface basis through the **ifconfig** command.
rfc2414	**Purpose:** Enables the increasing of TCP's initial window as described in RFC 2414. **Tuning:** When it is on, the initial window depends on setting the **tcp_init_window** tunable.
route_expire	**Purpose:** Specifies whether the route expires. **Tuning:** A value of 0 allows no route expiration. Negative values are not allowed for this option.
routerevalidate	**Purpose:** Specifies that each cached route of a connection must be validated when a new route is added to the routing table. **Tuning:** This option ensures that applications that keep the same connection open for long periods of time (for example NFS) uses the correct route after routing table changes occur. A value of 0 does not revalidate the cached routes. Turning on this option can cause some performance degradation.
rto_high	**Purpose:** Specifies the TCP Retransmit Time out high value that is used in calculating factors, and the allowable maximum retransmits that is used in TCP data segment retransmits. **Tuning:** **rto_high** is the high factor.
rto_length	**Purpose:** Specifies the TCP Retransmit Time Out length value that is used in calculating factors, and the allowable maximum retransmits that is used in TCP data segment retransmits. **Tuning:** **rto_length** is the total number of time segments.
rto_limit	**Purpose:** Specifies the TCP Retransmit Time out limit value that is used in calculating factors, and the allowable maximum retransmits that is used in TCP data segment retransmits. **Tuning:** **rto_limit** is the number of time segments from **rto_low** to **rto_high**.

Item	Description
rto_low	**Purpose:** Specifies the TCP Retransmit Time Out low value that is used in calculating factors, and the allowable maximum retransmits that is used in TCP data segment retransmits. **Tuning:** **rto_low** is the low factor.
sack	**Purpose:** Enables TCP Selective Acknowledgment as described in RFC 2018. **Tuning:** A value of 1 makes all TCP connections negotiate sack. Default is zero, which disables the negotiation. sack feature needs support from the peer TCP. The negotiation phase during connection initiation determines that. When out of order segments are received , Selective Acknowledgments from the receiver informs the sender of the data that is received so that the sender can retransmit only the missing segments. This results in less unnecessary retransmitted segments. Sack is useful for recovering fast from multiple packet drops in a window of data.
sb_max	**Purpose:** Specifies the maximum buffer size that is allowed for a TCP and UDP socket. Limits **setsockopt**, **udp_sendspace**, **udp_recvspace**, **tcp_sendspace**, and **tcp_recvspace**. **Tuning:** Increase size, preferably to multiple of 4096. Must be approximately two to four times the largest socket buffer limit.
send_file_duration	**Purpose:** Specifies the cache validation duration for all the file objects that system call send_file accessed in the network buffer cache. **Tuning:** This attribute is in number of seconds. A value of 0 means that the cache is validated for every access.
site6_index	**Purpose:** Specifies the maximum interface number for site local routing.
sockthresh	**Purpose:** Specifies the maximum amount of network memory that can be allocated for sockets. Used to prevent new sockets or TCP connections from exhausting all MBUF memory and reserve the remaining memory for the existing sockets or TCP connections. **Tuning:** When the total amount of memory that is allocated by the **net_malloc** subroutine reaches the **sockthresh** threshold, the socket and **socketpair** system calls fail with an error of ENOBUFS. Incoming connection requests are silently discarded. Existing sockets can continue to use more memory. The **sockthresh** attribute represents a percentage of the **thewall** attribute.
sodebug	**Purpose:** Specifies whether the newly created sockets has SO_DEBUG flag on.

Item	Description
sodebug_env	**Purpose:** Specifies whether SODEBUG process environment variable is checked for the newly created sockets; if so, these sockets has the SO_DEBUG flag on.
somaxconn	**Purpose:** Specifies the maximum listen backlog. **Tuning:** Increase this parameter on busy web servers to handle peak connection rates.
soreuseport_lb	**Purpose:** Specifies whether the SO_REUSEPORT socket option is enabled or disabled for load balancing. **Tuning:** This tunable parameter can have the following values: • **1** - Enables the SO_REUSEPORT socket option. • **0** - Disables the SO_REUSEPORT socket option.
strctlsz	**Purpose:** Specifies the maximum number of bytes of information that a single system call can pass to a Stream to place into the control part of a message (in an **M_PROTO** or **M_PCPROTO** block). **Tuning:** The **putmsg** call with a control part that exceeds this size fails with **ERANGE**.
strmsgsz	**Purpose:** Specifies the maximum number of bytes of information that a single system call can pass to a Stream to place into the data part of a message (in **M_DATA** blocks). **Tuning:** Any write call that exceeds this size is broken into multiple messages. The **putmsg** call with a data part that exceeds this size fails with **ERANGE**.
strthresh	**Purpose:** Specifies the maximum number of bytes Streams are normally allowed to allocate. **Tuning:** When the threshold is passed, **strthresh** does not allow users without the appropriate privilege to open Streams, push modules, or write to Stream devices, and returns ENOSR. The threshold applies to the output and does not affect the data coming into the system (for example, console continues to work properly). A value of zero means that there is no threshold. The **strthresh** attribute represents a percentage of the **thewall** attribute. The **thewall** attribute indicates the maximum number of bytes that can be allocated by Streams and Sockets by using the **net_malloc** call.

Item	Description
strturncnt	**Purpose:** Specifies the maximum number of requests that are handled by the current running thread for Module or Elsewhere level Streams synchronization. **Tuning:** The Module level synchronization works in a way that only one thread can run in the module at any time and all other threads, which try to acquire the same module enqueues their requests and leave. After the current running thread completes its work, it dequeues all the previously enqueued requests one by one and runs them. If there are many requests that are enqueued in the list, then the current running thread has to serve everyone and will always be busy serving others and starves itself. To avoid this situation, the current running thread serves only the **strturncnt** number of threads after that a separate kernel thread activates and runs all the pending requests.
subnetsarelocal	**Purpose:** Specifies whether all subnets that match the subnet mask are to be considered local for purposes of establishing, for example, the TCP maximum segment size. **Tuning:** This parameter is used by the **in_localaddress** subroutine. The default value, **1** specifies that addresses that match the local network mask are local. If the value is 0, addresses that match the local subnetwork are local. This is a configuration decision with performance consequences. If all the subnets do not have the same MTU, fragmentation at bridges can degrade performance. If the subnets do have the same MTU, and **subnetsarelocal** is 0, TCP sessions can use a small MSS.
tcp_bad_port_limit	**Purpose:** Specifies the number of TCP segments to a port, which does not have a socket connection, within the time duration of half a second. TCP stops sending TCP reset segments in response after this time. **Tuning:** If the value is set to 0, TCP indicates a bad port number error by sending TCP reset segments. A value greater than 0 indicates the number of TCP segments received by a port, which does not have a socket connection, within the time duration of half a second before TCP stops sending TCP reset segments.
tcp_cwnd_modified	**Purpose:** Allows the TCP IP applications with specific socket options to adjust the network congestion window. This parameter might be used only in a specific wide area network (WAN) environment. **Tuning:** Default value is 0, which disables the tuning parameter. Tuning it to a value of 1 allows to adjust the network congestion window.

Item	Description
tcp_ecn	**Purpose:** Enables TCP level support for Explicit Congestion Notification as described in RFC 2481. **Tuning:** Default is off (0). Turning it on (1) makes all connections negotiate ECN capability with the peer. For this feature to work, you need support from the peer TCP and also IP level ECN support from the routers in the path.
tcp_ephemeral_high	**Purpose:** Specifies the largest port number to allocate for TCP ephemeral ports. **Tuning:** The number of ephemeral sockets is determined by **tcp_ephemeral_high** minus **tcp_ephemeral_low**. For maximum number of ephemeral sockets, set **tcp_ephemeral_high** to 65535 and **tcp_ephemeral_low** to 1024.
tcp_ephemeral_low	**Purpose:** Specifies the smallest port number to allocate for TCP ephemeral ports. **Tuning:** The number of ephemeral sockets is determined by **tcp_ephemeral_high** minus **tcp_ephemeral_low**. For maximum number of ephemeral sockets, set **tcp_ephemeral_high** to 65535 and **tcp_ephemeral_low** to 1024.
tcp_fastlo	**Purpose:** Allows the TCP loopback traffic to cutoff the entire TCP/IP stack protocol and interface to achieve better performance. **Tuning:** A value of 1 enables the TCP loopback traffic to cutoff the entire TCP/IP stack. A value of 0 disables this option.
tcp_finwait2	**Purpose:** Specifies the length of time to wait in the FIN_WAIT2 state before closing the connection, measured in half seconds.
tcp_icmpsecure	**Purpose:** Specifies whether or not ICMP (Internet Control Message Protocol) attacks on TCP are avoided. **Tuning:** This option should be turned on to protect TCP connections against ICMP attacks. The ICMP attacks may be of the form of ICMP source quench attacks and PMTUD (Path MTU Discovery) attacks. If this network option is turned on, the system does not react to ICMP source quench messages. This will protect against ICMP source quench attacks. Also, if this network option is enabled, the payload of the ICMP message is tested to determine if the sequence number of the TCP header portion of the payload is within the range of acceptable sequence numbers. This will mitigate PMTUD attacks to a large extent.

Item	Description
tcp_init_window	**Purpose:** This value is used only when rfc2414 is turned on (ignored otherwise). **Tuning:** If rfc2414 is on and this value is zero, then the initial window computation is done according to rfc2414. If this value is non-zero, the initial (congestion) window is initialized a number of maximum sized segments equal to **tcp_init_window**. Changing f**tcp_init_window** allows you to tune the TCP slow start to control the number of TCP segments (packets) outstanding before an ACK is received. For example, setting this value to 6 would allow 6 packets to be sent initially, instead of the normal 2 or 3 packets, thus speeding up the initial packet rate.
tcp_inpcb_hashtab_siz	**Purpose:** Specifies the size of the inpcb hash table for TCP connections. **Tuning:** This table holds the **inpcbs** required for connection management and is implemented as a table of hash chains. A larger table means that the linked hash chains will be smaller and lower traversal time on the average but the memory footprint will be larger. This value should be a prime number. This option impacts performance and should be used with extreme caution. Please consult a performance analyst in case it is felt that the value needs to be changed. The execution environment could have an influence on the value. It is strongly encouraged to maintain the system defined defaults as they tend to execute optimally in most environments.
tcp_keepcnt	**Purpose:** **tcp_keepcnt** represents the number of keepalive probes that could be sent before terminating the connection.
tcp_keepidle	**Purpose:** Specifies the length of time to keep the connection active, measured in half seconds.
tcp_keepinit	**Purpose:** Sets the initial timeout value for a TCP connection, which is measured in half seconds.
tcp_keepintvl	**Purpose:** Specifies the interval, which is measured in half seconds, between packets that are sent to validate the connection. **Tuning:** For example, 150 half seconds results in 75 seconds between validation probes. This allows TCP to know that a connection is still valid and keep the connection open when it is otherwise idle. This is a configuration decision with minimal performance consequences. No change is recommended. If the interval were shortened significantly, processing and bandwidth costs might become significant.

Item	Description
tcp_limited_transmit	**Purpose:** Enables the feature that enhances TCP's loss recovery as described in the RFC 3042. **Tuning:** A value of 1 enables this option and zero disables the option.
tcp_low_rto	**Purpose:** Specifies the TCP retransmit timeout (RTO) in milliseconds for connections that are experiencing packet drops. **Tuning:** A tick is 10 ms (one 100th of a second). The option `timer_wheel_tick` must be set to non-zero value before setting the `tcp_low_rto` option. Also, `tcp_low_rto` can be equal to zero or a multiple of ten times the `timer_wheel_tick` value. This tunable allows TCP to use smaller timeout values for packet timeout and retransmit on high speed networks. Normal TCP retransmit timeout is 1.5 seconds.
tcp_maxburst	**Purpose:** Specifies the number of back-to-back packets that TCP can send before pausing to allow those packets to be forwarded to their destination. **Tuning:** This can be useful if routers are unable to handle large bursts of TCP packets and are dropping some of them. A value of 0 means no limitation for back-to-back packets before pausing.
tcp_maxqueuelen	**Purpose:** Specifies the maximum number of TCP segments that can be processed in the reassembly queue. **Tuning:** Values for this tunable parameter are in the range 0 - 32767. A value of 0 means unlimited queue length. The default value is 1000.
tcp_mssdflt	**Purpose:** Default maximum segment size that is used in communicating with remote networks. **Tuning:** **tcp_mssdflt** is only used if path MTU discovery is not enabled or path MTU discovery fails to discovery a path MTU. The **tcp_mssdflt** network option can also be set on a per interface basis (see the documentation for ISNO options). Limiting data to (MTU - 40) bytes ensures that, where possible, only full packets are sent.
tcp_nagle_limit	**Purpose:** This is the Nagle algorithm threshold in bytes, which can be used to disable Nagle. **Tuning:** The default is Nagle turned on. To disable Nagle, set this value to 0 or 1. TCP disables Nagle for data segments larger than or equal to this threshold value.

Item	Description
tcp_nagleoverride	**Purpose:** Setting the option tcp_nagle_limit turns off the Nagle algorithm system wide and setting tcp_nodelay option for a socket turns off the Nagle algorithm for that specific connection whereas setting tcp_nagleoverride disables the Nagle algorithm only for certain situations during the connection. **Tuning:** The value of 1 disables Nagle algorithm only for certain TCP packets in a connection.
tcp_ndebug	**Purpose:** Specifies the number of **tcp_debug** structures.
tcp_newreno	**Purpose:** Enables the modification to TCP's Fast Recovery algorithm as described in RFC 2582. **Tuning:** This fixes the limitation of TCP's Fast Retransmit algorithm to recover fast from dropped packets when multiple packets in a window are dropped. sack also achieves the same thing but sack needs support from both ends of the TCP connection; the NewReno modification is only on the sender side.
tcp_nodelayack	**Purpose:** Turning this parameter on causes TCP to send immediate acknowledgement (Ack) packets to the sender. When **tcp_nodelayack** is disabled, TCP delays sending Ack packets by up to 200ms. This allows the Ack to be piggy-backed onto a response and minimizes system overhead. **Tuning:** This option can be used to overcome bugs in other implementations of the TCP nagle algorithm. Setting this option to 1 will cause slightly more system overhead, but can result in much higher performance for network transfers if the sender is waiting on the receiver's acknowledgement.
tcp_pmtu_discover	**Purpose:** Enables or disables path MTU discovery for TCP applications. **Tuning:** A value of 0 disables path MTU discovery for TCP applications, while a value of 1 enables it.
tcp_recvspace	**Purpose:** Specifies the system default socket buffer size for receiving data. This affects the window size used by TCP. **Tuning:** The optimum buffer size is the product of the media bandwidth and the average round-trip time of a packet. The **tcp_recvspace** network option can also be set on a per interface basis (reference documentation on Interface Specific Network Options (ISNO)). Most interfaces now have this tunable set in the ISNO defaults. The **tcp_recvspace** attribute must specify a socket buffer size less than or equal to the setting of the **sb_max** attribute.

Item	Description
tcp_sendspace	**Purpose:** Specifies the system default socket buffer size for sending data. **Tuning:** The optimum buffer size is the product of the media bandwidth and the average round-trip time of a packet: `optimum_window=bandwidth * average_round_trip_time`. The **tcp_sendspace** network option can also be set on a per interface basis (reference documentation on Interface Specific Network Options (ISNO)). Most interfaces now have this tunable set in the ISNO defaults. The **tcp_sendspace** attribute must specify a socket buffer size less than or equal to the setting of the **sb_max** attribute.
tcp_syn_rto	**Purpose:** Specifies the TCP retransmission timeout (RTO) value, in interval of half-seconds, for a connection experiencing packet drops before the connection is established. **Tuning:** The value of the **tcp_syn_rto** tunable parameter will be set as the initial retransmission timeout value for retransmissions that occur before the connection is established. The values are in the range 0 - 32767. The default value is 0.
tcp_tcpsecure	**Purpose:** Specifies whether connection reset attacks and data corruption attacks on TCP are avoided. **Tuning:** This option is used to protect TCP connections from one or more of the following three vulnerabilities. The first vulnerability involves sending of a fake SYN to an established connection to abort the connection. A **tcp_tcpsecure** value of 1 provides protection from this vulnerability. The second vulnerability involves the sending of a fake RST to an established connection to abort the connection. A **tcp_tcpsecure** value of 2 provides protection from this vulnerability. The third vulnerability involves injecting fake data in an established TCP connection. A **tcp_tcpsecure** value of 4 provides protection from this vulnerability. Values for **tcp_tcpsecure** can range from a minimum of 0 (this is the default value and provides no protection from these vulnerabilities) to a maximum value of 7. Values of 3, 5, 6, or 7 protects the connection from combinations of these three vulnerabilities.
tcp_timewait	**Purpose:** The **tcp_timewait** option is used to configure how long connections are kept in the **timewait** state. **Tuning:** It is given in 15 second intervals. Increasing this value degrades performance of web servers or applications that open and close many TCP connections.
tcp_ttl	**Purpose:** Specifies the time to live for TCP packets, expressed in ticks. **Tuning:** A tick is 0.6 seconds (there are 100 ticks per minutes).

Item	Description
tcprexmtthresh	**Purpose:** Specifies the number of consecutive duplicate acknowledgements, which cause TCP to goto fast retransmit phase. **Tuning:** Increase this parameter if TCP performance is low due to an increased number of duplicate acknowledgements but the network is not congested. Be aware that setting a high value for this option can cause TCP to time out and retransmit.
tcptr_enable	**Purpose:** Enables TCP traffic regulation that is defined by policies that created by using the tcptr command. A value of 0 means disabled. Any non-zero value means traffic regulation is enabled. **Tuning:** A value of 0 disables this option. This option must be turned on for servers that must protect against network attacks.
thewall	**Purpose:** Specifies the maximum amount of memory, in kilobytes, that is allocated to the memory pool. **Tuning:** Cannot be set anymore.
timer_wheel_tick	**Purpose:** Specifies the slot interval of the timer wheel, in ticks, where a tick=1000/HZ=10ms. **Tuning:** This attribute is used with `tcp_low_rto` attribute to reduce the TCP timeout values to smaller units.
tn_filter	**Purpose:** The option is valid for Trusted AIX environment only. If the option is disabled in this environment, the MAC checks are bypassed at the IP layer.
udp_bad_port_limit	**Purpose:** Specifies the number of UDP packets to a port with no socket that can be received in a 500-millisecond period before UDP stops sending ICMP errors in response to such packets. **Tuning:** If set to 0, ICMP errors will always be sent when UDP packets are received for a bad port number. If greater than 0, it specifies the number of packets to be received before UDP stops sending ICMP errors.
udp_ephemeral_high	**Purpose:** Specifies the largest port number to allocate for UDP ephemeral ports.
udp_ephemeral_low	**Purpose:** Specifies the smallest port number to allocate for UDP ephemeral ports.

Item	Description
udp_inpcb_hashtab_siz	**Purpose:** Specifies the size of the inpcb hash table for UDP connections. This table holds the inpcbs that is required for connection management and is implemented as a table of hash chains. A larger table means that the linked hash chains is smaller and lower traversal time on the average but the memory footprint is larger. **Tuning:** This value must be a prime number. This option impacts performance and must be used with extreme caution. Consult a performance analyst in case it is felt that the value must be changed. The execution environment can have an influence on the value. It is encouraged to maintain the system defined defaults as they tend to run optimally in most environments.
udp_pmtu_discover	**Purpose:** Enables or disables path MTU discovery for UDP applications. **Tuning:** UDP applications must be written to use path MTU discovery. A value of 0 disables the feature, while a value of 1 enables it.
udp_recvspace	**Purpose:** Specifies the system default socket buffer size for receiving UDP data. **Tuning:** Change when nonzero n in **netstat -s** report of udp: n socket buffer overflows. The **udp_recvspace** parameter must specify a socket buffer size less than or equal to the setting of the **sb_max** parameter. Increase size, preferably to multiple of 4096.
udp_send_perf	**Purpose** Improves the UDP Transmit performance by caching address information and Memory Buffers (**mbufs**) that are used to transmit packets over a network. **Tunning** The default value is 0 and it disables caching. To enable caching, specify a value of 1. For example, to enable caching, enter the following command: `no -o udp_send_perf=1`
udp_sendspace	**Purpose:** Specifies the system default socket buffer size (in bytes) for sending UDP data. **Tuning:** The **udp_sendspace** attribute must specify a socket buffer size less than or equal to the setting of the **sb_max** attribute. **udp_sendspace** must be at least as large as the largest datagram size that the application sends. Increase size, preferably to multiple of 4096.
udp_ttl	**Purpose:** Specifies the time to live (in seconds) for UDP packets.

Item	Description
udpcksum	**Purpose:** Allows UDP checksum to be turned on/off. **Tuning:** A value of 0 turns it off; while a value of 1 turns it on.
use_sndbufpool	**Purpose:** Enables caching of mbuf clusters to improve performance. **Tuning:** If this value is disabled, then to allocate a mbuf cluster, AIX allocates a cluster buffer and also a mbuf buffer to point to it, thus requiring two buffer allocation operations. Likewise, to free the cluster, two buffer free operations are required. With this option enabled, AIX maintains a cache of clusters for each cluster size that is being used. This improves performance by reducing overhead to allocate and free mbuf clusters. The default value of 1 enables this option on a system-wide scale. The mbuf cluster cache can be displayed by using the **netstat -M** command.

Compatibility Mode

When running in pre 5.2 compatibility mode that is controlled by the **pre520tune** attribute of **sys**0, see **AIX 5.2 compatibility mode**. The reboot values for parameters, except those of type Bosboot, are not applicable because in the pre 5.2 compatibility mode they are not applied during boot.

In pre 5.2 compatibility mode, setting reboot values to tuning parameters continues to be achieved by embedding calls to tuning commands in scripts that are called during the boot sequence. Parameters of type **Reboot** can therefore be set without the **-r** flag, so that existing scripts continue to work.

This mode is automatically turned ON when a machine is MIGRATED to AIX 5L Version 5.2. For complete installations, it is turned OFF and the reboot values for parameters are set by applying the content of the **/etc/tunables/nextboot** file during the reboot sequence. Only in that mode are the **-r** and **-p** flags fully functional. See **Kernel Tuning** in the *Performance Tools Guide and Reference* for details.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the maximum size of the mbuf pool, type:

```
no -o thewall
```

2. To reset the time to live for UDP packets its default size, type:

```
no -d udp_ttl
```

3. To change the default socket buffer sizes on your system, type:

```
no -r -o tcp_sendspace=32768
no -r -o udp_recvspace=32768
```

4. To use a system as an internet work router over Internet Protocol networks, type:

```
 no -o ipforwarding=1
```

5. To list the current and reboot value, range, unit, type and dependencies of all tunable parameters that are managed by the **no** command, type:

```
no -L
```

6. To display the help information about the udp_ephemeral_high option, type:

```
no -h udp_ephemeral_high
```

7. To permanently turn off the ip6srcrouteforward option, type:

```
no -p -o ip6srcrouteforward=0
```

8. To list the reboot values for all Network tuning parameters, type:

```
no -r -a
```

9. To list (spreadsheet format) the current and reboot value, range, unit, type and dependencies of all tunable parameters that are managed by the **no** command, type:

```
no -x
```

10. To log all allocations and frees of type mbuf or socket that are size 256 or 4096, type:

```
no -o net_buf_type={mbuf:socket} -o net_buf_size={256:4096} -o net_malloc_police=1
```

11. To log all allocations and frees of type mbuf, type:

```
no -o net_buf_type={mbuf} -o net_buf_size={all} -o net_malloc_police=1
```

12. To log all **ns_alloc**s and **ns_free**s for en0 or en3 by using a 2000 events buffer size, type:

```
no -o ndd_event_name={en0:en3} -o ndd_event_tracing=2000
```

13. To log all **ns_alloc**s and **ns_free**s for all en adapters by using a 2000 events buffer size, type:

```
no -o ndd_event_name={en} -o ndd_event_tracing=2000
```

14. To log all **ns_alloc**s and **ns_free**s for all adapters, type:

```
no -o ndd_event_name={all} -o ndd_event_tracing=1
```

nohup Command

Purpose

Runs a command without hangups.

Syntax

nohup { -p *pid* | *Command* [*Arg* ...] [**&**] }

Description

The **nohup** command runs the command specified by the *Command* parameter and any related *Arg* parameters, ignoring all hang up signals (SIGHUP) or modifies the process specified with the **-p** option to ignore all SIGHUP signals. SIGHUP is a signal that is sent to a process when the controlling terminal of the process is closed.

The **nohup** command can also be used to run programs in the background after logging off. To run a nohup command in the background, add an & (ampersand) to the end of the command.

If the standard error is displayed on the terminal and if the standard output is neither displayed on the terminal, nor sent to the output file specified by the user (the default output file is `nohup.out`), both the `./nohup.out` and `$HOME/nohup.out` files are not created or opened for appending the error message. The **nohup** command does not execute the parameter utility that is specified with the **nohup** command and exits with exit status 127.

Note: The **-p** *pid* and *Command* options can not be specified together.

When the **-p** *pid* flag is used, the output of the specified process will not be re-directed to the `nohup.out` file.

Flags

Item	Description
-p *pid*	*pid* is the process-id of a running process. The nohup command modifies the specified process, to ignore all hangup (SIGHUP) signals.

Exit Status

This command returns the following exit values:

Item	Description
126	The command specified by the *Command* parameter was found but could not be invoked.
127	An error occurred in the nohup command or the command specified by the *Command* parameter could not be found.

Otherwise, the exit status of the nohup command is that of the command specified by the *Command* parameter.

Examples

1. To run a command in the background after you log off, enter:

   ```
   $ nohup find / -print &
   ```

   After you enter this command, the following is displayed:

   ```
   670
   $ Sending output to nohup.out
   ```

   The process ID number changes to that of the background process started by & (ampersand). The message `Sending output to nohup.out` informs you that the output from the **find / -print** command is in the **nohup.out** file. You can log off after you see these messages, even if the `find` command is still running.
2. To run a command in the background and redirect the standard output to a different file, enter:

   ```
   $ nohup find / -print >filenames &
   ```

   This example runs the **find / -print** command and stores its output in a file named `filenames`. Now only the process ID and prompt are displayed:

   ```
   677
   $
   ```

   Wait before logging off because thenohup command takes a moment to start the command specified by the *Command* parameter. If you log off too quickly, the command specified by the *Command* parameter may not run at all. Once the command specified by the *Command* parameter starts, logging off does not affect it.
3. To run more than one command, use a shell procedure. For example, if you write the shell procedure:

```
neqn math1 | nroff > fmath1
```

and name it the `nnfmath1` file, you can run the nohup command for all of the commands in the `nnfmath1` file with the command:

```
nohup sh nnfmath1
```

4. If you assign execute permission to the `nnfmath1` file, you get the same results by issuing the command:

```
nohup nnfmath1
```

5. To run the `nnfmath1` file in the background, enter:

```
nohup nnfmath1
&
```

6. To run the `nnfmath1` file in the Korn shell, enter:

```
nohup ksh nnfmath1
```

7. To make a running process ignore all hangup signals, enter:

```
nohup -p 161792
```

enotifyevent Command, notifyevent Command

Purpose

Mails event information generated by the event response resource manager (ERRM) to a specified user ID.

Syntax

`enotifyevent` [`-h`] [*user-ID*]

`notifyevent` [`-h`] [*user-ID*]

Description

The `enotifyevent` script always return messages in English. The language in which the messages of the `notifyevent` script are returned depend on the locale settings.

These scripts capture event information that is posted by the event response resource manager (ERRM) in environment variables that are generated by the ERRM when an event occurs. These scripts can be used as actions that are run by an event response resource. They can also be used as templates to create other user-defined actions.

Event information is returned about the ERRM environment variables, and also includes the following:

Local Time
Time when the event or rearm event is observed. The actual environment variable supplied by ERRM is ERRM_TIME. This value is localized and converted to readable form before being displayed.

In AIX, these scripts use the `mail` command to send event information to the specified user ID. When a user ID is specified, it is assumed to be valid, and it is used without verifying it. If a user ID is not specified, the user who is running the command is used as the default.

user-ID is the optional ID of the user to whom the event information will be mailed. If *user-ID* is not specified, the user who is running the command is used as the default.

Flags

-h
Writes the script's usage statement to standard output.

Parameters

log_file
Specifies the name of the file where event information is logged. An absolute path for the *log_file* parameter should be specified.

For AIX, the *log_file* is treated as a circular log and has a fixed size of 64KB. When *log_file* is full, new entries are written over the oldest existing entries.

For other platforms, the size of the *log_file* is not limited, and it will not overwrite itself. The file size will increase indefinitely unless the administrator periodically removes entries.

If *log_file* already exists, event information is appended to it. If *log_file* does not exist, it is created so that event information can be written to it.

Exit Status

0
Command has run successfully.

Restrictions

1. These scripts must be run on the node where the ERRM is running.
2. The `mail` command is used to read the file.

Standard Output

When the `-h` flag is specified, the script's usage statement is written to standard output.

Examples

1. You can use the **mail** command to read the contents of the event information. The following example shows how a warning event for the `/var` file system (a file system resource) is formatted and logged:

```
============================================================================
Event reported at Sun Mar 26 16:38:03 2002

Condition Name:                      /var space used
Severity:                   Warning
Event Type:                    Event
Expression:                    PercentTotUsed>90

Resource Name:                    /var
Resource Class Name:                IBM.FileSystem
Data Type:                   CT_UINT32
Data Value:                   91
```

Location

/opt/rsct/bin/enotifyevent
Contains the `enotifyevent` script

/opt/rsct/bin/notifyevent
Contains the `notifyevent` script

nroff Command

Purpose

Formats text for printing on typewriter-like devices and line printers.

Syntax

nroff [**-e**] [**-h**] [**-i**] [**-q**] [**-z**] [**-o** *List*] [**-n** *Number*] [**-s** *Number*] [**-r** *ANumber*] [**-u** *Number*] [**-T** *Name*] [**-man**] [**-me**] [**-mm**] [**-mptx**] [**-ms**] [*File* ... | **-**]

Description

The **nroff** command reads one or more files for printing on typewriter-like devices and line printers. If no file is specified or the **-** (minus sign) flag is specified as the last parameter, standard input is read by default. The *File* variable specifies files to be printed on a typewriter-like device by the **nroff** command. The default is standard input.

The **col** command may be required to postprocess **nroff** command output in certain cases.

Flags

Item	Description
-e	Produces equally spaced words in adjusted lines, using the full resolution of a particular terminal.
-h	Uses output tabs during horizontal spacing to speed output and reduce the output character count. Tab settings are assumed to be every eight nominal character widths.
-i	Reads standard input after reading all specified files.
-man	Selects the **man** macro processing package.
-me	Selects the **me** macro processing package.
-mm	Selects the **mm** macro processing package.
-mptx	Selects the **mptx** macro processing package.
-ms	Selects the **ms** macro processing package.
-n *Number*	Assigns the specified number to the first printed page.
-o *List*	Prints only those pages specified by the *List* variable, which consists of a comma-separated list of page numbers and ranges, as follows: • A range of *Start-Stop* means print pages *Start* through *Stop.* For example, 9-15 prints pages 9 through 15. • An initial *-Stop* means print from the beginning to page *Stop.* • A final *Start-* means print from page *Start* to the end. • A combination of page numbers and ranges prints the specified pages. For example, -3, 6-8,10,12- prints the beginning through page 3, pages 6 through 8, page 10, and page 12 to the end. **Note:** When the **-o***List* flag is used in a pipeline (as with one or more of the **eqn** or **tbl** commands) you may receive a `broken pipe` message if the last page in the document is not specified in the *List* parameter. This broken pipe message is not an indication of any problem and can be ignored.

Item	Description
-q	Calls the simultaneous input/output mode of the **.rd** request.
-r *ANumber*	Sets register *A* to the specified number. The value specified by the *A* variable must have a one-character ASCII name.
-s *Number*	Stops every specified number of pages (the default is 1). The **nroff** command halts every specified number of pages to allow paper loading or changing, then resumes upon receipt of a linefeed or new-line character. This flag does not work in pipelines (for example, with the **mm** command). When the **nroff** command halts between pages, an ASCII BEL character is sent to the workstation.
-T *Name*	Prepares the output for the specified printing device. Typewriter-like devices and line printers use the following *Name* variables for international extended character sets, as well as English-language character sets, digits, and symbols: **hplj** Hewlett-Packard LaserJet II and other models in the same series of printers. **ibm3812** 3812 Pageprinter II. **ibm3816** 3816 Pageprinter. **ibm4019** 4019 LaserPrinter. **Note:** The 4019 and the HP Laser Jet II printer both have nonprintable areas at the top and bottom of a page. If a file is targeted for these printers, be sure to define top and bottom margins (for example, by formatting with the **-mm** flag) so that all output can be positioned within the printable page. **37** Teletype Model 37 terminal (default) for terminal viewing only. This device does not support extended characters that are inputted by the \[N] form. Inputting Extended Single-Byte Characters provides more information. **lp** Generic name for printers that can underline and tab. All text sent to the **lp** value using reverse linefeeds (for example, text that includes tables) must be processed with the **col** command. This device does not support extended characters that are inputted by the \[N] form. Inputting Extended Single-Byte Characters provides more information. **ppds** Generic name for printers that support the personal printer data streams such as the Quietwriter III, Quickwriter, and Proprinters. **ibm5575** 5575 Kanji Printer. **ibm5577** 5577 Kanji Printer. **Note:** For completeness of the text formatting system, the following devices are shipped *as is* from the AT&T Distribution center. No support is provided for these tables.

Item	Description
-T *Name* (Continued)	**2631** Hewlett-Packard 2631 printer in regular mode. **2631-c** Hewlett-Packard 2631 printer in compressed mode. **2631-e** Hewlett-Packard 2631 printer in expanded mode. **300** DASI-300 printer. **300-12** DASI-300 terminal set to 12 characters per inch. **382** DTC-382. **4000a** Trendata 4000a terminal (4000A). **450** DASI-450 (Diablo Hyterm) printer. **450-12** DASI-450 terminal set to 12 characters per inch. **832** Anderson Jacobson 832 terminal. **8510** C.ITOH printer. **tn300** GE Terminet 300 terminal. **X** Printers equipped with a TX print train. **300s** DASI-300s printer (300S). **300s-12** DASI-300s printer set to 12 characters per inch (300S-12).
-u *Number*	Sets the bold factor (number of character overstrokes) for the third font position (bold) to the specified number, or to 0 if the *Number* variable is missing.
-z	Prints only messages generated by **.tm** (workstation message) requests. **Note:** See the Macro Packages for Formatting Tools in the **troff** command for information about the macros.
-	Forces input to be read from standard input.

Files

Item	Description
/usr/share/lib/tmac/tmac.*	Contains pointers to standard macro files.
/usr/share/lib/macros/*	Contains standard macro files.
/usr/share/lib/nterm/*	Contains the terminal driving tables for the **nroff** command.
/usr/share/lib/pub/terminals	Contains a list of supported terminals.

nslookup Command

Purpose

Queries internet domain name servers interactively.

Syntax

nslookup [- option] [name | -] [server]

Description

The **nslookup** command queries internet domain name servers in two modes. Interactive mode allows you to query name servers for information about various hosts and domains, or to print a list of the hosts in a domain. In noninteractive mode, the names and requested information are printed for a specified host or domain.

The **nslookup** command enters interactive mode when no arguments are given, or when the first argument is a - (minus sign) and the second argument is the host name or internet address of a name server. When no arguments are given, the command queries the default name server. The **nslookup** command enters non-interactive mode when you give the name or internet address of the host to be looked up as the first argument. The optional second argument specifies the host name or address of a name server. You can specify options on the command line if they precede the arguments and are prefixed with a hyphen. For example, to change the default query type to host information, and the initial timeout to 10 seconds, enter the following command:

```
nslookup -query=hinfo  -timeout=10
```

Interactive commands

Item	Description
host [*server*]	Looks up information for the host using the current default server or using server, if specified. If the host is an Internet address and the query type is **A** or **PTR**, the **nslookup** command returns the name of the host. If the host is a name and does not have a trailing period, the search list is used to qualify the name. To look up a host not in the current domain, append a period to the name.
server *Domain* **lserver** *Domain*	Changes the default server to the value specified by the *Domain* parameter. The **lserver** subcommand uses the initial server to look up information about the domain. The **server** subcommand uses the current default server. If an authoritative answer cannot be found, the names of any additional servers that might have the answer are returned.
exit	Exits the program.

Item	Description
set *Keyword*[**=***Value*]	Changes state information that affects lookups. You can specify the following keywords: **all** Prints the current values of the frequently used options to **set**. Information about the current default server and host is also printed. **class=*value*** Changes the query class to one of the following value. The class specifies the protocol group of the information. The default is **IN**. **IN** The Internet class. **CH** The Chaos class. **HESIOD** The Hesiod class. **ANY** Wildcard (any of the above). **[no]debug** Turns debugging mode on. The default is **nodebug**. **[no]d2** Turns comprehensive debugging on. The default is **nod2**. **domain=*name*** Changes the default domain name to the name specified by the *name* parameter. **[no]search** Appends the domain names in the domain search list to the request until an answer is received, if the lookup request contains a period other than a trailing period. The default is **search**. **port=*value*** Changes the default TCP/UDP name server port to the number specified by the *value* parameter. The default value is 53. **querytype=*value*type=*value*** Changes the type of the information query to the type specified by the *value* parameter. The default value is A. **[no]recurse** Tells the name server to query other servers if it does not have the information. The default is **recurse**. **retry=*number*** Sets the number of retries to the number specified by the *number* parameter. **timeout=*number*** Changes the initial timeout interval for waiting for a reply to the seconds specified by the *number* parameter. **[no]vc** Always uses a virtual circuit when sending requests to the server. The default is **novc**. **[no]fail** Tries the next name server if a name server responds with SERVFAIL or a referral (**nofail**) or terminate query (**fail**) on such a response. The default is **nofail**.

Files

Item	Description
/etc/resolv.conf	Contains the initial domain name and nameserver addresses.

nsupdate Command

Purpose

Updates a DNS server.

Syntax

Refer to the syntax for the **nsupdate4**, **nsupdate8**or **nsupdate9** command.

Description

AIX 7.1 supports only BIND version 9. BIND 8 application code is removed from AIX 7.1 and the **named** daemon links to **named9** now, and **nsupdate** to **nsupdate4**. To use a different version of **nsupdate**, you must relink the symbolic links accordingly to the **nsupdate** command.

For example, to use **nsupdate9**, type:

```
ln -fs /usr/sbin/nsupdate9 /usr/sbin/nsupdate
```

nsupdate4 can be used with **named8** (which is now removed from AIX 7.1), but **nsupdate9** must be used with **named9** because the security process is different.

Files

/usr/sbin/named
Contains a symbolic link to the version of **named** being used on the system.

/usr/sbin/nsupdate
Contains a symbolic link to the version of **nsupdate** being used on the system.

/usr/sbin/nsupdate4
Contains the BIND version 4 **nsupdate** command.

/usr/sbin/nsupdate8
Contains the BIND version 8 **nsupdate** command.

/usr/sbin/nsupdate9
Contains the BIND version 9 **nsupdate** command.

nsupdate4 Command

Purpose

Updates a DNS server.

Syntax

nsupdate4 [**-a**] [**-g**] [**-i**] [**-q**] [**-v**] [**-?**] [**-k** *KeyFile*] [**-h** *HostName*] [**-d** *DomainName*] [**-p** *PrimaryName*] [**-r** *IPAddress*] [**-s** "*CommandString*"]

Description

The **nsupdate4** command updates the DNS server. The **nsupdate4** command runs in either interactive mode or command mode. If a command string is provided, the **nsupdate4** command runs the command string and then exits. The return code is dependent upon the successfulness of the command string.

The valid internal commands for the command string or interactive modes are:

Item	Description
r	Reset update packets. This must be first.
d	Delete a record. Following this command are questions for a record type and the value to delete.
a	Add a record. Following this command are questions for a record type and the value to add.
n	Add a record only if it doesn't exist yet. Following this command are questions for a record type and the value to add.
e	Add a record only if it already exists. Following this command are questions for a record type and the value to add.
t	Sets the default time to live value for the updated records.

Item	Description
s	Signs the update. Depending on if the **-a** or **-g** flags were specified, a key will be generated and the update will be signed.
x	Transmit the update packet to the server specified by the **-p** flag.
v	Turns on or off verbose mode.
i	Returns the information passed in by the parameters.
p	Prints the update packet in record format.
q	Exits the command

The **-g** flag allows you to generate a set of keys to distribute to clients for use in secure mode. This flag takes the hostname and the primary name and generates a public and a private key. For secure mode zone operation, the public is entered into the DNS server's database for the data to secure and the private key is placed on the client so that it can update that information at a later time.

The **-a** flag allows you to enter administrative mode. The zone may be secured by a zone key. This key gives the user full access to the zone. The **-a** flag tries to use the zone key for update signatures instead of the individual records key.

Flags

Item	Description
-a	Administrative mode. Attempts to use zone key instead of individual records key.
-d *DomainName*	Specifies the name of the domain to apply the update to. This is used with all records except PTR records.
-g	Generation mode. Used to generate a key pair for a primary name and a hostname.
-h *HostName*	Specifies the name of the record to update. This is used with all records except PTR records.
-i	Ignores errors and runs all the commands in the string.
-k *KeyFile*	Specifies the name of the default keyfile. This is the file for keys.
-p *PrimaryName*	Specifies the name or IP address of a DNS server. The primary DNS server is prefered.
-q	Turns off output.
-r *IPAddress*	Specifies the IP Address of the record to update. This is used only with PTR records.
-s "*CommandString*"	A set of internal commands separated by spaces or colons.
-v	Turns on verbose output.
-?	Command line options list

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.

Item	Description
>0	An error occurred.

Security

Access Control: Any User

Example

To initialize a packet, delete all A records for the specified hostname, add an A record for the hostname to 9.3.145.2 association, signed and valid for 300 seconds with a default KEY pad of 3110400, transmit the packet, and quit, enter: (where ";" is pressing the enter key)

```
r;d;a;*;a;a;9.3.145.2;s;300;3110400;x;q
```

If any one of the items had failed, a message would be printed. In command line mode, an error would cause the program to exit and return 1.

Files

Item	Description
/usr/sbin/nsupdate4	Contains the **nsupdate4** command.
/usr/sbin/named	Contains the DNS server.

nsupdate8 Command

Purpose

Generates a DNS update packet readable by a BIND 8 nameserver.

Syntax

nsupdate8 [**-v**] [**-d**] [*Filename*]

Description

The **nsupdate8** command can read from a file specified on the command line, from stdin for pipes or redirected input from a file, or interactively from a tty. All three methods use the same format specified below. The input defines a DNS update packet that can be used to update a ZONE. There are two sections to an update, a prerequisite section and an update section. The DNS name server verifies that all the prerequisites are true before processing the update section.

Flags

Item	Description
-d	Causes **nsupdate8** to generate additional debug information about its actions.
-v	Tells **nsupdate8** to use a virtual circuit (TCP connection), instead of the usual UDP connection.

The input format is defined as a set of update packets. Each packet is a set of strings terminated with a newline. The last string in the input stream may end with an EOF. If the stream is to contain multiple update packets, each packet must be separated from the next packet by a blank line (single newline

character). The semi-colon is used a comment character. Anything after it is ignored and thrown out of the update packet.

The input format for nsupdate8 is a follows:

```
section opcode  name [ttl] [class] [type] [data]
```

This is the general form. Each value of *section* and *opcode* modify what is required for later arguments.

Item	Description
section	Defines the section of the update this record is for. Values are: **prereq** Indicates the record is for the prerequisites section. **update** Indicates the record is for the update section.
opcode	Defines the operation to do with this record. **Values are:** **Prerequisite operations:** **nxdomain** Indicates that the name should be checked for non-existance. The ttl must be a non-zero value to indicate for how long it shouldn't exist. An optional class can be specified to restrict the search to only that class. The type of T_ANY is used as a wildcard to match any record type. **nydomain** Indicates that the name should be checked for existance. The ttl must be a non-zero value to indicate for how long the name should continue to exist. An optional class is allowed to restrict the search to only that class. The record type is T_NONE. This forces the check to make sure the name exists. **nxrrset** Indicates that the record of a specific type doesn't exist for the name. An optional class and ttl are allowed to restrict the search. A type is mandatory. **nyrrset** Indicates that the record of a specific type must exist for the name. The ttl and class are optional to restrict the search. The type and data are mandatory. Data may be a wild card. If the data is not a wildcard, it must match the format for the type specified. **Values are:** **Update operations:** **add** Indicates that the record should be added to the zone. The type and data are mandatory. Wildcards are not allowed as data. The ttl is mandatory and must be non-zero. The class is optional. **delete** Indicates that the record should be deleted from the zone. The type and data are optional. A wildcard is allowed for data. data defaults to the NULL string and type defaults to T_ANY. ttl and class are optional. If ttl is specified, it is reset to 0.
name	The name of the DNS entry that one is testing or modifying.
[*ttl*]	Optional time-to-live for the record being added. In some forms, this is not optional.

Item	Description
[*class*]	Class of the record to be added to the zone. Values are IN, HESIOD, and CHAOS. The default for all messages is IN.
[*type*]	The type of the record to be added to or checked against the zone. Values are A, NS, CNAME, SOA, MB, MR, NULL, WKS, PTR, HINFO, MINFO, MX, TXT, RP, AFSDB, X25, ISDN, RT, NSAP, NSAP_PTR, PX, and LOC. NOTE: The CNAME type may only be added with TSIG and TKEY records which are not currently supported in BIND 8.
[*data*]	The data to be added or checked against the zone. The data should be valid for the specified type and in the DOMAIN data file format of a DNS server zone file. For prerequisite checking, an asterik (*) is used to match any value. This can also be used to delete all records of a particular type.

Here are the specific format cases:

```
prereq nxdomain <name> <ttl != 0> [class]
prereq nydomain <name ttl != 0> [class]
prereq nxrrset <name> [ttl] [class] <type>
prereq nyrrset <name> [ttl] [class] <type> <data>
update delete <name> [ttl] [class] [type] [data]
update add <name> <ttl != 0> [class] <type> <data>
```

Diagnostics

Messages indicating the different actions done and/or problems encountered by the program.

nsupdate9 Command

Purpose

Dynamic DNS update utility.

Syntax

nsupdate9 [**-d**] [**-y** [*hmac:*] *keyname: secret* | **-k** *keyfile*] [**-t** *timeout*] [**-u** *udptimeout*] [**-r** *udpretries*] [**-v**] [*filename*]

Description

The **nsupdate9** command is used to submit Dynamic DNS Update requests as defined in RFC2136 to a name server. This allows resource records to be added or removed from a zone without manually editing the zone file. A single update request can contain requests to add or remove more than one resource record.

Zones that are under dynamic control via **nsupdate9** or a DHCP server should not be edited by hand. Manual edits could conflict with dynamic updates and cause data to be lost.

The resource records that are dynamically added or removed with **nsupdate9** have to be in the same zone. Requests are sent to the zone's master server. This is identified by the MNAME field of the zone's SOA record.

The **-d** option makes **nsupdate9** operate in debug mode. This provides tracing information about the update requests that are made and the replies received from the name server.

Transaction signatures can be used to authenticate the Dynamic DNS updates. These use the TSIG resource record type described in RFC2845 or the SIG(0) record described in RFC3535 and RFC2931. The signatures rely on a shared secret that should only be known to **nsupdate9** and the name server. Currently, the only supported encryption algorithm for TSIG is HMAC-MD5, which is defined in RFC 2104. Once other algorithms are defined for TSIG, applications will need to ensure they select the

appropriate algorithm as well as the key when authenticating each other. For instance suitable key and server statements would be added to **/etc/named.conf** so that the name server can associate the appropriate secret key and algorithm with the IP address of the client application that will be using TSIG authentication. SIG(0) uses public key cryptography. To use a SIG(0) key, the public key must be stored in a KEY record in a zone served by the name server. **nsupdate9** does not read **/etc/named.conf.**

nsupdate9 uses the **-y** or **-k** option to provide the shared secret needed to generate a TSIG record for authenticating Dynamic DNS update requests. The default type is HMAC-MD5. These options are mutually exclusive. With the **-k** option, **nsupdate9** reads the shared secret from the file *keyfile*, whose name is of the form **K{name}.+157.+{random}.private**. For historical reasons, the file **K{name}.+157.+{random}.key** must also be present. When the **-y** option is used, a signature is generated from [*hmac*:] *keyname:secret*. *keyname* is the name of the key, and *secret* is the base64 encoded shared secret. Use of the **-y** option is discouraged because the shared *secret* is supplied as a command line argument in clear text. This may be visible in the output from ps(1) or in a history file maintained by the user's shell.

You can also use the **-k** flag to specify a SIG(0) key used to authenticate Dynamic DNS update requests. In this case, the key specified is not an HMAC-MD5 key.

By default **nsupdate9** uses UDP to send update requests to the name server unless they are too large to fit in a UDP request in which case TCP is used. The **-v** option makes **nsupdate9** use a TCP connection. This may be preferable when a batch of update requests is made.

Flags

Item	Description
-d	Makes **nsupdate9** operate in debug mode.
-k *keyfile*	Reads the shared secret from the file *keyfile*.
-r *udpretries*	Sets the number of UDP retries. The default is 3. If zero, only one update request is made.
-t *timeout*	Sets the maximum time a update request can take before it is aborted. The default is 300 seconds. You can use zero to disable the timeout.
-u *udptimeout*	Sets the UDP retry interval. The default is 3 seconds. If zero, the interval is computed from the timeout interval and number of UDP retries.
-v	Makes **nsupdate9** use a TCP connection.
-y [*hmac:*] *keyname:secret*	Generates a signature from keyname:*secret*.

Parameters

Item	Description
filename	File to be updated.

Input Format

nsupdate9 reads input from the file *filename* or standard input. Each command is supplied on exactly one line of input. Some commands are for administrative purposes. The others are either update instructions or prerequisite checks on the contents of the zone. These checks set conditions that some name or set of resource records (RRset) either exists or is absent from the zone. These conditions must be met if the entire update request is to succeed. Updates will be rejected if the tests for the prerequisite conditions fail.

Every update request consists of zero or more prerequisites and zero or more updates. This allows a suitably authenticated update request to proceed if some specified resource records are present or missing from the zone. A blank input line (or the **send** command) causes the accumulated commands to be sent as one Dynamic DNS update request to the name server.

The command formats and their meaning are as follows:

Item	Description
server [*servername*] [*port*]	Sends all dynamic update requests to the name server *servername*. When no **server** statement is provided, **nsupdate9** will send updates to the master server of the correct zone. The MNAME field of that zone's SOA record will identify the master server for that zone. *port* is the port number on *servername* where the dynamic update requests get sent. If no *port* number is specified, the default DNS port number of 53 is used.
local [*address*] [*port*]	Sends all dynamic update requests using the local address. When no local statement is provided, **nsupdate9** will send updates using an *address* and *port* choosen by the system. *port* can additionally be used to make requests come from a specific port. If no port number is specified, the system will assign one.
zone [*zonename*]	Specifies that all updates are to be made to the zone *zonename*. If no zone statement is provided, **nsupdate9** will attempt determine the correct zone to update based on the rest of the input.
key [*name*] [*secret*]	Specifies that all updates are to be TSIG signed using the *keyname keysecret* pair. The key command overrides any key specified on the command line via **-**y or **-k**.
prereq nxdomain [*domain-name*]	Requires that no resource record of any type exists with name *domain-name*.
prereq yxdomain [*domain-name*]	Requires that *domain-name* exists (has as at least one resource record, of any type).
prereq nxrrset [*domain-name*] [*class*] [*type*]	Requires that no resource record exists of the specified *type*, *class* and *domain-name*. If class is omitted, IN (internet) is assumed.
prereq yxrrset [*domain-name*] [*class*] [*type*]	This requires that a resource record of the specified *type*, *class* and *domain-name* must exist. If class is omitted, IN (internet) is assumed.
prereq yxrrset [*domain-name*] [*class*] [*type*] [*data...*]	The data from each set of prerequisites of this form sharing a common *type*, *class*, and *domain-name* are combined to form a set of RRs. This set of RRs must exactly match the set of RRs existing in the zone at the given *type*, *class*, and *domain-name*. The *data* are written in the standard text representation of the resource record's RDATA.
update delete [*domain-name*] [*ttl*] [*class*] [*type*] [*data...*]	Deletes any resource records named *domain-name*. If *type* and *data* is provided, only matching resource records will be removed. The internet *class* is assumed if class is not supplied. The *ttl* is ignored, and is only allowed for compatibility.
update add [*domain-name*] [*ttl*] [*class*] [*type*] [*data...*]	Adds a new resource record with the specified *ttl*, *class* and *data*.
show	Displays the current message, containing all of the prerequisites and updates specified since the last send.
send	Sends the current message. This is equivalent to entering a blank line.
answer	Displays the answer.

Lines beginning with a semicolon are comments and are ignored.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

Note: The **nsupdate9** command does not sort two updates combined in one update into different zones. Two updates need to be made individually by inserting a blank line or the **send** command between them.

The examples below show how **nsupdate9** could be used to insert and delete resource records from the example.com zone. Notice that the input in each example contains a trailing blank line so that a group of commands are sent as one dynamic update request to the master name server for example.com.

```
# nsupdate9
> update delete oldhost.example.com A
> update add newhost.example.com 86400 A 172.16.1.1
>
```

Any A records for oldhost.example.com are deleted. and an A record for newhost.example.com it IP address 172.16.1.1 is added. The newly-added record has a 1 day TTL (86400 seconds)

```
# nsupdate9
> prereq nxdomain nickname.example.com
> update add nickname.example.com CNAME somehost.example.com
> send
```

The prerequisite condition gets the name server to check that there are no resource records of any type for **nickname.example.com**. If there are, the update request fails. If this name does not exist, a CNAME for it is added. This ensures that when the CNAME is added, it cannot conflict with the long-standing rule in RFC1034 that a name must not exist as any other record type if it exists as a CNAME. (The rule has been updated for DNSSEC in RFC2535 to allow CNAMEs to have SIG, KEY and NXT records.)

```
# nsupdate9
> update delete 61.26.31.9.in-addr.arpa 0 IN PTR
> update add 61.26.31.9.in-addr.arpa 86400 IN PTR newhost.example.com.
```

Any PTR records for IP address 9.31.26.61 are deleted and a PTR record for IP address 9.31.26.61 and hostname **newhost.example.com** is added. The newly-added record has a 1 day-TTL (86400 seconds).

Files

Item	Description
/etc/resolv.conf	Used to identify default name server
K{name}.+157.+{random}.key	Base-64 encoding of HMAC-MD5 key created by **dnssec-keygen**(8).
K{name}.+157.+{random}.private	Base-64 encoding of HMAC-MD5 key created by **dnssec-keygen**(8).

ntpd4 Daemon

Purpose

Network Time Protocol (NTP) Daemon.

Syntax

ntpd4 [**-4**] [**-6**] [**-a**] [**-A**] [**-b**] [**-c** *conffile*] [**-d**] [**-D** *level*] [**-f** *driftfile*] [**-g**] [**-i** *jaildir*] [**-k** *keyfile*] [**-l** *logfile*] [**-L**] [**-n**] [**-N**] [**-p** *pidfile*] [**-P** *priority*] [**-q**] [**-r** *broadcastdelay*] [**-s** *statsdir*] [**-t** *key*] [**-u** *user[:group]*] [**-U** *interface update interval*] [**-v** *variable*] [**-V** *variable*] [**-x**]

Description

The **ntpd** program is an operating system daemon that sets and maintains the system time-of-day in synchrony with the Internet Standard Time servers. The **ntpd** program is a complete implementation of the Network Time Protocol (NTP) version 4, and also retains compatibility with version 3, as defined by the RFC-1305, and version 1 and 2, as defined by RFC-1059 and RFC-1119. The **ntpd** program generally computes in 64-bit floating point arithmetic mode. If a precision of 232 picoseconds needs to be maintained, then **ntpd** computes in 64-bit fixed-point mode. The ultimate precision of 232 picoseconds is not achievable with existing workstations and networks, however, this precision may be required with future Gigahertz CPU clocks and Gigabit LANs.

Frequency discipline

The **ntpd** behavior at startup depends on the frequency file, usually **ntp.drift**. This file contains the latest estimate of clock frequency error. When the **ntpd** daemon is started and the file does not exist, the **ntpd** enters a special mode designed to quickly adapt to the particular system clock oscillator time and frequency error. This takes approximately 15 minutes, after which the time and frequency are set to nominal values and the **ntpd** enters normal mode of operation, where the time and frequency are continuously tracked relative to the server. After one hour the frequency file is created and the current frequency offset is written to this file. When the **ntpd** is started and the file does exist, the **ntpd** frequency is initialized from the file and **ntpd** enters the normal mode of operation. After that the current frequency offset is written to the file at hourly intervals.

Operating modes

The **ntpd** program can operate in any of the several modes, including symmetric active/passive, client/server, and broadcast/multicast. The **ntpd** normally operates continuously while monitoring for small changes in frequency and trimming the clock for the ultimate precision. The **ntpd** can operate in a one-time mode where the time is set from an external server and frequency is set from a previously recorded frequency file. A broadcast or multicast client can discover remote servers, compute server-client propagation delay correction factors and configure itself automatically. This makes it possible to deploy a fleet of workstations without specifying configuration details specific to the local environment.

By default, **ntpd** runs in continuous mode where each of the possibly several external servers are polled at intervals determined by an intricate state machine. The state machine measures the incidental roundtrip delay jitter and the oscillator frequency wander and determines the best poll interval using a heuristic algorithm. Ordinarily, and in most operating environments, the state machine starts with 64-seconds intervals and eventually increases in steps to 1024 seconds. A small amount of random variation is introduced in order to avoid bunching at the servers. In addition, should a server become unreachable for some time, the poll interval is increased in steps to 1024 seconds in order to reduce network overhead.

In some cases, it might not be practical for **ntpd** to run continuously. A common workaround has been to run the **ntpdate** program from a **cron** job at designated times. However, this program does not have the crafted signal processing, error checking and mitigation algorithms of **ntpd**. The **-q** option is intended for this purpose. Setting this option will cause **ntpd** to exit just after setting the clock for the first time. The procedure for initially setting the clock is the same as in continuous mode; most applications specify the **iburst** command with the server configuration command. With this command a volley of messages are exchanged to groom the data and the clock is set in to about 10 second. If no response is received, after a couple of minutes, the daemon times out and exits. After a certain period if no response is received, the **ntpdate** program is stopped.

Flags

Item	Description
-4	Forces DNS resolution of hostnames to the IP version 4 namespace.
-6	Force DNS resolution of hostnames to the IP version 6 namespace.
-a	Requires cryptographic authentication for broadcast client, multicast client and symmetric passive associations. This is the default value.
-A	Does not require cryptographic authentication for broadcast client, multicast client, and symmetric passive associations.
-b	Enables the client to synchronize to broadcast servers.
-c *conffile*	Specifies the name and path of the configuration file, default `/etc/ntp.conf`.
-d	Specifies debugging mode. This option may occur more than once, with each occurrence indicating greater detail of display.
-D *level*	Specifies the debugging level directly.
-f *driftfile*	Specifies the name and path of the frequency file, default **/etc/ntp.drift**. This is the same operation as the **driftfile driftfile** configuration command.
-g	Allows the time to be set to any value without restriction; this can happen only once. The **ntpd** command exits with a message to the system log if the offset exceeds the panic threshold, which is 1000 seconds by default. If the threshold is exceeded after that, **ntpd** will exit with a message to the system log. This option can be used with the **-q** and **-x** options.
-i *jaildir*	The **chroot** command directs the server to the directory **jaildir**. This option also implies that the server attempts to drop root privileges at startup (otherwise, **chroot** gives little additional security), and it is only available if the operating system supports to run the server without full root privileges. You must specify a **-u** option.
-k *keyfile*	Specifies the name and path of the symmetric key file, default `/etc/ntp.keys`. This is the same operation as the keys keyfile configuration command.
-l *logfile*	Specifies the name and path of the log file. The default is the system log file. This is the same operation as the **logfile** configuration command.
-L	Does not listen to virtual IPs. The default is to listen.
-n	Does not fork.
-N	Runs the **ntpd** at the highest priority level to the extent permitted by the operating system.
-p *pidfile*	Specifies the name and path of the file used to record the **ntpd** process ID. This is the same operation as the **pidfile pidfile** configuration command.
-P *priority*	Runs the **ntpd** at the specified priority to the extent permitted by the operating system.
-q	Exits the **ntpd** just after the first time the clock is set. This behavior mimics that of the **ntpdate** program, which is to be retired. The **-g** and **-x** options can be used with this option. **Note:** The kernel time discipline is disabled with this option.
-r *broadcastdelay*	Specifies the default propagation delay from the broadcast/multicast server to the client. This is necessary only if the delay cannot be computed automatically by the protocol.
-s *statsdir*	Specifies the directory path for files created by the statistics facility. This is the same operation as the **statsdir** configuration command.
-t *key*	Adds a key number to the trusted key list. This option can occur more than once.

Item	Description
-u *user[:group]*	Specifies an user, and optionally a group, to switch. This option is only available if the operating system supports running the server without complete root privileges.
-U *interface update interval*	Specifies the number of seconds to wait between the interface list scans to pick up new and deleted network interface. Set to 0 to disable dynamic interface list updating. The default is to scan every 5 minutes.
-v *variable*	Adds a system variable listed by default.
-V *variable*	
-x	Slews the time if the offset is less than the step threshold, which is 128 milliseconds by default, and steps up if above the threshold. This option sets the threshold to 600 seconds, which is well within the accuracy window to set the clock manually.

Exit Status

This command returns the following exit values:

0
Successful completion.

> 0
An error occurred.

Security

Access Control : You must have root authority to run this command.

Auditing Events : N/A

Examples

By default, the symbolic link `/usr/sbin/xntpd` points to NTP version 3 daemon (`/usr/sbin/ntp3/xntpd`). To run NTP version 4 daemon (`/usr/sbin/ntp4/ntpd4`), modify the symbolic link so that it points to the version 4 daemon.

```
/usr/sbin/xntpd-->
/usr/sbin/ntp4/ntpd4
```

1. To start the **xntpd** daemon, enter:

   ```
   startsrc -s xntpd
   ```

2. To stop the **xntpd** daemon, enter:

   ```
   stopsrc -s xntpd
   ```

By default, the NTP symbolic link `/usr/sbin/xntpd` points to the NTP version 3 daemon (`/usr/sbin/ntp3/xntpd`). To run the NTP version 4 daemon (`/usr/sbin/ntp4/ntpd4`), switch to the NTP version 4 binaries by using the following command:

```
/usr/sbin/ntp_ssw -v4
```

After you run the **ntp_ssw** command successfully, all NTP symbolic links point to NTP version 4 binaries as shown in the following example:

```
/usr/sbin/ntpq -> /usr/sbin/ntp4/ntpq4
/usr/sbin/sntp -> /usr/sbin/ntp4/sntp4
/usr/sbin/ntptrace -> /usr/sbin/ntp4/ntptrace4
/usr/sbin/xntpd -> /usr/sbin/ntp4/ntpd4
/usr/sbin/ntpdate -> /usr/sbin/ntp4/ntpdate4
/usr/sbin/ntp-keygen -> /usr/sbin/ntp4/ntp-keygen4
```

Files

Item	Description
`/usr/sbin/ntp4/ntpd4`	Contains the **ntpd4** daemon. Default symbolic link to NTP version 3 binary from `/usr/sbin` directory. `/usr/sbin/xntpd --> /usr/sbin/ntp3/xntpd`
`/etc/ntp.conf`	Contains the default configuration file.
`/etc/ntp.drift`	Contains the default drift file.

ntpdate Command

Purpose

Sets the date and time using the Network Time Protocol (NTP).

Syntax

ntpdate [**-b**] [-c] [**-d**] [**-s**] [**-u**] [**-a** *Keyid*] [**-e** *AuthenticationDelay*] [**-k** *KeyFile*] [**-o** *Version*] [**-p** *Samples*] [**-t** *TimeOut*] *Server ...*

Description

The **ntpdate** command sets the local date and time by polling the NTP servers specified to determine the correct time. It obtains a number of samples from each server specified and applies the standard NTP clock filter and selection algorithms to select the best of the samples.

The **ntpdate** command makes time adjustments in one of the following ways:

- If it determines that the clock is off by more than 0.5 seconds, it steps the clock's time by calling the **settimeofday** subroutine. This is the preferred method at boot time.
- If it determines that the clock is off by less than 0.5 seconds, it slews the clock's time by calling the **adjtime** subroutine with the offset. This method tends to keep a badly drifting clock more accurate, though at some expense to stability. When running the **ntpdate** command on a regular basis from the **cron** command instead of running a daemon, doing so once every hour or two results in precise enough timekeeping to avoid stepping the clock.

  Notes:

  1. The **ntpdate** command's reliability and precision improves dramatically with a greater number of servers. Although you can use a single server, you obtain better performance by providing at least three or four servers.
  2. If an NTP server daemon like the **xntpd** daemon is running on the same host, the **ntpdate** command will decline to set the date.
  3. You must have root authority on the local host to run this command.

Flags

Item	Description
-a *Keyid*	Enable the authentication function and authenticate all packets using *Keyid*. By default, the authentication function is disabled.
-b	Step the clock's time by calling the **settimeofday** subroutine.
-c	Slew the clock's time by calling the **adjtime** subroutine.
-d	Specifies debug mode. Determines what results the **ntpdate** command produces without actually doing them. The results appear on the screen. This flag uses unprivileged ports.
-e *AuthenticationDelay*	Specifies the amount of time in seconds to delay the authentication processing. Typical values range from 0.0001 to 0.003.
-k *KeyFile*	Specifies a different name for the file containing the keys when not using the default **/etc/ntp.keys** file. See ... for the description of the *KeyFile*.
-o *Version*	Specifies the NTP version implementation to use when polling its outgoing packets. The values for *Version* can be 1, 2 or 3. The default is 3.
-p *Samples*	Specifies the number of samples to acquire from each server. The values for *Samples* can be between 1 and 8 inclusive. The default is 4.
-s	Specifies the use of the syslog facility to log actions instead of using standard output. Useful when running the **ntpdate** command with the **cron** command.
-t *TimeOut*	Specifies the amount of time to wait for a response. The value given for *TimeOut* is rounded to a multiple of 0.2 seconds. The default is 1 second.
-u	Specifies the use of an unprivileged port to send the packets from. Useful when you are behind a firewall that blocks incoming traffic to privileged ports, and you want to synchronize with hosts beyond the firewall. A firewall is a system or machine that controls the access from outside networks to a private network.

Parameters

Item	Description
Server ...	Specifies the servers to poll.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Auditing Events: N/A

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To set the local date and time by polling the NTP servers at address `9.3.149.107`, enter:

```
/usr/sbin/ntpdate 9.3.149.107
```

Output similar to the following appears:

```
28 Feb 12:09:13 ntpdate [18450]: step time server 9.3.149.107
offset 38.417792 sec
```

Files

Item	Description
/usr/sbin/ntpdate	Contains the **ntpdate** command.
/etc/ntp.keys	Contains the default key file.

ntpdate4 Command

Purpose

Sets the date and time using the Network Time Protocol (NTP).

Syntax

ntpdate4 [**-4**] [**-6**] [**-a** *key*] [**-B**] [**-b**] [**-d**] [**-e** *authdelay*] [**-k** *keyfile*] [**-o** *version*] [**-p** *samples*] [**-q**] [**-s**] [**-t** *timeout*] [**-u**] [**-v**] *server* [...]

Description

The **ntpdate** command sets the local date and time by polling the Network Time Protocol (NTP) server(s) given as the server arguments to determine the correct time. The **ntpdate** must be run as root on the local host. Samples are obtained from each of the servers specified and a subset of the NTP clock filter and selection algorithms are applied to select the best. Note that the accuracy and reliability of **ntpdate** depends on the number of servers, the number of polls each time it is run and the interval between runs.

The **ntpdate** can be run manually as necessary to set the host clock, or it can be run from the host startup script to set the clock at boot time. This is useful in some cases to set the clock initially before starting the NTP daemon **ntpd**. It is also possible to run **ntpdate** from a **cron** script. However, it is important to note that **ntpdate** with contrived **cron** scripts is not a substitute for the NTP daemon, which uses complex algorithms to maximize accuracy and reliability while minimizing resource use. Finally, since **ntpdate** does not tune the host clock frequency as does ntpd, the accuracy using **ntpdate** is limited.

Time adjustments are made by **ntpdate** in one of two ways. If **ntpdate** determines that the clock is in error of more than 0.5 seconds it will simply step the time by calling the system **settimeofday()** routine. If the error is less than 0.5 seconds, it will slew the time by calling the system **adjtime** () routine. The latter technique is less disruptive and more accurate when the error is small, and works quite well when **ntpdate** is run by **cron** every hour or two.

The **ntpdate** will decline to set the date if an NTP server daemon (**ntpd**) is running on the same host. When running **ntpdate** on a regular basis from **cron** as an alternative to running a daemon, doing so once every hour or two will result in precise enough timekeeping to avoid stepping the clock.

Note: Where a host name is expected, a -4 qualifier preceding the host name forces DNS resolution to the IP version 4 namespace, while a -6 qualifier forces DNS resolution to the IP version 6 namespace.

Flags

Item	Description
- 4	Forces DNS resolution of following host names on the command line to the IP v4 namespace
- 6	Forces DNS resolution of following host names on the command line to the IP v6 namespace.
- a *key*	Enables the authentication function and specifies the key identifier to be used for authentication as the argument **keyntpdate**. The keys and key identifiers must match in both the client and server key files. The default is to disable the authentication function.
- B	Forces the time to be slewed using the **adjtime** () system call, even if the measured offset is greater than + or - 128 millisecond. The default is to step the time using **settimeofday** () if the offset is greater than + or -128 millisecond. Note that, if the offset is much greater than + or -128 millisecond in this case, that it can take a long time (hours) to slew the clock to the correct value. During this time the host should not be used to synchronize clients.
- b	Forces the time to be stepped using the **settimeofday** () system call, rather than slewed (default) using the **adjtime** () system call. This option should be used when called from a startup file at boot time.
- d	Enables the debugging mode, in which **ntpdate** will go through all the steps, but not adjust the local clock. Information useful for general debugging is also printed.
- e *authdelay*	Specifies the processing delays to perform an authentication function as the value *authdelay*, in seconds and fraction (See the ntpd for more details). This number is usually small enough to be negligible for most purposes, though specifying a value may improve timekeeping on very slow CPUs.
- k *keyfile*	Specifies the path for the authentication key file as the string keyfile. The default is `/etc/ntp.keys`.
- o *version*	Specifies the NTP version for outgoing packets as the integer version, which can be 1 or 2. The default is 3. This allows **ntpdate** to be used with older NTP versions.
- p *samples*	Specifies the number of samples to be acquired from each server as integer samples, with values from 1 to 8 inclusive. The default value is 4.
- q	Specifies the query. Does not set the clock.
- s	Diverts logging output from the standard output (default) to the system **syslog** facility. This is designed primarily for convenience of **cron** scripts.
- t *timeout*	Specifies the maximum time waiting for a server response as the value timeout, in seconds and fraction. The value is rounded to a multiple of 0.2 seconds. The default is 1 second, a value suitable for polling across a LAN.
- u	Directs **ntpdate** to use an unprivileged port or outgoing packets. You can use this option when behind a firewall that blocks incoming traffic to privileged ports, and you want to synchronize with hosts beyond the firewall. Note that the **-d** option always uses unprivileged ports.
- v	Verbose output. This option causes the **ntpdate** version identification string to be logged.

Parameters

Item	Description
Server...	Specifies the servers to poll

Exit Status

This command returns the following exit values:

Item	Description
`0`	Successful completion.
`>0`	An error occurred.

Security

Item	Description
`Access Control`	You must have root privilege to run this command.
`Auditing Events`	N/A

Examples

1. To set the local date and time by polling the NTP servers at address 9.41.254.24, enter:

   ```
   ntpdate 9.41.254.24
   ```

   Output similar to the following appears:

   ```
   address: ::
   address: 0.0.0.0
   25 Feb 12:19:41 ntpdate[434262]: adjust time server 9.41.254.24 offset -0.005270  sec
   ```

Files

Item	Description
`/usr/sbin/ntp4/ntpdate4`	Contains the **ntpdate** command for NTP version 4. Default Symbolic link to NTP version 3 binary from `/usr/sbin` directory. `/usr/sbin/ntpdate --> /usr/sbin/ntp3/ntpdate`
`/etc/ntp.keys`	Encryption keys used by **ntpdate**.

ntpdc4 Command

Purpose

Starts the query or control program for the Network Time Protocol (NTP) daemon, **ntpd**.

Syntax

ntpdc [**-4**] [**-6**] [**-d**] [**-i**] [**-l**] [**-n**] [**-p**] [**-s**] [**-c** *command*] [*host*] [...]

Description

The **ntpdc** command is used to query the **ntpd** daemon about its current state and to request changes in the state. The program may be run either in interactive mode or controlled using command line arguments. Extensive state and statistics information is available through the **ntpdc** interface. In addition, all the configuration options which can be specified at startup using **ntpd**'s configuration file might also be specified at run time using **ntpdc**.

If one or more request options are included in the command line when **ntpdc** is executed, each of the requests will be sent to the NTP servers running on each of the hosts given as command line arguments, or on localhost by default. If no request options are given, **ntpdc** will attempt to read commands from the standard input and execute these on the NTP server running on the first host given on the command line, again defaulting to localhost when no other host is specified. **ntpdc** will prompt for commands if the standard input is a terminal device.

ntpdc uses NTP mode 7 packets to communicate with the NTP server, and hence can be used to query any compatible server on the network which permits it. Note that since NTP is a UDP protocol this communication will be somewhat unreliable, especially over large distances in terms of network topology. **ntpdc** makes no attempt to retransmit requests, and will time requests out if the remote host is not heard from within a suitable timeout time.

The operation of **ntpdc** are specific to the particular implementation of the ntpd daemon and can be expected to work only with this and maybe some previous versions of the daemon. Requests from a remote **ntpdc** program that affects the state of the local server must be authenticated, which requires both the remote program and local server share a common key and key identifier.

Note that in contexts where a host name is expected, a -4 qualifier preceding the host name forces DNS resolution to the IP version 4 namespace, while a -6 qualifier forces DNS resolution to the IP version 6 namespace.

Specifying a command line option other than **-i** or **-n** will cause the specified query (queries) to be sent to the indicated host(s) immediately. Else, **ntpdc** will attempt to read interactive format commands from the standard input.

Flags

Item	Description
-4	Forces DNS resolution of following host names on the command line to the IP version 4 namespace.
-6	Forces DNS resolution of following host names on the command line to the IP version 6 namespace.
-c *command*	The following argument is interpreted as an interactive format command and is added to the list of commands to be executed on the specified host(s). You can run multiple **-c** options.
-d	Enables debugging mode. You can use this option more than once.
-i	Forces **ntpdc** to operate in interactive mode. Prompts will be written to the standard output and commands read from the standard input.
-l	Obtains a list of peers, which are known to the server(s). This switch is equivalent to **-c** *listpeers*.
-n	Outputs all host addresses in dotted-quad numeric format rather than converting them to the canonical host names.
-p	Prints a list of the peers known to the server as well as a summary of their state. This is equivalent to **-c** *peers*.

Item	Description
-s	Prints a list of the peers known to the server as well as a summary of their state. The print format is different from the **-p** switch. This is equivalent to **-c** *dmpeers*.

Parameters

Item	Description
Host...	Specifies the hosts.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must be part of the system group to run this command.

Auditing Events: N/A

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To start the query/control program for the Network Time Protocol daemon, enter:

```
ntpdc
```

2. To print a list of the peers known to the server as well as a summary of their state, enter:

```
ntpdc -p
```

Output similar to the following appears:

```
     remote              local        st   poll reach  delay     offset   disp
==============================================================================
ausgsa.austin.ibm.com 9.124.101.190    2   64    1    0.29128 -0.013381 2.81735
```

ntpdc Internal Commands

Interactive Commands

Interactive format commands consist of a keyword followed by zero to four arguments. Only enough characters of the full keyword to uniquely identify the command need be typed. The output of a command is normally sent to the standard output, but optionally the output of individual commands may be sent to a file by appending a <, followed by a file name, to the command line.

A number of interactive format commands are executed entirely within the ntpdc program itself and do not result in NTP mode 7 requests being sent to a server. The following list describes the interactive commands.

Item	Description
? [command_keyword] or **help [command_keyword]**	A question mark (?) by itself prints a list of all the command keywords known to this incarnation of **ntpq**. A question mark (?) followed by a command keyword will print function and the use of the command.
delay milliseconds	Specifies a time interval to be added to timestamps included in requests which require authentication. This is used to enable (unreliable) server reconfiguration over long delay network paths or between machines whose clocks are unsynchronized.
host hostname	Sets the host to which future queries will be sent. Hostname may be either a host name or a numeric address.
hostnames [yes | no]	If yes is specified, host names are printed in information displays. If no is specified, numeric addresses are printed instead. The default is yes, unless modified using the command line **-n** switch.
keyid keyid	Allows the specification of a key number to be used to authenticate configuration requests from **ntpdc** to the host(s). This must correspond to a key number which the host/server has been configured to use for this purpose (server options: trustedkey, and requestkey). If authentication is not enabled on the host(s) for **ntpdc** commands, the command **keyid 0** should be given, else the keyid of the next subsequent addpeer/addserver/broadcast command will be used.
quit	Exits **ntpdc**.
passwd	Prompts you to type in a password (which will not be echoed) which will be used to authenticate configuration requests. The password must correspond to the key configured for use by the NTP server for this purpose if such requests are to be successful.
timeout milliseconds	Specifies a timeout period for responses to server queries. The default is about 8000 milliseconds. **Note: ntpdc** retries each query once after a timeout, and hence the total waiting time for a timeout will be twice the timeout value set.

Control Message Commands

Query commands result in NTP mode 7 packets containing requests for information being sent to the server. These are read-only commands do not make any modification to the server configuration state.

Item	Description
listpeers	Obtains and prints a brief list of the peers for which the server is maintaining state. These should include all configured peer associations as well as those peers whose stratum is such that they are considered by the server to be possible future synchronization candidates.

Item	Description
peers	Obtains a list of peers for which the server is maintaining state, along with a summary of that state. Summary information includes the address of the remote peer, the local interface address (0.0.0.0 if a local address has yet to be determined), the stratum of the remote peer (a stratum of 16 indicates the remote peer is unsynchronized), the polling interval, in seconds, the reachability register, in octal, and the current estimated delay, offset and dispersion of the peer, all in seconds. The character in the left margin indicates the mode this peer entry is operating in. A + denotes symmetric active, a - indicates symmetric passive, a = means the remote server is being polled in client mode, a ^ indicates that the server is broadcasting to this address, a ~ denotes that the remote peer is sending broadcasts and a * marks the peer the server is currently synchronizing to. The contents of the host field may be one of four forms. It may be a host name, an IP address, a reference clock implementation name with its parameter or REFCLK(implementation number, parameter). On hostnames no only IP-addresses will be displayed.
dmpeers	A slightly different peer summary list. Identical to the output of the peers command, except for the character in the leftmost column. Characters only appear beside peers which were included in the final stage of the clock selection algorithm. A . indicates that this peer was cast off in the falseticker detection, while a + indicates that the peer made it through. A * denotes the peer the server is currently synchronizing with.
showpeer peer_address [...]	Shows a detailed display of the current peer variables for one or more peers. Most of these values are described in the NTP Version 2 specification.
pstats peer_address [...]	Displays per-peer statistic counters associated with the specified peer(s).
clockinfo clock_peer_address [...]	Obtains and print information concerning a peer clock. The values obtained provide information on the setting of fudge factors and other clock performance information.
kerninfo	Obtains and print kernel phase-lock loop operating parameters. This information is available only if the kernel has been specially modified for a precision timekeeping function.
loopinfo [oneline | multiline]	Prints the values of selected loop filter variables. The loop filter is the part of NTP which deals with adjusting the local system clock. The offset is the last offset given to the loop filter by the packet processing code. The frequency is the frequency error of the local clock in parts-per-million (ppm). The time_const controls the stiffness of the phase-lock loop and thus the speed at which it can adapt to oscillator drift. The watchdog timer value is the number of seconds which have elapsed since the last sample offset was given to the loop filter. The oneline and multiline options specify the format in which this information is to be printed, with multiline as the default.

Item	Description
sysinfo	Print a variety of system state variables, i.e., state related to the local server. All except the last four lines are described in the NTP Version 3 specification, RFC-1305. The system flags show various system flags, some of which can be set and cleared by the enable and disable configuration commands, respectively. These are the auth, bclient, monitor, pll, pps and stats flags. See the ntpd documentation for the meaning of these flags. There are two additional flags which are read only, the kernel_pll and kernel_pps. These flags indicate the synchronization status when the precision time kernel modifications are in use. The kernel_pll indicates that the local clock is being disciplined by the kernel, while the kernel_pps indicates the kernel discipline is provided by the PPS signal. The stability is the residual frequency error remaining after the system frequency correction is applied and is intended for maintenance and debugging. In most architectures, this value will initially decrease from as high as 500 ppm to a nominal value in the range .01 to 0.1 ppm. If it remains high for some time after starting the daemon, something may be wrong with the local clock, or the value of the kernel variable tick may be incorrect. The broadcastdelay shows the default broadcast delay, as set by the broadcastdelay configuration command. The authdelay shows the default authentication delay, as set by the authdelay configuration command.
sysstats	Prints statistics counters maintained in the protocol module.
memstats	Prints statistics counters related to memory allocation code.
iostats	Prints statistics counters maintained in the input-output module.
timerstats	Prints statistics counters maintained in the timer/event queue support code.
reslist	Obtains and print the server's restriction list. This list is printed in sorted order and may help to understand how the restrictions are applied.
ifstats	Lists interface statistics for interfaces used by ntpd for network communication.
ifreload	Forces the scan of current system interfaces. Outputs interface statistics for interfaces that could possibly change. Marks unchanged interfaces with ., added interfaces with + and deleted interfaces with -.
monlist [version]	Obtains and prints traffic counts collected and maintained by the monitor facility. The version number should not normally need to be specified.
clkbug clock_peer_address [...]	Obtains debugging information for a reference clock driver. This information is provided only by some clock drivers and cannot be decoded without a copy of driver source.

Runtime Configuration Requests

All requests which cause state changes in the server are authenticated by the server using a configured NTP key (the facility can also be disabled by the server by not configuring a key). The key number and the corresponding key must also be made known to **ntpdc**. This can be done using the **keyid** and **passwd** commands, the latter of which will prompt at the terminal for a password to use as the encryption key. You will also be prompted automatically for both the key number and password the first time a command which would result in an authenticated request to the server is given. Authentication not only provides verification that the requester has permission to make such changes, but also gives an extra degree of protection again transmission errors.

Authenticated requests always include a timestamp in the packet data, which is included in the computation of the authentication code. This timestamp is compared by the server to its receive time stamp. If they differ by more than a small amount the request is rejected. This is done for two reasons. First, it makes simple replay attacks on the server, by someone who might be able to overhear traffic on your LAN, much more difficult. Second, it makes it more difficult to request configuration changes to your server from topologically remote hosts. While the reconfiguration facility will work well with a server on the local host, and may work adequately between time-synchronized hosts on the same LAN, it will work very poorly for more distant hosts. As such, if reasonable passwords are chosen, care is taken in the distribution and protection of keys and appropriate source address restrictions are applied, the run time reconfiguration facility should provide an adequate level of security.

The following commands run authenticated requests.

Item	Description
addpeer peer_address [keyid] [version] [minpoll# | prefer | iburst | burst | minpoll N | maxpoll N [...]] **addpeer peer_address [prefer | iburst | burst | minpoll N | maxpoll N | keyid N | version N [...]]**	Add a configured peer association at the given address and operating in symmetric active mode. Note that an existing association with the same peer may be deleted when this command is executed, or may simply be converted to conform to the new configuration, as appropriate. If the **keyid** is nonzero, all outgoing packets to the remote server will have an authentication field attached encrypted with this key. If the value is 0 (or not given) no authentication will be done. If **ntpdc**'s key number has not yet been set (e.g., by the **keyid** command), it will be set to this value. The **version#** can be 1 through 4 and defaults to 3. The remaining options are either a numeric value for **minpoll** or literals prefer, **iburst**, **burst**, **minpoll N**, **keyid N**, **version N**, or **maxpoll N** (where N is a numeric value), and have the action as specified in the peer configuration file command of ntpd. Each flag (or its absence) replaces the previous setting. The prefer keyword indicates a preferred peer (and thus will be used primarily for clock synchronization if possible). The preferred peer also determines the validity of the PPS signal - if the preferred peer is suitable for synchronization so is the PPS signal.
addserver peer_address [keyid] [version] [minpoll# | prefer | iburst | burst | minpoll N | maxpoll N [...]] **addserver peer_address [prefer | iburst | burst | minpoll N | maxpoll N | keyid N | version N [...]]**	Identical to the **addpeer** command, except that the operating mode is client.
broadcast peer_address [keyid] [version] [prefer]	Identical to the **addpeer** command, except that the operating mode is broadcast. In this case a valid non-zero key identifier and key are required. The **peer_address** parameter can be the broadcast address of the local network or a multicast group address assigned to NTP. If a multicast address, a multicast-capable kernel is required.
unconfig peer_address [...]	This command causes the configured bit to be removed from the specified peer(s). In many cases this will cause the peer association to be deleted. When appropriate, however, the association may persist in an unconfigured mode if the remote peer is willing to continue on in this fashion.
fudge peer_address [time1] [time2] [stratum] [refid]	This command provides a way to set certain data for a reference clock. See the source listing for further information.

Item	Description
enable [auth | bclient | calibrate | kernel | monitor | ntp | pps | stats]	These commands operate in the same way as the enable and disable configuration file commands of **ntpd**.
disable [auth | bclient | calibrate | kernel | monitor | ntp | pps | stats]	
restrict address mask flag [flag]	This command operates in the same way as the restrict configuration file commands of **ntpd**.
unrestrict address mask flag [flag]	Removes the restriction of the matching entry from the restrict list.
delrestrict address mask [ntpport]	Deletes the matching entry from the restrict list.
readkeys	Causes the current set of authentication keys to be purged and a new set to be obtained by reading the keys file again (which must have been specified in the **ntpd** configuration file). This allows encryption keys to be changed without restarting the server.
trustedkey keyid [...]	These commands operate in the same way as the **trustedkey** and **untrustedkey** configuration file commands of **ntpd**.
untrustedkey keyid [...]	
authinfo	Returns information concerning the authentication module, including known keys and counts of encryptions and decryptions which have been done.
traps	Displays the traps set in the server. See the source listing for further information.
addtrap [address [port] [interface]	Sets a trap for asynchronous messages. See the source listing for further information.
clrtrap [address [port] [interface]	Clears a trap for asynchronous messages. See the source listing for further information.
reset	Clears the statistics counters in various modules of the server. See the source listing for further information.

Files

Item	Description
/usr/sbin/ntp4/ntpdc4	Contains the **ntpdc** command. The default symbolic link to NTP version 3 binaries from the `/usr/sbin` directory. `/usr/sbin/ntpdc --> /usr/sbin/ntp3/xntpdc`

ntp-keygen4 Command

Purpose

Generate public and private keys.

Syntax

ntp-keygen [**-d**] [**-e**] [**-G**] [**-g**] [**-H**] [**-I**] [**-M**] [**-P**] [**-T**] [**-c** [*RSA-MD2* | *RSA-MD5* | *RSA-SHA* | *RSA-SHA1* | *RSA-MDC2* | *RSA-RIPEMD160* | *DSA-SHA* | *DSA-SHA1*]] [**-i** *name*] [**-m** *modulus*] [**-p** *password*] [**-q** *password*] [**-S** [*RSA* | *DSA*]] [**-s** *name*] [**-v** *nkeys*] [**-V** *params*]

Description

The **ntp-keygen4** command generates cryptographic data files used by the NTP version 4 authentication and identification schemes. It generates MD5 key files used in symmetric key cryptography. In addition, if the OpenSSL software library has been installed, it generates keys, certificate and identity files used in public key cryptography. These files are used for cookie encryption, digital signature and challenge/response identification algorithms compatible with the Internet standard security infrastructure.

By default, files are not encrypted by ntp-keygen. The **-p** *password* option specifies the write password and **-q** *password* option the read password for previously encrypted files. The **ntp-keygen** program prompts for the password if it reads an encrypted file and the password is missing or incorrect. If an encrypted file is read successfully and no write password is specified, the read password is used as the write password by default.

The **ntpd** configuration command crypto **pw** password specifies the read password for previously encrypted files. The daemon expires on the spot if the password is missing or incorrect. For convenience, if a file has been previously encrypted, the default read password is the name of the host running the program. If the previous write password is specified as the host name, these files can be read by that host with no explicit password.

All files are in PEM-encoded printable ASCII format, so they can be embedded as MIME attachments in mail to other sites and certificate authorities. File names begin with the prefix `ntpkey_` and end with the postfix `_hostname.filestamp`, where hostname is usually the string returned by the UNIX **gethostname()** routine, and filestamp is the NTP seconds when the file was generated, in decimal digits. This both guarantees uniqueness and simplifies maintenance procedures, since all files can be quickly removed by a **rm ntpkey*** command or all files generated at a specific time can be removed by a **rm *filestamp** command. To further reduce the risk of misconfiguration, the first two lines of a file contain the file name and generation date and time as comments.

All files are installed by default in the keys `/usr/local/etc` directory , which is normally in a shared filesystem in NFS-mounted networks. The actual location of the keys directory and each file can be overridden by configuration commands, but this is not recommended. Normally, the files for each host are generated by that host and used only by that host, although exceptions exist as noted later on this page.

Normally, files containing private values, including the host key, sign key and identification parameters, are permitted root read/write-only; while others containing public values are permitted world readable. Alternatively, files containing private values can be encrypted and these files permitted world readable, which simplifies maintenance in shared file systems. Since uniqueness is insured by the hostname and file name extensions, the files for a NFS server and dependent clients can all be installed in the same shared directory.

The recommended practice is to keep the file name extensions when installing a file and to install a soft link from the generic names specified elsewhere on this page to the generated files. This allows new file generations to be activated simply by changing the link. If a link is present, **ntpd** follows it to the file name to extract the filestamp. If a link is not present, **ntpd** extracts the filestamp from the file itself. This allows clients to verify that the file and generation times are always current. The **ntp-keygen** program uses the same extension for all files generated at one time, so each generation is distinct and can be readily recognized in monitoring data.

Running the program

The safest way to run the ntp-keygen program is logged in directly as root. The recommended procedure is change to the keys directory, usually `/ust/local/etc`, then run the program. When run for the first time, or if all **ntpkey** files have been removed, the program generates a RSA host key file and matching RSA-MD5 certificate file, which is all that is necessary in many cases. The program also generates soft

links from the generic names to the respective files. If run again, the program uses the same host key file, but generates a new certificate file and link.

The host key is used to encrypt the cookie when required and so must be RSA type. By default, the host key is also the sign key used to encrypt signatures. When necessary, a different sign key can be specified and this can be either RSA or DSA type. By default, the message digest type is MD5, but any combination of sign key type and message digest type supported by the OpenSSL library can be specified, including those using the MD2, MD5, SHA, SHA1, MDC2 and RIPE160 message digest algorithms. However, the scheme specified in the certificate must be compatible with the sign key. Certificates using any digest algorithm are compatible with RSA sign keys; however, only SHA and SHA1 certificates are compatible with DSA sign keys.

Private/public key files and certificates are compatible with other OpenSSL applications and very likely other libraries as well. Certificates or certificate requests derived from them should be compatible with extant industry practice, although some users might find the interpretation of X509v3 extension fields somewhat liberal. However, the identification parameter files, although encoded as the other files, are probably not compatible with anything other than Autokey.

Running the program as other than root and using the UNIX su command to assume root may not work properly, since by default the OpenSSL library looks for the random seed file .rnd in the user home directory. However, there should be only one .rnd, most conveniently in the root directory, so it is convenient to define the $RANDFILE environment variable used by the OpenSSL library as the path to /.rnd.

Installing the keys as root might not work in NFS-mounted shared file systems, as NFS clients may not be able to write to the shared keys directory, even as root. In this case, NFS clients can specify the files in another directory such as /etc using the keysdir command. There is no need for one client to read the keys and certificates of other clients or servers, as these data are obtained automatically by the Autokey protocol.

Ordinarily, cryptographic files are generated by the host that uses them, but it is possible for a trusted agent (TA) to generate these files for other hosts; however, in such cases files should always be encrypted. The subject name and trusted name default to the hostname of the host generating the files, but can be changed by command line options. It is convenient to designate the owner name and trusted name as the subject and issuer fields, respectively, of the certificate. The owner name is also used for the host and sign key files, while the trusted name is used for the identity files.

Flags

Item	Description
-c [*RSA-MD2* | *RSA-MD5* | *RSA-SHA* | *RSA-SHA1* | *RSA-MDC2* | *RSA-RIPEMD160* | *DSA-SHA* | *DSA-SHA1*]	Selects certificate message digest/signature encryption scheme. Note that RSA schemes must be used with a RSA sign key and DSA schemes must be used with a DSA sign key. The default without this option is RSA-MD5.
-d	Enables debugging. This option displays the cryptographic data produced in eye-friendly billboards.
-e	Writes the IFF client keys to the standard output. This is intended for automatic key distribution by mail.
-G	Generates parameters and keys for the GQ identification scheme, obsoleting any that may exist.
-g	Generates keys for the GQ identification scheme using the existing GQ parameters. If the GQ parameters do not yet exist, create them first.
-H	Generates new host keys, obsoleting any that may exist.
-I	Generates parameters for the IFF identification scheme, obsoleting any that may exist.

Item	Description
-i name	Sets the subject name to name. This is used as the subject field in certificates and in the file name for host and sign keys.
-M	Generates MD5 keys, obsoleting any that may exist.
-m *modulus*	Sets prime modulus size in bits (256 - 2048). Default size is 512.
-P	Generates a private certificate. By default, the program generates public certificates.
-p *password*	Encrypts generated files containing private data with password and the DES-CBC algorithm.
-q *password*	Sets the password for reading files to password.
-S *[RSA | DSA]*	Generates a new sign key of the designated type, obsoleting any that may exist. By default, the program uses the host key as the sign key.
-s *name*	Sets the issuer name to name. This is used for the issuer field in certificates and in the file name for identity files.
-T	Generates a trusted certificate. By default, the program generates a non-trusted certificate.
-V *nkeys*	Generates parameters and keys for the Mu-Varadharajan (MV) identification scheme.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must be part of the system group to run this command.

Auditing Events: N/A

Examples

1. To generate RSA-SHA cryptographic keys, enter:

```
ntp-keygen -c RSA-SHA
```

2. To print a list of the peers known to the server as well as a summary of their state, enter:

```
ntpdc -p
```

Output similar to the following appears:

```
            Using OpenSSL version 90804f
            Generating RSA keys (512 bits)...
            RSA                                             3 1 2
            Generating new host file and link
            ntpkey_host_aixfvt12->ntpkey_RSAkey_aixfvt12.3444540821
            Using host key as sign key
```

```
Generating certificate RSA-SHA
X509v3 Basic Constraints: critical,CA:TRUE
X509v3 Key Usage: digitalSignature,keyCertSign
Generating new cert file and link
ntpkey_cert_aixfvt12->ntpkey_RSA-SHAcert_aixfvt12.3444540821
```

Files

Item	Description
/usr/sbin/ntp4/ntp-keygen4	Contains the **ntp-keygen** command. The default symbolic link to the NTP version 4 binary from `/usr/sbin` directory. `/usr/sbin/ntp-keygen --> /usr/sbin/ntp4/ntp-keygen4`

ntp_ssw Command

Purpose

Switches the symbolic links for Network Time Protocol (NTP) between NTP version 3 and NTP version 4.

Syntax

```
ntp_ssw [-v4|-v3]
```

Description

For NTP version 3, the `xntpd` and NTP executable files are linked to the NTP version 3 files. To switch to NTP version 4, all NTP executable files must be linked to the NTP version 4 files. You can switch between NTP version 3 and NTP version 4 by using the **ntp_ssw** command.

When the AIX LPAR is running with NTP version 3, the symbolic links are as follows:

```
/usr/sbin/xntpdc -> /usr/sbin/ntp3/xntpdc
/usr/sbin/ntpq -> /usr/sbin/ntp3/ntpq
/usr/sbin/ntptrace -> /usr/sbin/ntp3/ntptrace
/usr/sbin/ntpdate -> /usr/sbin/ntp3/ntpdate
/usr/sbin/sntp -> /usr/sbin/ntp3/sntp
```

To switch to NTP version 4, run the **ntp_ssw** command with the **-v4** flag. After you run the **ntp_ssw** command, the NTP symbolic links are updated as follows:

```
/usr/sbin/ntpq -> /usr/sbin/ntp4/ntpq4
/usr/sbin/sntp -> /usr/sbin/ntp4/sntp4
/usr/sbin/ntptrace -> /usr/sbin/ntp4/ntptrace4
/usr/sbin/xntpd -> /usr/sbin/ntp4/ntpd4
/usr/sbin/ntpdate -> /usr/sbin/ntp4/ntpdate4
/usr/sbin/ntp-keygen -> /usr/sbin/ntp4/ntp-keygen4
```

Flags

-v4
Switches the NTP symbolic links to NTP version 4 binary files.

-v3
Switches the NTP symbolic links to NTP version 3 binary files.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

- To switch the NTP symbolic links from NTP version 3 to NTP version 4, enter the following command:

```
/usr/sbin/ntp_ssw -v4
```

- To switch the NTP symbolic links from NTP version 4 to NTP version 3, enter the following command:

```
/usr/sbin/ntp_ssw -v3
```

Files

/usr/sbin/ntp_ssw
Contains the **ntp_ssw** command.

ntpq Command

Purpose

Starts the standard Network Time Protocol (NTP) query program.

Syntax

ntpq [**-i**] [**-n**] [**-p**] [**-c** *SubCommand*] [*Host ...*]

Description

The **ntpq** command queries the NTP servers running on the hosts specified which implement the recommended NTP mode 6 control message format about current state and can request changes in that state. It runs either in interactive mode or by using command-line arguments. You can make requests to read and write arbitrary variables, and raw and formatted output options are available. The **ntpq** command can also obtain and print a list of peers in a common format by sending multiple queries to the server.

If you enter the **ntpq** command with one or more flags, the NTP servers running on each of the hosts specified (or defaults to local host) receive each request. If you do not enter any flags, the **ntpq** command tries to read commands from standard input and run them on the NTP server running on the first host specified or on the local host by default. It prompts for subcommands if standard input is the terminal.

The **ntpq** command uses NTP mode 6 packets to communicate with the NTP server and can query any compatible server on the network which permits it.

The **ntpq** command makes one attempt to retransmit requests, and will time-out requests if the remote host does not respond within a suitable time.

Specifying a flag other than **-i** or **-n** sends the queries to the specified hosts immediately. Otherwise, the **ntpq** command attempts to read interactive format subcommands from standard input.

Flags

Item	Description
-c *SubCommand*	Specifies an interactive format command. This flag adds *SubCommand* to the list of commands to run on the specified hosts. You can enter multiple **-c** flags.
-i	Specifies interactive mode. Standard output displays prompts and standard input reads commands.
-n	Displays all host addresses in dotted decimal format (x.x.x.x) rather than the canonical host names.
-p	Displays a list of the peers known to the server and a summary of their state. Same as using the **peers** subcommand.

Parameters

Item	Description
Host ...	Specifies the hosts.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must be part of the system group to run this command.

Auditing Events: N/A

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To start the Network Time Protocol query program in interactive mode, type:

```
ntpq -i
```

2. To add a time interval of 1000 milliseconds to timestamps, type:

```
ntpq -c "delay 1000"
```

ntpq Internal Subcommands

The following subcommands can only be used while running the **ntpq** query program.

Interactive Format Subcommands

Interactive format subcommands consist of a keyword followed by zero to four arguments. You only need to type enough characters of the full keyword to uniquely identify the subcommand. The output of a

subcommand goes to standard output, but you can redirect the output of individual subcommands to a file by appending a > (greater than sign), followed by a file name, to the command line.

Some interactive format subcommands run entirely within the **ntpq** query program and do not result in sending NTP mode 6 requests to a server.

The data carried by NTP mode 6 messages consists of a list of items of the form:

```
Variable=Value
```

where *Value* is ignored, and can be omitted, in requests to the server to read variables. The **ntpq** query program maintains an internal list where data to be included in control messages can be assembled and sent using the **readlist** and **writelist** control message subcommands.

Item	Description
? [*SubCommand*]	Displays command usage information. When used without *SubCommand*, displays a list of all the **ntpq** command keywords. When used with *SubCommand*, displays function and usage information about the subcommand.
addvars *Variable* [=*Value*] [,...]	Specifies the variables and their optional values to be added to the internal data list. If adding more than one variable, the list must be separated by commas and not contain spaces.
authenticate yes | **no**	Specifies whether to send authentication with all requests or not. Normally the **ntpq** query program does not authenticate requests unless they are write requests.
clearvars	Removes all variables from the internal data list.
cooked	Displays all results received from the remote server reformatted. A trailing ? (question mark) marks variables that do not have decodable values.
debug more | **less** | **off**	Turns the **ntpq** query program debugging on or off. The **more** and **less** options control the verbosity of the output. If you enter this subcommand without an argument, it prints the current setting for this subcommand.
delay *MilliSeconds*	Specifies the time interval to add to timestamps included in requests which require authentication. This subcommand enables unreliable server reconfiguration over long delay network paths or between machines whose clocks are unsynchronized. If you enter this subcommand without an argument, it prints the current setting for this subcommand.
host *HostName*	Specifies the host to send queries to. *HostName* may be either a host name or a numeric address. If you enter this subcommand without an argument, it prints the current setting for this subcommand.
hostnames yes | **no**	Specifies whether to output the host name (**yes**) or the numeric address (**no**). Defaults to **yes** unless the **-n** flag is used. If you enter this subcommand without an argument, it prints the current setting for this subcommand.
keyid *Number*	Specifies the server key number to use to authenticate configuration requests. If you enter this subcommand without an argument, it prints the current setting for this subcommand.
ntpversion 1 | **2** | **3**	Specifies the NTP version implementation to use when polling its packets. The default is 3. If you enter this subcommand without an argument, it prints the current setting for this subcommand. **Note:** Mode 6 control messages and modes did not exist in NTP version 1.
passwd	Prompts you to type in the NTP server authentication password to use to authenticate configuration requests.
quit	Exits the **ntpq** query program.
raw	Displays all results received from the remote server without formatting. Only transforms non-ascii characters into printable form.
rmvars *Variable* [=*Value*] [,...]	Specifies the variables and their optional values to be removed from the internal data list. If removing more than one variable, the list must be separated by commas and not contain spaces.
timeout *MilliSeconds*	Specifies the time-out period for responses to server queries. The default is 5000 milliseconds. If you enter this subcommand without an argument, it prints the current setting for this subcommand. **Note:** Because **ntpq** query program retries each query once after a time-out, the total waiting time for a time-out is twice the time-out value set.

Control Message Subcommands

Each peer known to an NTP server has a 16-bit integer association identifier assigned to it. NTP control messages which carry peer variables must identify the peer that the values correspond to by including its association ID. An association ID of 0 is special and indicates the variables are system variables whose names are drawn from a separate name space.

The **ntpq** control message subcommands result in one or more NTP mode 6 messages sent to the server, and outputs the data returned in some format. Most subcommands currently implemented send a single message and expect a single response. The current exceptions are the **peers** subcommand, which sends a preprogrammed series of messages to obtain the data it needs, and the **mreadlist** and **mreadvar** subcommands, which iterate over a range of associations.

Item	Description
associations	Obtains and prints a list of association identifiers and peer statuses for in-spec peers of the server being queried. The list is printed in the following columns: • First column contains the index numbering the associations from 1 for internal use. • Second column contains the actual association identifier returned by the server. • Third column contains the status word for the peer. • Remaining columns contain data decoded from the status word. **Note:** The data returned by the **associations** subcommand is cached internally in the **ntpq** query program. When dealing with servers that use difficult association identifiers, use the index as an argument, in the form `&index`, as an alternative to the association identifier.
clockvar [*AssocID*] [*Variable* [=*Value*], ...] or **cv** [*AssocID*] [*Variable* [=*Value*], ...]	Displays a list of the server's clock variables. Servers which have a radio clock or other external synchronization respond positively to this. To request the system clock variables, leave *AssocID* blank or type `0`. If the server treats clocks as pseudo-peers and can possibly have more than one clock connected at once, referencing the appropriate peer association ID shows the variables of a particular clock. Omitting the variable list causes the server to return a default variable display.
lassociations	Displays a list of association identifiers and peer statuses for all associations for which the server is maintaining state. This subcommand differs from the **associations** subcommand only for servers which retain state for out-of-spec client associations.
lpassociations	Displays data for all associations, including out-of-spec client associations, from the internally cached list of associations.
lpeers	Displays a summary of all associations the server maintains state for Similar to the **peers** subcommand. This may produce a longer list of peers from out-of-spec client servers.
mreadvar *AssocID AssocID* [*Variable* [=*Value*], ...] or **mrv** *AssocID AssocID* [*Variable* [=*Value*], ...]	Displays the values of the specified peer variables for each server in the range of given nonzero association IDs. The association list cached by the most recent associations command determines the range.
mreadlist *AssocID AssocID* or **mrl** *AssocID AssocID*	Displays the values of the specified peer variables in the internal variable list for each server in the range of given nonzero association IDs. The association list cached by the most recent associations command determines the range.
opeers	An old form of the **peers** subcommand. Replaces the reference ID with the local interface address.
passociations	Displays association data concerning in-spec peers from the internally cached list of associations. This subcommand works like the **associations** subcommand except that it displays the internally stored data rather than making a new query.

Item	Description
peers	Displays a list of in-spec peers of the server and a summary of each peer's state. Summary information includes the following: • Address of the remote peer • Reference ID (0.0.0.0 for an unknown reference ID) • Stratum of the remote peer (a stratum of 16 indicates the remote peer is unsynchronized) • Type of peer (local, unicast, multicast, *or* broadcast) • Time the last packet was received, the polling interval (seconds) • Polling interval (seconds) • Reachability register (octal) • Current estimated delay, offset and dispersion of the peer (milliseconds) The character in the left margin indicates the fate of this peer in the clock selection process: **space** Discarded due to high stratum and/or failed sanity checks. **x** Designated falseticker by the intersection algorithm. **.** Culled from the end of the candidate list. **-** Discarded by the clustering algorithm. **+** Included in the final selection set. **#** Selected for synchronization but distance exceeds maximum. ***** Selected for synchronization. **o** Selected for synchronization, **pps** signal in use. The contents of the host field may be a host name, an IP address, a reference clock implementation name with its parameter or REFCLK (*ImplementationNumber*, *Parameter*). Only IP addresses display when using **hostnames no**. **Note:** The **peers** subcommand depends on the ability to parse the values in the responses it gets. It may fail to work from time to time with servers that poorly control the data formats. The **peers** subcommand is non-atomic and may occasionally result in spurious error messages about invalid associations occurring and terminating the command.
pstatus *AssocID*	Displays the names and values of the peer variables of the server with the given association by sending a read status request. The output displays the header preceding the variables, both in hexadecimal and in English.

Item	Description
readlist [*AssocID*] or **rl** [*AssocID*]	Displays the values of the peer variables in the internal variable list of the server with the given association. To request the system variables, leave *AssocID* blank or type `0`. If the internal variable list is empty, the server returns a default variable display.
readvar [*AssocID*] [*Variable* [=*Value*], ...] or **rv** [*AssocID*] [*Variable* [=*Value*], ...]	Displays the values of the specified peer variables of the server with the given association by sending a read variables request. To request the system variables, leave *AssocID* blank or type `0`. Omitting the variable list causes the server to return a default variable display.
writevar [*AssocID*] [*Variable* [=*Value*], ...]	Writes the values of the specified peer variables to the server with the given association by sending a write variables request.
writelist [*AssocID*]	Writes the values of the peer variables in the internal variable list of the server with the given association.

Files

Item	Description
/usr/sbin/ntpq	Contains the **ntpq** command.

ntpq4 Daemon

Purpose

Starts the standard Network Time Protocol (NTP) query program.

Syntax

ntpq [**-4** **-6** **-d** **-i** **-n** **-p**] [**-c** command] [host] [...]

Description

The **ntpq** program is used to monitor the NTP daemon, the **ntpd** operations, and determine performance. It uses the standard NTP version 3 mode 6 control message formats defined by RFC 1305. The same formats are used in NTP version 4.

The program can be run either in interactive or controlled mode using command line arguments. The raw and printed output options enables you to assemble the requests to read and write arbitrary variables. The **ntpq** program can also obtain and print a list of peers in a common format by sending multiple queries to the server.

If one or more request options are included in the command line when the **ntpq** program is executed, each request will be sent to the NTP servers running on the hosts given by the command line arguments, or on localhost by default. If no request options are given, the **ntpq** utility will attempt to read commands from the standard input and execute them on the NTP server running on the first host given by the command line, again defaulting to localhost when no other host is specified. The **ntpq** utility will prompt for commands if the standard input is a terminal device.

The **ntpq** utility uses NTP mode 6 packets to communicate with the NTP server, and hence can be used to query any compatible server on the network which permits it.

In the instance where a host name is expected, and when you add a **-4** qualifier preceding the host name, the utility forces the DNS resolution to the IP version 4 namespace. Similarly, and a **-6** qualifier forces DNS resolution to the IP version 6 namespace.

Specifying a command line option other than **-i** or **-n** will cause the specified query or queries to be sent to the indicated host or hosts immediately. Otherwise, the **ntpq** utility will attempt to read interactive format commands from the standard input.

Flags

Item	Description
-4	Forces DNS resolution of the host names on the command line to the IP version 4 namespace.
-6	Forces DNS resolution of the host names on the command line to the IP version 6 namespace.
-c	The following argument is interpreted as an interactive format command and is added to the list of commands to be executed on the specified host or hosts. Multiples of the **-c** options might be added.
-d	Enables the debugging mode.
-i	Forces the **ntpq** utility to operate in interactive mode. The results will be written to the standard output and the commands are read from the standard input.
-n	Outputs all host addresses in dotted-quad numeric format rather than converting to the canonical host names.
-p	Prints a list of peers known to the server as well as a summary of their state. This is equivalent to the peers interactive command.

Parameters

Item	Description
Host...	Specifies the hosts.

Exit Status

This command returns the following exit values:

0
Successful completion.

> 0
An error occurred.

Security

Access Control : You must have root authority to run this command.

Auditing Events : N/A

Examples

1. To start the Network Time Protocol query program in interactive mode, enter:

   ```
   ntpq -i
   ```

2. To print a list of peers known to the server and the summary of their state, enter:

   ```
   ntpq -p
   ```

Output similar to the following is displayed:

```
     remote           refid             st  t   when poll reach   delay        offset    jitter
==============================================================================================
ausgsa.austin.ibm.com 9.41.253.167     2 u   19      64  377      285.962   -8.792   2.989
```

ntpq Internal Commands

Interactive Format Commands

Interactive format commands consist of a keyword followed by a maximum of 4 arguments. You must type only the required number of characters of the keyword to uniquely identify the command. The output of a command is normally sent to the standard output. You can also opt to send the output of individual commands by appending a greater than symbol (>), followed by a file name, to the command line. A number of interactive format commands are executed entirely within the **ntpq** program and do not send the NTP mode 6 requests to a server.

Item	Description
? [command_keyword] or **help [command_keyword]**	A question mark (?) by itself will print a list of all the command keywords known to this incarnation of ntpq. A question mark (?) followed by a command keyword will print function and the use of the command.
addvars variable_name [= value] [...] or **rmvars variable_name [...]** or **clearvars**	The data carried by the NTP mode 6 messages consists of a list of items of the form *variable_name = value*, where the equals symbol (=) value is ignored, and can be omitted, in requests to the server to read variables. The **ntpq** program maintains an internal list in which data to be included in control messages can be assembled, and sent using the **readlist** and **writelist** commands described below. The **addvars** command allows variables and their optional values to be added to the list. If more than one variable is to be added, the list must be separated by using commas, and must not contain any white space. The **rmvars** command can be used to remove individual variables from the list, while the **clearlist** command removes all variables from the list.
cooked	Causes the output from query commands to be cooked, so that variables which are recognized by the **ntpq** command will have their values reformatted for human consumption. The **ntpq** program marks the variables with a trailing question mark symbol (?) when the variable value cannot be decoded.
debug more | less | no	Adjusts the level of **ntpq** debugging. The default is **debug no**.
delay milliseconds	Specifies a time interval to be added to timestamps included in requests which require authentication. This is used to enable server reconfiguration over long delay network paths or between machines whose clocks are not synchronized.
host hostname	Sets the host to which future queries will be sent. Hostname may be either a host name or a numeric address.
hostnames [yes | no]	If **yes** is specified, host names are printed in the information display. If **no** is specified, numeric addresses are printed. The default is yes, unless modified using the command line **-n** switch.
keyid keyid	Specifies the key number to be used to authenticate configuration requests. This must correspond to a key number the server has been configured to use for this purpose.

Item	Description
ntpversion 1 | 2 | 3 | 4	Sets the NTP version number which **ntpq** claims in packets. The default is 2. The mode 6 control messages did not exist in NTP version 1.
passwd	Prompts for a password that will not be echoed, and which will be used to authenticate configuration requests. The password must correspond to the key configured for NTP server for this purpose.
quit	Exits **ntpq**.
raw	Prints the output of query commands received from the remote server. The only formatting done on the data is transforming non-ASCII data to a printable form.
timeout millseconds	Specifies a timeout period for responses to server queries. The default is 5000 milliseconds. Since **ntpq** retries each query once after a timeout, the total waiting time for a timeout will be twice the timeout value set.

Control Message Commands

Each association known to an NTP server has a 16 bit integer association identifier. The NTP control messages that carry peer variables must identify the corresponding peer values, which are its association ID. An association ID 0 indicates that the variables are system variables, and their names are drawn from a separate name space.

Control message commands result in one or more NTP mode 6 messages being sent to the server, and cause the data returned to be printed in a format. Most commands currently implemented send a single message and expect a single response. The current exceptions is the **peers** command, which will send a pre-programmed series of messages to obtain the data it needs, and the **mreadlist** and **mreadvar** commands, which will iterate over a range of associations.

Item	Description
associations	Obtains and prints a list of association identifiers and peer statuses for in-spec peers of the server being queried. The list is printed in columns. The first column indicates the index numbering of associations from 1. The second column specifies the actual association identifier returned by the server, and the third column indicates the status word for the peer. This is followed by a number of columns containing data decoded from the status word. See the peers command for a decode of the condition field. **Note:** 1. The data returned by the associations command is cached internally in **ntpq**. 2. The index in the form **&index** is used when dealing with servers that use association identifiers wherein the subsequent commands require an association identifier as an argument.

Item	Description
clockvar [assocID] [variable_name [= value [...]] [...] **cv [assocID] [variable_name [= value [...]][...]**	Requests the server to send a list of the server's clock variables. Servers, which have a radio clock or other external synchronization will respond positively to this. If the association identifier is omitted or is a zero, you are requesting for the system clock variables and will get a positive response from all servers with a clock. If the server treats clocks as pseudo-peers, and hence can possibly have more than one clock connected at once, referencing the appropriate peer association ID will show the variables of a particular clock. Omitting the variable list will cause the server to return a default variable display.
lassociations	Obtains and prints a list of association identifiers and peer statuses for all associations for which the server is maintaining state. This command differs from the associations command only for servers which retain state for out-of-spec client associations (i.e., fuzzballs). Such associations are normally omitted from the display when the associations command is used, but are included in the output of lassociations.
lpassociations	Prints data for all associations, including out-of-spec client associations, from the internally cached list of associations. This command differs from passociations only when dealing with fuzzballs.
lpeers	Similar to R peers, except that a summary of states of all associations that the server is maintaining are printed. This can produce a much longer list of peers from fuzzball servers.
mreadlist assocID assocID **mrl assocID assocID**	Similar to the **readlist** command, except that the query is done for a range of (nonzero) association IDs. This range is determined from the association list cached by the most recent associations command.
mreadvar assocID assocID [variable_name [= value[...] **mrv assocID assocID [variable_name [= value[...]**	Similar to the **readvar** command, except that the query is done for a range of (nonzero) association IDs. This range is determined from the association list cached by the most recent associations command.
opeers	An old form of peers command with the reference ID replaced by the local interface address.
passociations	Displays association data concerning in-spec peers from the internally cached list of associations. This command performs identically to the associations except that it displays the internally stored data rather than making a new query.
peers	Obtains a current list peers of the server, along with a summary of each peer's state. Summary information includes the address of the remote peer, the reference ID (0.0.0.0 if this is unknown), the stratum of the remote peer, the type of the peer (local, unicast, multicast or broadcast), when the last packet was received, the polling interval, in seconds, the reachability register, in octal, and the current estimated delay, offset and dispersion of the peer, all in milliseconds.
pstatus assocID	Sends a read status request to the server for the given association. The names and values of the peer variables returned will be printed. Note that the status word from the header is displayed preceding the variables, both in hexadecimal and in pidgeon English.

Item	Description
readlist [assocID] **rl [assocID]**	Requests that the values of the variables in the internal variable list be returned by the server. If the association ID is omitted or is 0 the variables are assumed to be system variables. Otherwise they are treated as peer variables. If the internal variable list is empty a request is sent without data, which should induce the remote server to return a default display.
readvar assocID variable_name [= value] [...] **rv assocID [variable_name [= value] [...]**	Requests that the values of the specified variables be returned by the server by sending a read variables request. If the association ID is omitted or is given as zero the variables are system variables, otherwise they are peer variables and the values returned will be those of the corresponding peer. Omitting the variable list will send a request with no data which should induce the server to return a default display. The encoding and meaning of the variables derived from NTPv3 is given in RFC-1305; the encoding and meaning of the additional NTPv4 variables are given later in this page.
writevar assocID variable_name [= value [...]	Similar to the **readvar** request, except that the specified variables are written.
writelist [assocID]	Similar to the **readlist** request, except that the internal list of variables are written.

Files

Item	Description
`/usr/sbin/ntp4/ntpq4`	Contains the **ntpq** command. The default symbolic link to NTP version 3 binary file from the `/usr/sbin` directory. `/usr/sbin/ntpq --> /usr/sbin/ntp3/ntpq`

ntptrace Command

Purpose

Traces a chain of Network Time Protocol (NTP) hosts back to their master time source.

Syntax

ntptrace [**-d**] [**-n**] [**-v**] [**-r** *Retries*] [**-t** *TimeOut*] [*Server*]

Description

The **ntptrace** command determines where a given NTP server gets its time, and follows the chain of NTP servers back to their master time source. For example, stratum 0 server.

Flags

Item	Description
-d	Turns on debugging output.
-n	Outputs host IP addresses instead of host names.
-r *Retries*	Specifies the number of retransmission attempts for each host. The default is 5.

Item	Description
-t *TimeOut*	Specifies the retransmission timeout in seconds. The default is 2 seconds.
-v	Specifies verbose mode.

Parameters

Item	Description
Server	Specifies the server. The default is the local host.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must be part of the system group to run this command.

Auditing Events: N/A

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To trace where the local host NTP server gets its time from, enter:

```
ntptrace
```

Output similar to the following appears:

```
localhost: stratum 4, offset 0.0019529, sync distance 0.144135
server2.bozo.com: stratum 2, offset 0.0124263, sync distance 0.115784
usndh.edu: stratum 1, offset 0.0019298, sync distance 0.011993, refid
'WWVB'
```

On each line, the fields are:

1. the host's stratum,
2. the time offset between that host and the local host, as measured by the **ntptrace** command, (this is why it is not always zero for localhost).
3. the host's synchronization distance, which is a measure of the quality of the clock's time, and
4. the reference clock ID This only applies to stratum-1 servers.

All times are given in seconds.

Files

Item	Description
/usr/sbin/ntptrace	Contains the **ntptrace** command.

ntptrace4 Command

Purpose

Traces a chain of Network Time Protocol (NTP) hosts back to their master time source.

Syntax

ntptrace [**-n**] [**server**]

Description

The **ntptrace** command determines the time source for the Network Time Protocol (NTP) server and follows the chain of NTP servers back to their master time source. If no arguments are provided, it starts with localhost. Following is an example of the output of the **ntptrace** command:

```
% ntptrace
localhost: stratum 4, offset 0.0019529, sync distance 0.144135
server2ozo.com: stratum 2, offset 0.0124263, sync distance 0.115784
usndh.edu: stratum 1, offset 0.0019298, sync distance 0.011993, refid 'WWVB'
```

On each line, the fields from left to right are host name, host stratum, time offset between that host and the local host as measured by the **ntptrace** command. This is why it is not always zero for "localhost", host synchronization distance, and (applies only for the stratum-1 servers) the reference clock ID. All times are given in seconds. Note that the stratum is the server hop count to the primary source, while the synchronization distance is the estimated error relative to the primary source. These terms are precisely defined in RFC-1305.

Flags

Item	Description
-n	Turns off the printing of host names; instead, host IP addresses are printed. This may be useful if a nameserver is down.

Parameters

Item	Description
Server	Specifies the server. The default is the local host.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion
>0	An error occurred

Security

Access Control: You must be part of the system group to run this command.

Auditing Events: N/A

Examples

1. To trace the time source for the local host NTP server, enter:

```
ntptrace
```

Output similar to the following appears:

```
            loopback: stratum 5, offset 0.000076, synch distance 0.18291
            ganga08.in.ibm.com: stratum 4, offset -0.001854, synch distance 0.30600
            ganga10.in.ibm.com: stratum 3, offset 0.000251, synch distance 0.30550
            ausgsa.austin.ibm.com: stratum 2, offset -0.010158, synch distance 0.01921
            gsantp.austin.ibm.com: stratum 1, offset 0.016067, synch distance 0.00000, refid
          'GPS'
```

Files

Item	Description
/usr/sbin/ntp4/ntptrace4	Contains the **ntptrace** command. Default Symbolic link to NTP version 3 binary from `/usr/sbin` directory. `/usr/sbin/ntptrace --> /usr/sbin/ntp3/ntptrace`

nulladm Command

Purpose

Creates active accounting data files.

Syntax

/usr/sbin/acct/nulladm [*File ...*]

Description

The **nulladm** command creates the file specified by the *File* parameter, gives read (r) and write (w) permission to the file owner, and group and read (r) permission to other users, and ensures that the file owner and group are **adm**. Various accounting shell procedures call the **nulladm** command. A user with administrative authority can use this command to set up the active data files, such as the **/var/adm/wtmp** file.

> **Note:** You should not share accounting files among nodes in a distributed environment. Each node should have its own copy of the various accounting files.

Security

Access Control: This command should grant execute (x) access only to members of the adm group.

Files

Item	Description
/usr/sbin/acct	Contains the accounting commands.
/var/adm/acct/sum	Contains accounting data files.

number Command

Purpose

Displays the written form of a number.

Syntax

number

Description

The **number** command translates the numerical representation of an entered number to the written form. The largest number it can translate accurately contains 66 digits. For example:

```
12345678
twelve million.
three hundred forty five thousand.
six hundred seventy eight.
```

In the above example, you entered 12345678 and the computer translated it to `twelve million three hundred forty five thousand six hundred seventy eight`.

The **number** command does not prompt you for a number. Once started, it simply waits for input. To exit the program, press the Interrupt (Ctrl-C) or End Of File **(**Ctrl-D) key sequence.

Files

Item	Description
/usr/games	Contains the system games.

nxstat Command

Purpose

The **nxstat** command displays usage statistics of on-chip hardware accelerators that can implement GZIP functions.

Syntax

```
nxstat {-S | [-AUWT] [-a accel_type] [-u unit_index ] [-w window_index] [-t window_type]
interval [count]}
```

Description

Applications in an AIX logical partition use the `zlibNX` library to open a communication channel with the GZIP accelerator. This communication channel is also known as a *window*. For more information about the type of communication channels, see Nest Accelerators.

The low-level usage statistics per window might include a large amount of data that might not be relevant for a system administrator. The **nxstat** command can aggregate the usage statistics to provide global usage statistics for a specific accelerator type or for a specific hardware accelerator unit based on the options and arguments that are specified with the command.

The following values are displayed in the usage statistics for each aggregation level along with the window type and corresponding credits:

- Opens: Number of open and close operations on a window.

- `Bytes`: Number of bytes processed by the hardware accelerator unit.
- `Faults`: Number of page faults that are detected by the hardware accelerator unit.
- `Intrs`: Number of completion interrupts. This value is optional for accelerator operations.
- `Polls`: Number of completion interrupts that are detected during the polling phase of the **nxstat** command.
- `Waits`: Number of operations that were completed after the caller was actually blocked. The total of `Polls` and `Waits` values represent the total number of operations that were completed during the specific time interval.

The kernel service that waits for completion of active GZIP operations, first polls a `completion` bit for a period of time before blocking the thread. Total count of `Polls` and `Waits` values provide insight about efficiency and usefulness of the polling phase.

The **nxstat** command aggregates the usage statistics of a specific accelerator type during a specified time interval. The **nxstat** command can also display the accelerator usage statistics for successive time intervals. The **-A**, **-U**, **-T**, and **-W** flags determine the aggregation levels. If more than one flags are specified, for example, **-AU**, the **nxstat** command displays usage statistics for both aggregation levels. The **-a**, **-u**, **-t**, **-w** flags are used to limit the statistics data to specific objects such as accelerator unit, window, and window type.

Flags

-s

Displays types of accelerators that are available.

-A

Displays usage statistics for a specific type of accelerator. This is the default option.

-U

Displays usage statistics for a specific hardware accelerator unit.

-T

Displays usage statistics for a specific window type.

-W

Displays usage statistics for each window.

-a

Specifies the accelerator type for which the usage statistics is displayed. The **-a** flag value for the GZIP accelerator is 0. Currently, the only supported value for the **-a** flag is 0. Therefore, if you do not specify the **-a** flag, the default value of 0 is applied.

-u

Specifies the accelerator unit for which the usage statistics is displayed. You must specify the **-u** flag only with the **-U** flag. The accelerator unit index value for this flag is a number between zero and the number of accelerator units minus 1. For example, if the AIX logical partition has access to three accelerator units, valid values that can be specified for this flag are 0, 1, and 2.

If the **-u** flag is not specified along with the **-U** flag, the **nxstat** command displays the data for all the hardware accelerator units that can be accessed by the logical partition.

-w

Specifies a specific window for which the usage statistics is displayed. If the logical partition has access to **n** windows, the windows are numbered from 0 to n-1. You can specify the **-w** flag only with the **-W** flag.

-t

Specifies a specific window type (0 for Quality of Service (QoS), 1 for default) for which the usage statistics is displayed. You can specify the **-t** flag only with the **-T** flag.

Note: The **-u**, **-w** and **-t** flags are mutually exclusive.

interval
Specifies the time interval, in seconds, during which the usage statistics must be captured. This flag is mandatory. By default, the kernel service captures usage statistics every 2 seconds, so you must specify a time interval more than 2 seconds to display useful data.

count
Specifies the number of successive time intervals for which you want to display the usage statistics. The default value is 1.

Examples

- To display total usage statistics for all the accelerator type and for all window types, run the following command:

```
nxstat 10 4
```

An output that is similar to the following example is displayed:

```
Accelerator GZIP:   Window Types:  2    Units:  1   Credits:   72
Type Cred   Opens          Bytes   Faults      Intrs         Polls         Waits
 ALL   72  266826      35.747 GB        0      17281        222795         61223
 ALL   72  262812      35.638 GB        0      17524        220368         59641
 ALL   72  300647      33.583 GB        0      15990        257351         58969
 ALL   72  259405      35.557 GB        0      17111        216629         60048

 ** Average **
 ALL   72  272422      35.131 GB        0      16976        229285         59970
```

- To display accelerator level statistics by window type, run the following command:

```
nxstat -AT 10
```

An output that is similar to the following example is displayed:

```
Accelerator GZIP:   Window Types:  2    Units:  4   Credits:   92
Type Cred   Opens          Bytes   Faults      Intrs         Polls         Waits
 QOS   12         0    49.892 GB        0      28695          4998         52476
 DEF   80         0     4.406 GB        0       2561           422          4654
```

- To display unit level statistics by accelerator unit, run the following command:

```
nxstat -U 10 4
```

An output that is similar to the following example is displayed:

```
  Unit Index  0:   SRAD ID         0  Window Types:  2    Credits:   23
Type Cred   Opens          Bytes   Faults      Intrs         Polls         Waits
 ALL   23       0      16.735 GB        0       9589          1551         17728

  Unit Index  1:   SRAD ID         1  Window Types:  2    Credits:   23
Type Cred   Opens          Bytes   Faults      Intrs         Polls         Waits
 ALL   23       0      12.438 GB        0       7219          1270         13058

  Unit Index  2:   SRAD ID         2  Window Types:  2    Credits:   23
Type Cred   Opens          Bytes   Faults      Intrs         Polls         Waits
 ALL   23       0      12.427 GB        0       7131          1310         13006

  Unit Index  3:   SRAD ID         3  Window Types:  2    Credits:   23
Type Cred   Opens          Bytes   Faults      Intrs         Polls         Waits
 ALL   23       0      12.430 GB        0       7230          1383         12935
```

- To display types of accelerators that are available, run the following command:

```
nxstat -S
```

An output that is similar to the following example is displayed:

```
GZIP accelerator available
```

o

The following AIX commands begin with the letter *o*.

od Command

Purpose

Displays files in a specified format.

Syntax

To Display Files Using a Type-String to Format the Output

od [**-v**] [**-A** *AddressBase*] [**-N** *Count*] [**-j** *Skip*] [**-t** *TypeString* ...] [*File* ...]

To Display Files Using Flags to Format the Output

od [**-a**] [**-b**] [**-c**] [**-C**] [**-d**] [**-D**] [**-e**] [**-f**] [**-F**] [**-h**] [**-H**] [**-i**] [**-I**] [**-l**] [**-L**] [**-o**] [**-O**] [**-p**] [**-P**] [**-s**] [**-v**] [**-x**] [**-X**] [[**-S** [*N*]] [**-w** [*N*]] [*File*] [[**+**] *Offset* [**.** | **b** | **B**] [**+**] *Label* [**.** | **b** | **B**]] [*File* ...]

Description

The **od** command displays the file specified by the *File* parameter in the format specified. If the *File* parameter is not given, the **od** command reads standard input. Multiple types can be specified by using multiple -bcCDdFfOoSstvXx options.

In the first syntax format, the output format is specified by the **-t** flag. If no format type is specified, **-t o2** is the default.

In the second syntax format, the output format is specified by a combination of flags. The *Offset* parameter specifies the point in the file where the file output begins. By default, the *Offset* parameter is interpreted as octal bytes. If the **.** (dot) suffix is appended, the parameter is interpreted as a decimal; if the parameter begins with a leading x or 0x, it is treated as a hexadecimal. If the **b** suffix is added to the parameter, it is interpreted in blocks of 512 bytes; if the **B** suffix is added to the parameter, it is interpreted in blocks of 1024 bytes.

The *Label* parameter is interpreted as a pseudo-address for the first byte displayed. If used, it is given in **( )** (parentheses) following the *Offset* parameter. The suffixes have the same meanings as for the *Offset* parameter.

When the **od** command reads standard input, the *Offset* parameter and the *Label* parameter must be preceded by a **+** (plus sign).

The setting of environment variables such as **LANG** and **LC_ALL** affects the operation of the **od** command.

The **od** command copies each input file sequentially, writes them to the standard output and transforms the input data according to the types of output that are specified by using the **-t** flag. The default number of bytes in the input data correspond to the size of the type specifier characters **d, o, u,** or **x**. These type specifier characters allow you to specify an optional number of bytes in the input data that corresponds to the number of bytes in the **char**, **short**, **int**, and **long** data types in the C programming language. The number of bytes can also be specified by using an application (any program or use case that uses the **od** command) in which the character **C** is used for **char** data type, character **S** is used for **short** data type, character **I** is used for **int** data type, and character **L** is used for **long** data type. The type specifier characters supports the values 1, 2, 4 and 8 even though the data type as per the C programming language is not specified for the size of the input data. The type specifier characters also supports the decimal value similar to the **long long** data type in the C programming language.

Flags

The flags for the first format are:

Item	Description
-A *AddressBase*	Specifies the input offset base. The *AddressBase* variable is one of the following characters: **d** Offset base is written in decimal. **o** Offset base is written in octal. **x** Offset base is written in hexadecimal. **n** Offset base is not displayed. Unless **-A n** is specified, the output line will be preceded by the input offset, cumulative across input files, of the next byte to be written. In addition, the offset of the byte following the last byte written will be displayed after all the input data has been processed. Without the **-A** address_base option and the [offset_string] operand, the input offset base is displayed in octal.
-j *Skip*	Jumps over the number of bytes given by the *Skip* variable before beginning to display output. If more than one file is specified, the **od** command jumps over the designated number of bytes of the concatenated input files before displaying output. If the combined input is not at least the length of the skip bytes, the **od** command will write a diagnostic message to standard error and exit non-zero status. By default, the value of the *Skip* variable is interpreted as a decimal number. With a leading 0x or 0X, the offset is interpreted as a hexadecimal number; otherwise, with a leading 0, the offset shall be interpreted as an octal number. If the characters **b**, **k**, or **m** are appended to the number contained by the *Skip* variable, the offset is equal to the value, in bytes, of the *Skip* variable multiplied by 512, 1024, or 1024*1024, respectively.
-N *Count*	Formats no more than the number of input bytes specified by the *Count* variable. By default, the value of the *Count* variable is interpreted as a decimal number. With a leading 0x or 0X, it is treated as a hexadecimal number. If it begins with a 0, it is treated as an octal number. The base of the address displayed is not implied by the base of the *Count* option-argument.

Item	Description
-t *TypeString*	Specifies the output type. The *TypeString* variable is a string specifying the types to be used when writing out data. Multiple types can be concatenated within the same *TypeString* variable, and the **-t** flag can be specified more than once. Output lines are written for each type specified, in the order in which the type specification characters are given. The *TypeString* variable can consist of the following characters: **a** Displays bytes as named characters. Bytes with the least seven bits in the range of 0 through 01777 are written using the corresponding names for those characters. **c** Displays bytes as characters. The number of bytes transformed by the **c** type string is determined by the **LC_CTYPE** local category. Printable multibyte characters are written in the area corresponding to the first byte of the character; the two character sequence ** is written in the area corresponding to each remaining byte in the character, as an indication that the character is continued. The following nongraphic characters are used as C-language escape sequences: **** Backslash **\a** Alert **\b** Backspace **\f** Form-feed **\n** New-line character **\0** Null **\r** Carriage return **\t** Tab **\v** Vertical tab **d** Displays bytes as signed decimals. By default, the **od** command transforms the corresponding number of bytes in the C-language type **int**. The **d** type string can be followed by an unsigned decimal integer that specifies the number of bytes to be transformed by each instance of the output type. An optional **C**, **I**, **L**, or **S** character can be appended to the **d** option, indicating that the conversion should be applied to an item of type **char**, **int**, **long**, or **short**, respectively. **f** Displays bytes as floating points. By default, the **od** command transforms the corresponding number of bytes in the C-language type **double**. The **f** type string can be followed by an unsigned decimal integer that specifies the number of bytes to be transformed by each instance of the output type. An optional **F**, **D**, or **L** character can be appended to the **f** option, indicating that the conversion should be applied to an item of type **float**, **double**, or **long double**, respectively. **o** Displays bytes as octals. By default, the **od** command transforms the corresponding number of bytes in the C-language type **int**. The **o** type string can be followed by an unsigned decimal integer that specifies the number of bytes to be transformed by each instance of the output type. An optional **C**, **I**, **L**, or **S** character can be appended to the **o** option, indicating that the conversion should be applied to an item of type **char**, **int**, **long**, or **short**, respectively.

Item	Description
	u Display bytes as unsigned decimal. By default, the **od** command transforms the corresponding number of bytes in the C-language type **int**. The **u** type string can be followed by an unsigned decimal integer that specifies the number of bytes to be transformed by each instance of the output type. An optional **C**, **I**, **L**, or **S** character can be appended to the **u** option, indicating that the conversion should be applied to an item of type **char**, **int**, **long**, or **short**, respectively.
	x Display bytes as hexadecimal. By default, the **od** command transforms the corresponding number of bytes in the C-language type **int**. The **x** type string can be followed by an unsigned decimal integer that specifies the number of bytes to be transformed by each instance of the output type. An optional **C**, **I**, **L**, or **S** character can be appended to the **x** option, indicating that the conversion should be applied to an item of type **char**, **int**, **long**, or **short**, respectively.
-v	Writes all input data. By default, output lines that are identical to the immediately preceding output lines are not printed, but are replaced with a line containing only an * (asterisk). When the **-v** flag is specified, all the lines are printed.

The flags for the second format are:

Item	Description
-a	Displays bytes as characters and displays them with their ASCII names. If the **-p** flag is also given, bytes with even parity are underlined. The **-P** flag causes bytes with odd parity to be underlined. Otherwise, parity is ignored.
-b	Displays bytes as octal values.
-c	Displays bytes as ASCII characters. The following nongraphic characters appear as C-language escape sequences: **** Backslash **\a** Alert **\b** Backspace **\f** Form-feed **\n** New-line character **\0** Null **\r** Carriage return **\t** Tab **\v** Vertical tab Others appear as three-digit octal numbers.
-C	Displays extended characters as standard printable ASCII characters (using the appropriate character escape string) and displays multibyte characters in hexadecimal form.
-d	Displays 16-bit words as unsigned decimal values.
-D	Displays long words as unsigned decimal values.
-e	Displays long words as double-precision, floating point. (same as the **-F** flag)
-f	Displays long words as floating points.
-F	Displays long words as double-precision, floating point. (same as the **-e** flag)
-h	Displays 16-bit words as unsigned hexadecimal.
-H	Displays long words as unsigned hexadecimal values.

Item	Description
-i	Displays 16-bit words as signed decimal.
-I	(Uppercase i) Displays long words as signed decimal values.
-l	(Lowercase L) Displays long words as signed decimal values.
-L	Displays long words as signed decimal values.
-o	Displays 16-bit words as unsigned octal.
-O	Displays long words as unsigned octal values.
-p	Indicates even parity on **-a** conversion.
-P	Indicates odd parity on **-a** conversion.
-s	Displays 16-bit words as signed decimal values.
-S[*N*]	Searches for strings of characters ending with a null byte. The *N* variable specifies the minimum length string to be recognized. If the *N* variable is omitted, the minimum length defaults to 3 characters.
-v	Writes all input data. By default, output lines that are identical to the immediately preceding output lines are not printed, but are replaced with a line containing only an * (asterisk). When the **-v** flag is specified, all the lines are printed.
-w [*N*]	Specifies the number of input bytes to be interpreted and displayed on each output line. If the **-w** flag is not specified, 16 bytes are read for each display line. If the **-w** flag is specified without the *N* variable, 32 bytes are read for each display line. The maximum input value is 4096 bytes. Input values greater than 4096 bytes will be reassigned the maximum value.
-x	Displays 16-bit words as hexadecimal values.
-X	Displays long words as unsigned hexadecimal values. (same as the **-H** flag)

Note: The flags **-I** (uppercase i), **-l** (lowercase L), and **-L** are identical.

Exit Status

This command returns the following exit values:

Item	Description
0	All input files were processed successfully.
>0	An error occurred.

Examples

1. To display a file in octal, a page at a time, enter:

   ```
   od a.out | pg
   ```

   This command displays the a.out file in octal format and pipes the output through the **pg** command.
2. To translate a file into several formats at once, enter:

   ```
   od -t cx a.out > a.xcd
   ```

   This command writes the contents of the a.out file, in hexadecimal format (**x**) and character format (**c**), into the a.xcd file.
3. To start displaying a file in the middle (using the first syntax format), enter:

```
od -t acx -j 100 a.out
```

This command displays the a.out file in named character (**a**), character (**c**), and hexadecimal (**x**) formats, starting from the 100th byte.

4. To start in the middle of a file (using the second syntax format), enter:

```
od -bcx a.out +100.
```

This displays the **a.out** file in octal-byte (**-b**), character (**-c**), and hexadecimal (**-x**) formats, starting from the 100th byte. The **.** (period) after the offset makes it a decimal number. Without the period, the output would start from the 64th (100 octal) byte.

Files

Item	Description
/usr/bin/od	Contains the **od** command.

odmadd Command

Purpose

Adds objects to created object classes.

Syntax

odmadd [*InputFile* ...]

Description

The **odmadd** command takes as input one or more *InputFile* files and adds objects to object classes with data found in the stanza files. Each *InputFile* file is an ASCII file containing the data that describes the objects to be added to object classes. If no file is specified, input is taken from stdin (standard input).

The classes to be added to are specified in the ASCII input file. The file is in the following general format:

```
class1name:
        descriptor1name = descriptor1value
        descriptor2name = descriptor2value
        descriptor3name = descriptor3value

class2name:
        descriptor4name = descriptor4value
.
.
.
```

The input file can contain the **** (backslash), which is handled as it is in C language. String and method values in the input file must be enclosed in " " (double-quotation marks). A descriptor value can span more than one line.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

An ASCII input file used by the **odmadd** command looks like the following:

```
Fictional_Characters:
Story_Star       = "Cinderella"
Birthday         = "Once upon a time"
Age              = 19
Friends_of       = Cinderella
Enemies_of       = "Cinderella"

Friend_Table:
Friend_of        = "Cinderella"
Friend           = "Fairy godmother"

Friend_Table:
Friend_of        = "Cinderella"
Friend           = "Mice"

Enemy_Table:
Enemy_of         = "Cinderella"
Enemy            = "Wicked sisters"

Enemy_Table:
Enemy_of         = "Cinderella"
Enemy            = "Mean stepmother"
```

If the preceding file is named `NewObjects`, the following command adds the objects to existing object classes:

```
odmadd NewObjects
```

See html

odmchange Command

Purpose

Changes the contents of a selected object in the specified object class.

Syntax

odmchange -o *ObjectClass* [**-q** *Criteria*] [*InputFile*]

Description

The **odmchange** command, given the object class to modify, the search criteria, and the new object (only for attributes that need to change), modifies all objects that satisfy the search criteria. The *InputFile* file has the same format as the *InputFile* file (the ASCII input file) for the **odmadd** command.

Flags

Item	Description
-o *ObjectClass*	Specifies the object class to modify.
-q *Criteria*	Specifies the criteria used to select objects from the object class. For information on qualifying criteria, see html

odmcreate Command

Purpose

Produces the **.c** (source) and **.h** (include) files necessary for ODM application development and creates empty object classes.

Syntax

odmcreate [**-p**] [**-c** | **-h**] *ClassDescriptionFile*

Description

The **odmcreate** command is the ODM class compiler. The command takes as input an ASCII file that describes the objects a user wishes to use in a specific application. The **odmcreate** command can create empty object classes as part of its execution.

The output of the **odmcreate** command is a **.h** file (an include file) that contains the C language definitions for the object classes defined in the ASCII *ClassDescriptionFile* file. The resulting include file is used by the application for accessing objects stored in ODM. The **odmcreate** command also produces a **.c** file that must be compiled and bound in with the application. The **.c** file contains structures and definitions that are used internally by ODM at run time.

The *ClassDescriptionFile* parameter specifies an ASCII file that contains descriptions of one or more object classes. The general syntax for the *ClassDescriptionFile* parameter is as follows:

Item	Description
file	: classes
classes	: class | classes class
class	: head body tail
head	: **struct** *ClassName* **{**
tail	: **}**
body	: elements
elements	: elements | elements element
element	:**char** *DescriptorName* **[** *DescriptorSize* **];**
	vchar *DescriptorName* **[** *DescriptorSize* **];**
	binary *DescriptorName* **[** *DescriptorSize* **];**
	short *DescriptorName* **;**
	long *DescriptorName* **;**
	long64 or **int64** or **ODM_LONG_LONG** *DescriptorName* **;**
	method *DescriptorName* **;**
	link *StdClassName StdClassName ColName DescriptorName* **;**

The default suffix for a *ClassDescriptionFile* file is **.cre**. If no suffix is specified on the **odmcreate** command, then a **.cre** suffix is appended. The file can have C language comments if run with the **-p** flag, and can include **#define** and **#include** lines that can be preprocessed if the **-p** flag is used to run the C language preprocessor on the file.

> **Note:** ODM databases are 32-bit databases. The long type, when used in the class description file is a 32-bit data item. The long64 or int64 type, when used in the class description file

is a 64-bit data item. The generated files will function the same for both 32- and 64-bit applications.

Flags

Item	Description
-c	Creates empty object classes only; does not generate the C language **.h** and **.c** files.
-h	Generates the **.c** and **.h** files only; does not create empty classes.
-p	Runs the C language preprocessor on the *ClassDescriptionFile* file.

Example

Assuming that a *ClassDescriptionFile* file named `FileName.cre` exists, the following command creates object classes:

```
odmcreate FileName.cre
```

Below is the `FileName.cre` source file and the resulting **.h** file:

```
/* This is an example odmcreate input file */
/* FileName.cre */

       class Class2 {
            char keys[32];
            method card;
            long cash;
            };
      class TstObj {
            long a;
            char b[80];
            link Class2 Class2 card Class2Ln;
            };

/* End of FileName.cre */

/* This is the generated header file FileName.h */

#include <odmi.h>

struct Class2 {
     long _id;         /* unique object id within object class */
     long _reserved;   /* reserved field */
     long _scratch;    /* extra field for application use */
     char keys[32];
     char card[256];   /* method */
     long cash;
     };
#define Class2_Descs 3

extern struct Class Class2_CLASS[];
#define get_Class2_list (a,b,c,d,e) (struct Class2 * ) odm_get_list (a,b,c,d,e)

struct TstObj {
     long _id;         /* unique object id within object class */
     long _reserved;   /* reserved field */
     long _scratch;    /* extra field for application use */
     long a;
     char b[80];
     struct Class2 *Class2Ln;  /* link */
     struct objlistinfo *Class2Ln_info; /* link */
     char Class2Ln_Lvalue[256];        /* link */
     };
#define TstObj_Descs 3

extern struct Class TstObj_CLASS[];
#define get_TstObj_list (a,b,c,d,e) (struct TstObj * ) odm_get_list (a,b,c,d,e)

/* End of generated header file FileName.h */
```

odmdelete Command

Purpose

Deletes selected objects from a specified object class.

Syntax

odmdelete -o *ObjectClass* [**-q** *Criteria*]

Description

The **odmdelete** command, given the object class to delete from and the search criteria, deletes all objects that meet those criteria.

Flags

Item	Description
-o *ObjectClass*	Specifies the object class to delete from.
-q *Criteria*	Specifies the criteria used to select objects from the object class. For information on qualifying criteria, see html

odmdrop Command

Purpose

Removes an object class.

Syntax

odmdrop -o *ClassName*

Description

The **odmdrop** command removes an entire object class and all of its objects. No checking is done to see if other object classes are linked to this one.

Flags

Item	Description
-o *ClassName*	Specifies the object class to remove.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

Assuming that an object class named `MyObjectClass` exists, the following command removes the object class:

```
odmdrop -o MyObjectClass
```

odmget Command

Purpose

Retrieves objects from the specified object classes into an **odmadd** input file.

Syntax

odmget [**-q** *Criteria*] *ObjectClass* ...

Description

The **odmget** command takes as input a search criteria and a list of object classes, retrieves the selected objects from the specified object classes, and writes an ASCII **odmadd** input file to standard output.

Flags

Item	Description
-q *Criteria*	Specifies the search criteria used to select objects from the object class or classes.

Examples

1. To display information about the Customized Device Drivers (CuDvDr) or the major (value1) and minor (value2) numbers of a device that has a logical name, $logical_name, enter the following command:

   ```
   # odmget -q "resource='devno' and value3='$logical_name'" CuDvDr
   ```

   The output might be similar to the following example:

   ```
   CuDvDr:
           resource = "devno"
           value1 = "19"
           value2 = "8"
           value3 = "hdisk0"
   ```

2. To display information about the Customized Devices (CuDv) and the corresponding Predefined Devices (PdDv) of a device that has a logical name, $logical_name, enter the following command:

   ```
   # odmget -q uniquetype=$(odmget -q name=$logical_name CuDv | tee /dev/tty | grep PdDvLn |
   cut -d'"' -f2) PdDv

   # logical_name=hdisk0

   # odmget -q uniquetype=$(odmget -q name=$logical_name CuDv | tee /dev/tty | grep PdDvLn |
   cut -d'"' -f2) PdDv
   ```

   The output might be similar to the following example:

   ```
   CuDv:
           name = "hdisk0"
           status = 1
           chgstatus = 2
           ddins = "scsidisk"
           location = "C5-T1-01"
           parent = "fscsi1"
           connwhere = "W_2"
           PdDvLn = "disk/fcp/mpioapdisk"

   PdDv:
           type = "mpioapdisk"
           class = "disk"
           subclass = "fcp"
   ```

```
        prefix = "hdisk"
        devid = ""
        base = 1
        has_vpd = 1
        detectable = 1
        chgstatus = 0
        bus_ext = 0
        fru = 1
        led = 1574
        setno = 2
        msgno = 0
        catalog = "scdisk.cat"
        DvDr = "scsidisk"
        Define = "/usr/lib/methods/define"
        Configure = "/usr/lib/methods/cfgscsidisk"
        Change = "/usr/lib/methods/chgdisk"
        Unconfigure = "/usr/lib/methods/ucfgdevice"
        Undefine = "/usr/lib/methods/undefine"
        Start = ""
        Stop = ""
        inventory_only = 0
        uniquetype = "disk/fcp/mpioapdisk"
```

3. To display information about the CuDv and Customized Attribute (CuAt) of a device that has a logical name, $logical_name, enter the following command:

```
# logical_name=vio0

# odmget -q "name='$logical_name'" CuDv CuAt
```

The system displays the following output:

```
CuDv:
        name = "vio0"
        status = 1
        chgstatus = 2
        ddins = "vdev_busdd"
        location = ""
        parent = "sysplanar0"
        connwhere = "vdevice"
        PdDvLn = "bus/chrp/vdevice"

CuAt:
        name = "vio0"
        attribute = "bus_id"
        value = "0x90000340"
        type = "R"
        generic = "D"
        rep = "n"
        nls_index = 5
```

odmshow Command

Purpose

Displays an object class definition on the screen.

Syntax

odmshow *ObjectClass*

Description

The **odmshow** command takes as input an object class name (*ObjectClass*) and displays the class description on the screen. The class description is in the format taken as input to the **odmcreate** command.

Example

Assuming that an object class named `MyObjectClass` exists, the following command displays the description of `MyObjectClass` on the screen:

```
odmshow MyObjectClass
```

Also, see the **odmcreate** command or **ODM Example Code and Output** in *General Programming Concepts: Writing and Debugging Programs* for an example of the output listing.

on Command

Purpose

Executes commands on remote systems.

Syntax

/usr/bin/on [**-i**] [**-d**] [**-n**] *Host Command* [*Argument ...*]

Description

The **on** command executes commands on other systems in an environment that is similar to the one running the program. The **on** command passes the local environment variables to the remote machine, thus preserving the current working directory. When using the **on** command, both users must have the same user identification. Relative path names work only if they are within the current file system. Absolute path names can cause problems since commands are issued at one machine and executed on another.

The standard input is connected to the standard input of the remote command. The standard output and standard error from the remote command are sent to the corresponding files for the **on** command. The root user cannot execute the **on** command.

> **Attention:** When the working directory is remotely mounted over the Network File System (NFS), the Ctrl-Z key sequence causes the window to hang.

Flags

Item	Description
-d	Specifies debug mode. Displays status messages as work progresses.
-i	Specifies interactive mode. Uses remote echoing and special character processing. This option is needed for programs that expect to be talking to a terminal. All terminal modes and window size changes are increased.
-n	Specifies no input. This option causes the remote program to get an end-of-file (EOF) message when it reads from standard input. This flag is necessary when running commands in the background with job control.

Example

To execute the **ls -al** command on another machine and display the in-progress status messages on your terminal, enter:

```
on  -d zorro ls -al
```

In this example, the **on** command executes the **ls** command on a workstation named `zorro`.

Files

Item	Description
/etc/inetd.conf	Defines how the **inetd** daemon handles Internet service requests.

openpts Command

Purpose

Allows enrolling and certifying a remote system.

Syntax

openpts [**-i** [**-f**] | [**-v**] | **-r** | **-D**] [**-h**] [**-V**] [**-u**] [**-l** *username*] [**-p** *port*] [**-c** *configfile*] **host**

Description

The **openpts** command allows the system (the verifier) to connect to a remote **host** (the collector) to determine whether the collector has performed a trusted boot. A machine is considered to have performed trusted boot when the contents of the collector's trusted platform module (TPM) is interrogated for consistency against a reference set of measurements (reference manifest) maintained by the verifier. To acquire the set of reference measurements, the verifier must first enroll the collector by using the **-i** option. After enrollement, the verifier can attest the collector with the default **-v** option that compares the current values represented in the integrity report against the reference set. The success or failure of this operation is reported to you along with the reason of failure. Examples of operations that may cause a failed certification include booting from a different device, changing the boot flags, and modifying the boot image.

If updates are pending to the state of the collector (for example, an OS upgrade that affects the next boot operation) these updates are reported during an attestation. The user is prompted to accept or reject the new values. Updates can be automatically accepted by using the **-u** option. The attestation request uses secure shell (SSH) as the communication mechanism between the collector and the verifier. The **openpts** command uses parameters such as **-l** for ssh command username and **-p** for port.

Flags

Item	Description
-c *configfile*	Specifies the configuration file to use. The default is ~/.openpts/openpts.conf.
-D	Displays the configuration settings of the target and all the options.
-h	Displays the command usage information.
-i [*-f*]	Enrolls a new collector partition or forces the enrollment of an existing collector.
-l *username*	Specifies the ssh command username.
-p *port*	Specifies the ssh command port number.
-r	Removes all information about a target system.
-u	Allow the command to accept updates to the manifest from the collector without prompting the **yes** option. The default is no.
-v (**default**)	Verifies a collector against its existing reference manifest.
-V	Displays the information in verbose mode. Multiple **-V** options increase the verbosity. This is used for debugging the data.

Files

Item	Description
~/.openpts/	This directory is the default location for all configuration and remote host information.
~/.openpts/openpts.conf	The configuration of the verifier.
~/.openpts/uuid	The UUID file of the verifier.
~/.openpts/UUID/ir.xml	The last integrity report received from the remote host.
~/.openpts/UUID/ newrm_uuid	The UUID file of the new reference manifest (for example, for the next boot operation after a system update).
~/.openpts/UUID/ policy.conf	The policy to verify the properties of a remote host.
~/.openpts/UUID/rm_uuid	The UUID file of the reference manifest.
~/.openpts/UUID/UUID/ rmN.xml	The reference manifests of the remote host.
~/.openpts/UUID/ target.conf	The configuration of the remote host.
~/.openpts/UUID/ vr.properties	The platform properties of the remote host derived from the integrity report.

OS_install Command

Purpose

Performs network installation operations on `OS_install` objects.

Syntax

Traditional usage:

`OS_install` [**-K** *keyfile_path_name*]{ `-o` *Operation* } [**-F**] [-a`-` *attr=value...*] {*ObjectName*}

For system plan installations (System Plan mode):

`OS_install` [**-K** *keyfile_path_name*] `-i sysplan` { `-x` *sysplan.xml* } [`-d`] [**-F**]

For listing `OS_install` objects (List mode):

`OS_install -l` [`-v`] [`-t` *object_type* | *object_name*]

For managing network daemons:

`OS_install` **-S** | **-U**

Description

The `OS_install` command performs a network installation operation on an `OS_install` object. The type of operation is dependent on the type of object specified by the *ObjectName* parameter. The object pointed to by the *ObjectName* parameter can be one of four types: `Client`, `OS_Resource`, `Remote_Resource` or `Control_Host`. Command operations involve the creation and management of `OS_install` objects that enable network installation to install an operation system on a client system.

`OS_install` can also be run in System Plan mode by passing the `-i sysplan` flag instead of specifying an operation. This operation provides the ability to combine multiple `OS_install` operations into a single XML document.

The operations involving Remote_Resource objects require configuring an SSH key that is generated with the ssh-keygen command. The SSH key is required to run ssh commands on the local platform and remote resource server. On an HMC, the default name of the file *keyfile_path_name* containing the SSH key is /home/hscroot/ssh_keys. This file name can be overridden with the **-K** option. On other platforms, there is no default file name for the SSH key file. If the **-K** option is not specified on other platforms, the standard path names of SSH key files must be accessible to the OS_install command process.

The List mode of OS_install is used to list the current configuration of objects in the OS_install environment.

The HMC or IVM network daemons can be started and stopped with the **S** and **U** options, without modifying the OS_install objects.

Flags

Item	Description
-a *attr=value*	Assigns the specified value to the specified attribute. Operations lists the required and optional attributes for a specific operation.
-d	Deletes all OS_install objects created during System Plan mode after all operations are completed.
-F	Authorizes a reset of the existing remote server client system objects if required, during an OS_install allocate operation or system plan installation.
-i sysplan	Specifies System Plan mode.
-K *keyfile_path_name*	Specifies the absolute path name of the file where the SSH keys are generated.
-l	Lists all OS_install objects in the environment by default.
-o *Operation*	Specifies an operation to perform on an OS_install object.
-S	Starts the network daemons without modifying the OS_install objects.
-t *object_type* | *object_name*	Narrows the list returned by the -l flag to only objects of type *object_type* or to the single OS_install object specified by *object_name*.
-U	Stops the network daemons without modifying the OS_install objects.
-v	Displays the list returned by the -l flag.
-x *sysplan.xml*	Specifies the XML file that contains the system plan.

Operations

Operation	Description	Required Attributes	Optional Attributes
`define_client` [-a *attr=value*...] {*ClientObjectName*}	Defines a new client object.	**ip_addr** Client's IP address. **mac_addr** MAC address of the network interface of the client system. **gateway** IP gateway address of the client system. **subnet_mask** IP subnet mask of the client system. **lpar** LPAR name to install client (required attribute for the netboot operation). **profile** LPAR profile to use for the client (required attribute for the netboot operation). **managed_system** Name of the managed system that contains LPAR (required attribute for the netboot operation). **ctrl_host** Name of the Hardware Control Host object for this client (required attribute for the netboot operation).	**adapter_speed** Speed of the network adapter of the client system. **adapter_duplex** Duplex setting of the network adapter of the client system. **disk_location** Location of the disk to install client. **vlan_tag** Specifies the virtual logical area network (VLAN) tag to be used for tagging Ethernet frames during network installation for virtual network communication. Valid values are 0 - 4094. **vlan_pri** Specifies the virtual logical area network (VLAN) tag to be used for tagging Ethernet frames during network installation for virtual network communication. Valid values are 0 - 7.
`define_resource` [-a *attr=value*...] {*ResourceObjectName*}	Defines a new `OS_Resource` object.	**type** AIX or VIOS. **version** OS version. **location** Absolute path where `OS_Resource` resides. **source** Source of installation images.	**configfile** Install configuration file.

Operation	Description	Required Attributes	Optional Attributes
`define_remote_resource` [-a *attr=value*...] {*ResourceObjectName*}	Defines a new `Remote_Resource` object.	**server** Host name of the remote resource server. **type** AIX or Linux. **remote_identifier** Name of the resource or resource set on the remote resource server.	**communication_method** Supports ssh communication method.
`define_ctrl_host` [-a *attr=value*...] {*ControlHostObjectName*}	Defines a new Hardware Control_Host object.	**communication_method** Supports ssh communication method. **hostname** Host name of control host (the host name localhost can be specified if `OS_install` is run on the HMC control host). **type** hmc or ivm.	None.
`allocate` [-F][-a *attr=value*...] {*ClientObjectName*}	Allocates an `OS_Resource` or `Remote_Resource` to a client object. Both objects must exist in the `OS_install` environment. An error occurs if the client object has an `OS_Resource` or `Remote_Resource` already allocated to it.	**os_resource** Existing `OS_Resource` or `Remote_Resource` object to allocate to the client object. **remote_resource** Existing `Remote_Resource` object to allocate to the client object. **install_resource** Existing `OS_Resource` or `Remote_Resource` object to allocate to the client object.	**config_file** Install configuration file (applies for an `OS_Resource` object).
`netboot` {*ClientObjectName*}	Instructs the hardware control host of the client object to initiate a network boot.	None.	None.
`monitor_installation` {*ClientObjectName*}	Monitors the installation status of the client object.	None.	None.

Operation	Description	Required Attributes	Optional Attributes
`deallocate` {*ClientObjectName*}	Deallocates the `OS_Resource` or `Remote_Resource` that was allocated to the client object by an allocate operation.	None.	None.
`remove` {*ObjectName*}	Removes the object from the `OS_install` environment.	None.	None.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Examples

1. To define a client object, enter a command similar to the following:

   ```
   OS_install -o define_client -a ip_addr=128.0.64.117 -a mac_addr=ab:cc:de:10:23:45 -a \
   gateway=128.0.64.1 -a subnet_mask=255.255.255.0 -a ctrl_host=myhmc -a lpar=AIX1 -a \
   profile=AIX1 -a managed_system=myMngSys myclient01
   ```

   The preceding client object is a logical partition in a managed system.
2. To define an `OS_Resource` object, enter a command similar to the following:

   ```
   OS_install -o define_resource -a location=/images/AIX/53ML3 -a type=AIX -a version=53ML3
   my53resource
   ```
3. To define a `Remote_Resource` object (using the `OS_install` default SSH key file for HMC), enter a command similar to the following:

   ```
   OS_install -o define_remote_resource -a server=MyNimServer -a type=AIX
     -a remote_identifier=NimResGrp1 myRemoteResource
   ```
4. To define a `Remote_Resource` object (using a previously generated ssh-keygen key located in /home/hscroot/id_dsa file), enter the following:

   ```
   OS_install -K /home/hscroot/id_dsa -o define_remote_resource -a server=MyNimServer -a
   type=AIX -a
   remote_identifier=NimResGrp1 myRemoteResource
   ```
5. To allocate the `OS_Resource` object defined in example 2 to a client object, enter a command similar to the following:

   ```
   OS_install -o allocate -a os_resource=my53resource myclient01
   ```

   or

   ```
   OS_install -o allocate -a install_resource=my53resource myclient01
   ```
6. To allocate the `Remote_Resource` object defined in example 3 to a client object and authorize reset on an existing client, enter a command similar to the following:

   ```
   OS_install -o allocate -F -a remote_resource=myRemoteResource myclient01
   ```

   or

```
OS_install -o allocate -F -a install_resource=myRemoteResource myclient01
```

7. To deallocate the `my53resource` client object that was allocated in the example 5, enter:

```
OS_install -o deallocate myclient01
```

8. To define a `Control_Host` object to be specified for the `ctrl_host` attribute of a Client object, enter a command similar to the following:

```
OS_install -o define_ctrl_host -a type=hmc -a hostname=hmc_hostname -a
communication_method=ssh myhmc
```

   Although the preceding example shares the same name of the `ctrl_host` attribute in the first example, the `define_client` operation allows an undefined `Control_Host` object to be specified for the `ctrl_host` attribute. In that case the controlling host of the Client object must be the HMC or IVM on which the `netboot` operation for the client is executed.

9. To execute a `netboot` operation, enter:

```
OS_install -o netboot myclient01
```

10. To view a `myclient01` installation, enter:

```
OS_install -o monitor_installation myclient01
```

11. To remove the definition of the `my53resource` object, enter:

```
OS_install -o remove my53resource
```

12. To remove the definition of the `myclient01` object, enter:

```
OS_install -o remove myclient01
```

    If an `OS_Resource` object is specified, the `remove` operation removes OS images that exist in the file system directory specified by the `location` attribute of the object.

Configuring SSH

- Generate SSH Rivest-Shamir-Adleman (RSA) keys and place them in an accessible ssh_keys file in the HMC HOME directory, by entering the command:

```
ssh-keygen -t rsa -f /home/hscroot/ssh_keys
```

- On the remote resource server, append or copy the content of the /home/hscroot/ssh_keys.pub file that is generated by using the **ssh-keygen** command to the resource server's .ssh/authorized_keys file.
- If `OS_install` command is used to run a netboot operation on a target client of a remote HMC control host, append the content of the /home/hscroot/ssh_keys.pub file that is generated by using the **ssh-keygen** command to the remote HMC hscroot user's .ssh/authorized_keys2 file, by entering the following command as a hscroot user on the remote HMC:

```
mkauthkeys -a '<content_of_ssh_keys.pub>'
```

Location

Item	Description
`/usr/sbin/OS_install`	
`/opt/osinstall`	Directory containing the `OS_install` Perl module files.

Files

Item	Description
`/var/osinstall`	Directory containing configuration files for the `OS_install` environment.
`/home/hscroot/ssh_keys`	Default file name for SSH keys on an HMC.

oslevel Command

Purpose

Reports the latest installed level (technology level, maintenance level and service pack) of the system.

Syntax

oslevel [**-l** *Level* | **-g** *Level* | **-q**] [**-r** | **-s**] [**-f**]

Description

The **oslevel** command reports the technology level and service pack of the operating system using a subset of all filesets installed on your system. These filesets include the Base Operating System (BOS), base devices, base printers, and X11.

The **oslevel** command also prints information about the technology level and service pack, including which filesets are not at a specified technology level or service pack.

Flags

Item	Description
-l *Level*	Lists filesets that are earlier (less) than the technology level or service pack specified by the *Level* parameter.
-f	Forces the **oslevel** command to rebuild the cache for this operation.
-g *Level*	Lists filesets that are later (greater) than the technology level or service pack specified by the *Level* parameter.
-q	Lists names of known technology levels (when used with the **-r** flag) or service packs (when used with the **-s** flag) that can be specified using the **-l** or **-g** flag.
-r	Applies all flags to technology levels.
-s	Applies all flags to service packs. The service pack level returned is in the format 6100-00-01-0748, where 6100 refers to base level 6.1.0.0; 00 refers to technology level 0; 01 refers to service pack 1; and 0748 is the build sequence identifier, which is used to determine valid technology levels and service packs that can be applied to the current level. Attempts to apply a technology level or service pack with a lower build sequence identifier will fail.

If no flags are specified, the base system software is entirely at or above the level that is listed in the output of the **oslevel** command.

Examples

1. To determine the base level of the system, type:

   ```
   oslevel
   ```

   The output will be similar to the following:

```
6.1.0.0
```

2. To determine the highest technology level reached for the current version of AIX on the system, type:

```
oslevel -r
```

3. To list all known technology levels on the system, type:

```
oslevel -rq
```

The levels returned can be used with the [**-r -l**] or [**-r -g**] flags, and will be similar to the following:

```
Known Recommended Maintenance Levels
------------------------------------
5300-02
5300-01
5300-00
```

4. To list which software is below AIX Version 5.3 technology level 1, type:

```
oslevel -r -l 5300-01
```

5. To list which software is at a level later than AIX Version 5.3 technology level 1, type:

```
oslevel -r -g 5300-01
```

6. To determine the highest service pack reached for the current technology level on the system, type:

```
oslevel -s
```

7. To list the known service packs on a system, type:

```
oslevel -sq
```

The levels returned can be used with the [**-s -l**] or [**-s -g**] flags, and will be similar to the following:

```
Known Service Packs
-------------------
6100-00-02-0750
6100-00-01-0748
6100-00-00-0000
```

8. To list which software is below AIX Version 6.1 technology level 0, service pack 1, type:

```
oslevel -s -l 6100-00-01-0748
```

9. To list which software is at a level later than AIX Version 6.1 technology level 0, service pack 1, type:

```
oslevel -s -g 6100-00-01-0748
```

Files

Item	Description
/usr/bin/oslevel	Contains the **oslevel** command.

ospf_monitor Command

Purpose

Monitors the OSPF gateways.

Syntax

ospf_monitor *mon_db_file*

Description

The **ospf_monitor** command is used to query OSPF routers. The **ospf_monitor** command operates in interactive mode. It allows the user to query the various OSPF routers to provide detailed information on I/O statistics, error logs, link-state data bases, AS external data bases, the OSPF routing table, configured OSPF interfaces, and OSPF neighbors.

Specify the complete pathname of a database composed of records configuring destinations for **ospf_monitor** remote commands with *mon_db_file*. Each destination record is a single-line entry which lists the destination IP address, the destination hostname, and an OSPF authentication key (if authentication is activated by the destination). Since authentication keys may be present in the destination records, it is recommended that general access to this database be restricted.

Refer to RFC-1583 (OSPF Specification, version 2) for details about OSPF database and packet formats.

Commands

Upon entering interactive mode, **ospf_monitor** presents the `'[ # ] dest command params >'` prompt, at which you can enter any of **ospf_monitor**'s interactive commands. Interactive commands can be interrupted at any time with a keyboard interrupt.

Note: The command line length must be less than 200 characters.

Local Commands

Item	Description
?	Displays all local commands and their functions.
?R	Displays all remote commands and their functions.
d	Displays all configured destinations. This command displays *dest_index* , the IP address, and the hostname of all potential **ospf_monitor** command destinations configured in *mon_db_file*.
h	Displays the command history buffer showing the last 30 interactive commands.
x	Exits the **ospf_monitor** program.
@ *remote_command*	Sends *remote_command* to the same (previous) destination.
@*dest_index remote_command*	Sends *remote_command* to configured destination *dest_index*.
F *filename*	Sends all **ospf_monitor** output to *filename*.
S	Sends all **ospf_monitor** output to stdout.

Remote Commands

Item	Description
a *area_id type ls_id adv_rtr*	Displays link state advertisement. *Area_id* is the OSPF area for which the query is directed. *adv_rtr* is the router-id of the router which originated this link state advertisement. *Type* specifies the type of advertisement to request and should be specified as follows: **1** Request the router links advertisements. They describe the collected states of the router's interfaces. For this type of request, the *ls_id* field should be set to the originating router's Router ID. **2** Request the network links advertisements. They describe the set of routers attached to the network. For this type of request, the *ls_id* field should be set to the IP interface address of the network's Designated Router. **3** Request the summary link advertisements describing routes to networks. They describe inter-area routes, and enable the condensing of routing information at area borders. For this type of request, the *ls_id* field should be set to the destination network's IP address. **4** Request the summary link advertisements describing routes to AS boundary routers. They describe inter-area routes, and enable the condensing of routing information at area borders. For this type of request, the *ls_id* field should be set to the Router ID of the described AS boundary router. **5** Request the AS external link advertisements. They describe routes to destinations external to the Autonomous System. For this type of request, the *ls_id* field should be set to the destination network's IP address.
c	Displays cumulative log. This log includes input/output statistics for monitor request, hello, data base description, link-state request, link-state update, and link-state ack packets. Area statistics are provided which describe the total number of routing neighbors and number of active OSPF interfaces. Routing table statistics are summarized and reported as the number of intra-area routes, inter-area routes, and AS external data base entries.
e	Displays cumulative errors. This log reports the various error conditions which can occur between OSPF routing neighbors and shows the number of occurrences for each.
h	Displays the next hop list. This is a list of valid next hops mostly derived from the SPF calculation.
l [*retrans*]	Displays the link-state database (except for ASE's). This table describes the routers and networks making up the AS. If *retrans* is non-zero, the retransmit list of neighbors held by this lsdb structure will be printed.
A [*retrans*]	Displays the AS external data base entries. This table reports the advertising router, forwarding address, age, length, sequence number, type, and metric for each AS external route. If *retrans* is non-zero, the retransmit list of neighbors held by this lsdb structure will be printed.

Item	Description
o [*which*]	Displays the OSPF routing table. This table reports the AS border routes, area border routes, summary AS border routes, networks, summary networks and AS external networks currently managed via OSPF. If *which* is omitted, all of the above will be listed. If specified, the value of *which* (between 1 and 63) specifies that only certain tables should be displayed. The appropriate value is determined by adding up the values for the desired tables from the following list: **1** Routes to AS border routers in this area. **2** Routes to area border routers for this area. **4** Summary routes to AS border routers in other areas. **8** Routes to networks in this area. **16** Summary routes to networks in other areas. **32** AS routes to non-OSPF networks.
I	Displays all interfaces. This report shows all interfaces configured for OSPF. Information reported includes the area, interface IP address, interface type, interface state, cost, priority, and the IP address of the DR and BDR for the network.
N	Displays all OSPF routing neighbors. Information reported includes the area, local interface address, router ID, neighbor IP address, state, and mode.
V	Displays Gated version information.

p

The following AIX commands begin with the letter *p*.

pac Command

Purpose

Prepares printer/plotter accounting records.

Syntax

/usr/sbin/pac [**-c**] [**-m**] [**-p***Price*] [**-P***Printer*] [**-q***File*] [**-r**] [**-s**] [*Name ...*]

Description

The **pac** command prepares printer/plotter accounting records for each user of the selected printer or for the users specified by the *Name* parameter. For printer choices, see the **-P** flag.

The unit of measure is the number of pages, with the exception of raster devices, for which feet of paper is measured. Output is expressed both as the number of units used and the charge in dollars. For information on the charge (price) per unit, see the **-p** flag.

The accounting file specified in the **/etc/qconfig** file and the file created to contain the summary information must grant read and write permissions to the root user or printq group. The **pac** command generates the summary file name by appending **_sum** to the path name specified by the `acctfile =` clause in the **/etc/qconfig** file. For example, if the **qconfig** file reads:

```
acctfile = /var/adm/1p0acct
```

The **pac** command expects the summary file to be named **/var/adm/1p0acct_sum**.

Flags

Item	Description
-c	Sorts the output by price instead of alphabetically by user.
-m	Groups all the printing charges for a user, regardless of the host machine.
-p*Price*	Specifies the price, in dollars, charged per unit of output. By default, the system charges $0.02 per unit.
-P*Printer*	Specifies the printer for which accounting records are prepared. By default, the system selects the printer named by the **PRINTER** environment variable or the default value **lp0**. **Note:** When the **LPDEST** environment variable is set, it takes precedence over the **PRINTER** environment variable, which has an identical function. Any destination options issued from the command line override both the **LPDEST** and **PRINTER** environment variables.
-q*File*	Specifies the queue configuration file. The default value is the **/etc/qconfig** file.
-r	Reverses the sorting order, so that records are sorted alphabetically from z to a, or in descending order by price.
-s	Summarizes the accounting information in a summary file. This flag is needed for busy systems.

Examples

1. To produce printer/plotter accounting information for all users of the `lp0` printer, enter:

   ```
   /usr/sbin/pac
   ```

   The command displays the number of printed pages and the charge, sorted by user. This example assumes that there is no **PRINTER** environment variable.
2. To collect printer/plotter accounting records in a summary file, enter:

   ```
   /usr/sbin/pac  -s
   ```
3. To produce printer/plotter accounting information for `smith`, `jones`, and `greene` from the `lp12` printer, enter:

   ```
   /usr/sbin/pac  -Plp12 smith jones greene
   ```

   Note: Do not place a space between a flag and its variable; for example, the **-p***Price,* **-P***Printer*, and **-q***File.*

Files

Item	Description
/usr/sbin/pac	Contains the **pac** command.
/etc/qconfig	Specifies the path to the file.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

pack Command

Purpose

Compresses files.

Syntax

pack [**-f**] [**-**] *File ...*

Description

The **pack** command stores the file specified by the *File* parameter in a compressed form. The input file is replaced by a packed file with the same name and the suffix **.z** appended. If the invoking process has appropriate privileges, the packed file maintains the same access modes, access and modification dates, and owner as the original file. The input file name can contain no more than 253 bytes to allow space for the added **.z** suffix. If the **pack** command is successful, the original file is removed. Packed files can be restored to their original form using the **compress** command.

The exit value of the **pack** command is the number of files that it could not pack. The **pack** command does not pack under any of the following conditions:

- The file is already packed.
- The input file name has more than 253 bytes.

- The file has links.
- The file is a directory.
- The file cannot be opened.
- No storage blocks are saved by packing.
- A file called *File***.z** already exists.
- The **.z** file cannot be created.
- An I/O error occurred during processing.

Flags

Item	Description
-f	Forces packing of the file specified by the *File* parameter. This is useful for packing an entire directory, even if some of the files will not benefit.

Parameters

Item	Description
File	Specifies the file to be packed.
-	Displays statistics about the file specified by the *File* parameter. The statistics are calculated from a Huffman minimum redundancy code tree built on a byte-by-byte basis. Additional occurrences of the **-** (minus sign) parameter on the command line toggles this function for the next specified file. See example 2.

Exit Status

This command returns the following exit values:

Item	Description
0	Specifies that the file was successfully packed.
>0	Specifies that an error occurred.

Examples

1. To compress the files named chap1 and chap2 and display the revised file names, enter:

   ```
   pack chap1 chap2
   ```

   The compressed versions are renamed chap1.z and chap2.z. The **pack** command displays the percent decrease in size for each file compressed.
2. To display statistics about the amount of compression done, enter:

   ```
   pack - chap1 - chap2
   ```

   This compresses the files named chap1 and chap2 and displays statistics about the file named chap1, but not about the file named chap2. The first **-** (minus sign) parameter turns on the statistic display, and the second **-** parameter turns it off.

Files

Item	Description
/usr/bin/pack	Contains the **pack** command.

packf Command

Purpose

Compresses the contents of a folder into a file.

Syntax

packf [**+***Folder*] [*Messages*] [**-file** *File*]

Description

The **packf** command compresses the messages in a folder into a specified file. By default, the **packf** command compresses messages from the current folder and places them in the **msgbox** file. If the file does not exist, the system prompts you for permission to create it. Each message in the file is separated with four Ctrl-A characters and a new-line character.

Note: You can use the **inc** command to unpack compressed messages.

Flags

Item	Description
-file *File*	Specifies the file in which to put compressed messages. The default is the **./msgbox** file. If the file exists, the **packf** command appends the messages to the end of the file. Otherwise, the system prompts you for permission to create the file.
+*Folder*	Identifies the folder containing the messages you want to pack.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.

Item	Description
Messages	Specifies what messages to pack. The *Messages* parameter can specify several messages, a range of messages, or a single message. If several messages are specified, the first message packed becomes the current message. Use the following references to specify messages: ***Number*** Number of the message. When specifying several messages, separate each number with a space. When specifying a range, separate the first and last numbers in the range with a hyphen. ***Sequence*** A group of messages specified by the user. Recognized values include: **all** All the messages in the folder. This is the default. **cur or . (period)** Current message. **first** First message in a folder. **last** Last message in a folder. **next** Message immediately after the current message. **prev** Message immediately before the current message.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Current-Folder:`	Sets your default current folder.
`Msg-Protect:`	Sets the protection level for your new message files.
`Path:`	Specifies the user's MH directory.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To pack all the messages in the current folder and place the resulting text in the **schedule** file, enter:

   ```
   packf  -file schedule
   ```

   The system responds with a message similar to the following:

   ```
   Create file "/home/mary/schedule"?
   ```

   Enter y to create the file.

2. To pack the range of messages from 3 to 7 from the **test** folder into an existing **msgbox** file, enter:

   ```
   packf  +test 3-7
   ```

   The system responds with the `shell` prompt when the command is complete.
3. To pack the current, first, and last message in the **inbox** folder into an existing **msgbox** file, enter:

   ```
   packf cur first last
   ```

Files

Item	Description
$HOME/.mh_profile	Specifies the MH user profile.
/usr/bin/packf	Contains the **packf** command.

pagdel Command

Purpose

Removes any existing PAG association within the current process' credentials.

Syntax

paginit [**-R** *module_name*] [*username*]

Description

The **pagdel** command will remove the PAG identifier from the current process' credentials structure. If the **-R** option is omitted, the registry attribute will be used as the **module_name**.

Flags

Item	Description
-R *module_name*	Specifies a load module found in **/usr/lib/security/modules.cfg**. The **load_module** will be asked to delete any PAG currently associated with the process.

Security

Access Control: This command should grant execute (x) access only to the `root` user and members of the security group. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the `root` user with the **setuid** (SUID) bit set.

Auditing

USER_PagDelete

Example

To remove the PKI authentication certificate associated with the current process, type:

```
pagdel -R FPKI
```

pagesize Command

Purpose

Displays the system page size.

Syntax

pagesize [**-a**] [-f]

Description

The **pagesize** command prints the size, in bytes, of a page of memory, as returned by the **getpagesize** subroutine. Provided for system compatibility, this command is useful when constructing portable shell scripts.

If the **-a** flag is specified, the **pagesize** command prints all of the page size values (in bytes) supported on the system.

Flags

-a
Prints all of the page size values (in bytes) supported on the system.

-f
Prints the formatted page sizes with an alphabetical suffix rather than the page size in bytes (for example, 4K)

Example

1. To obtain the size system page, enter:

   ```
   pagesize
   ```

   The system returns the number of bytes, such as 4096.
2. To print the formatted page size, enter:

   ```
   pagesize -f
   ```

   The system returns the formatted page size (for example, 4K).
3. To print all of the supported page size with an alphabetical suffix, enter:

   ```
   pagesize -af
   ```

   The system returns all of the supported page sizes. For example:

   ```
   4K
   64K
   16M
   ```

Files

Item	Description
/usr/bin/pagesize	Contains the **pagesize** command.

paginit Command

Purpose

Authenticate a user and create a PAG association.

Syntax

paginit [**-R** *module_name*] [*username*]

Description

The **paginit** command authenticates *username* (by default, the user issuing the command) and creates an association between the *username* and a kernel token called a Process Authentication Group entry (PAG). A new login shell is spawned by this command.

If the **-R** flag is not given, **paglist** queries the user's registry attribute and use that value for *module_name*.

To associate the *username* with an alternate Identification and Authentication (I&A) mechanism, the **-R** flag can be used to specify the I&A load module used to create the user. Load modules are defined in the **/usr/lib/security/methods.cfg** file.

Flags

Item	Description
-R *module_name*	Specifies the loadable I&A module used to authenticate the user.

Parameters

Item	Description
username	Specifies the user. This parameter defaults to the user issuing the command. Only the **root** user may override the default.

Security

Access Control: This command should be executable by all. It should be owned by **root** and should be **setuid**.

Auditing

USER_Paginit

Example

```
paginit -R FPKI
```

The user is authenticated using the registry FPKI, which is defined in the **/usr/lib/security/methods.cfg** file. A PAG is associated with the current process credentials.

paglist Command

Purpose

Lists authentication information associated with the current process.

Syntax

paglist [**-R** *module_name*]

Description

The **paglist** command queries the current process' credentials to display its authentication certificate.

If the **-R** option is not given, **paglist** will query the user's registry attribute and use that value for **module_name**.

Flags

Item	Description
-R *module_name*	Specifies that the load module *module_name* is to list its authentication certificate associated with the current process.

Security

Access Control: This command runs with the ID of the invoking user, without any elevated privileges. It should be owned by `root`, but executable by all.

Example

```
paglist -R FPKI
```

This example will list the PAG associated with the current process within the FPKI registry.

panel20 Command

Purpose

Diagnoses activity between an HIA and the 5080 Control Unit.

Syntax

panel20 [**HIA0** | **HIA1** | **HIA2**]

Description

Use the **panel20** command as a diagnostic tool to determine whether the Host Interface Adapter (HIA) is correctly installed and communicating with the 5088 Graphics Channel Control Unit (GCCU).

The **panel20** command displays a diagnostic screen with the following columns: `Device Name`, `Channel Address`, `Link Address`, `Link Status`, `Poll Counter`, `SNRM Counter`.

If the HIA is correctly installed and the host operating system is correctly configured to support 3270 devices on the 5088, the entries in the Set Normal Response Mode (`SNRM Counter`) column will be increasing. If the entries in `SNRM Counter` are not increasing, refer to problem determination procedures for the HIA and verify that the host operating system is correctly configured.

Examples

To start the **panel20** command, enter:

```
panel20
```

By default, the **panel20** command will monitor HIA0. To monitor HIA1 or HIA2, enter:

```
panel20 HIA1
```

OR

```
panel20 HIA2
```

passwd Command

Purpose

Changes a user's password.

Syntax

passwd [**-R** *load_module*] [**-f** | **-s** **-a**] [*User*]

Description

The **passwd** command sets and changes passwords for users. Use this command to change your own password or another user's password. You can also use the **passwd** command to change the full name (gecos) associated with your login name and the shell you use as an interface to the operating system.

Depending on how the user is defined, the user's password can exist locally or remotely. Local passwords exist in the **/etc/security/passwd** database. Remote passwords are stored in the database provided by the remote domain.

To change your own password, enter the **passwd** command. The **passwd** command prompts the nonroot user for the old password (if one exists) and then prompts for the new password twice. (The password is never displayed on the screen.) If the two entries of the new password do not match, the **passwd** command prompts for the new password again.

Note: The **passwd** command uses only the first eight characters of your password for local and NIS passwords. Only 7-bit characters are supported in passwords. For this reason, globalization code points are not allowed in passwords. You can set a password of up to 255 characters.

To change another user's password, enter the **passwd** command and the user's login name (the *User* parameter). Only the root user or a member of the security group is permitted to change the password for another user. The **passwd** command prompts you for the old password of the user as well as the new password. For local passwords, the **passwd** command does not prompt the root user for either the old user password or the root password. For remote passwords, by default the root user will be prompted to input the old password so the remote domain can make the decision to use the password or ignore it. To change this behavior, see the **rootrequiresopw** option in the **/usr/lib/security/methods.cfg** file. The **passwd** command does not enforce any password restrictions upon the root user.

The **/etc/passwd** file records your full name and the path name of the shell that you use. To change your recorded name, enter the **passwd -f** command. To change your login shell, enter the **passwd -s** command.

Construct locally-defined passwords according to the password restrictions in the **/etc/security/user** configuration file. This file contains the following restrictions:

Item	Description
dictionlist	Specifies the list of dictionary files checked when a password is changed.
histexpire	Specifies the number of weeks that a user cannot reuse a password.
histsize	Specifies the number of previous passwords that the user cannot reuse.
maxage	Specifies the maximum age of a password. A password must be changed after a specified amount of time measured in weeks.

Item	Description
maxexpired	Specifies the maximum number of weeks beyond the `maxage` value that a password can be changed by the user.
maxrepeats	Specifies the maximum number of times a single character can be used in a password.
minalpha	Specifies the minimum number of alphabetic characters.
minother	Specifies the minimum number of other characters.
minlen	Specifies the minimum number of characters. **Note:** This value is determined by either the `minalpha` value plus the `minother` value or the `minlen` value, whichever is greater.
mindiff	Specifies the minimum number of characters in the new password that are not in the old password. **Note:** This restriction does not consider position. If the new password is `abcd` and the old password is `edcb`, the number of different characters is `1`.
minage	Specifies the minimum age at which a password can be changed. Passwords must be kept for a minimum period. This value is measured in weeks.
minloweralpha	Specifies the minimum number of lower case alphabetic characters.
minupperalpha	Specifies the minimum number of upper case alphabetic characters.
mindigit	Specifies the minimum number of digits.
minspecialchar	Specifies the minimum number of special characters.
pwdchecks	Specifies the list of external password restriction methods invoked when a password is changed.

If the root user adds the **NOCHECK** attribute to your flags entry in the **/etc/security/passwd** file, your password does not need to meet these restrictions. Also, the root user can assign new passwords to other users without following the password restrictions.

If the root user adds the **ADMIN** attribute to your flags entry or if the **password** field in the **/etc/passwd** file contains an * (asterisk), only the root user can change your password. The root user also has the exclusive privilege of changing your password if the **password** field in **/etc/passwd** contains an ! (exclamation point) and the **password** field in the **/etc/security/passwd** file contains an * (asterisk).

If the root user changes your password, the **ADMCHG** attribute is automatically added to your flags entry in the **/etc/security/passwd** file. In this case, you must change the password the next time you log in.

If the user's **registry** value in the **/etc/security/user** file is either DCE or NIS, the password change can only occur in the specified database.

The **passwd** command creates the user keystore, if the keystore does not exist and if the **efs_keystore_access** attribute value of the user is not **none**. The keystore is created with the Encrypted File System (EFS) attributes that are found in the **/etc/security/user** file. If the old password can open the keystore, it also changes the keystore password. That is to say, if the login and keystore passwords are same, then the **passwd** command changes both of the passwords. If the file system is an Encrypted File System (EFS), then the command performs as though the **-a** flag is specified. If you specify the **-a** flag, the result is that the EFS password is not synchronized with user login password after a password change. Therefore, the keystore is not be loaded automatically on next logins.

Flags

Item	Description
-a	Changes a user's password in all modules (compat, LDAP, NIS, and so on).

Item	Description
-f	Changes the user information accessed by the **finger** command. You can use this flag to provide your full name in the **/etc/passwd** file.
-s	Changes the login shell.
-R *load_module*	Specifies the loadable I&A module used to change a user's password.

Security

The **passwd** command is a PAM-enabled application with a service name of passwd. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the usw stanza of **/etc/security/login.cfg**, to PAM_AUTH as the root user.

The authentication mechanisms used when PAM is enabled depend on the configuration for the passwd service in **/etc/pam.conf**. The **passwd** command requires **/etc/pam.conf** entries for the password module type. Listed below is a recommended configuration in **/etc/pam.conf** for the passwd service:

```
#
# AIX passwd configuration
#

passwd password required /usr/lib/security/pam_aix
```

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To change your password, type:

   ```
   passwd
   ```

   The **passwd** command prompts you for your old password, if it exists and you are not the root user. After you enter the old password, the command prompts you twice for the new password.

2. To change your full name in the **/etc/passwd** file, type:

   ```
   passwd -f
   ```

   The **passwd** command displays the name stored for your user ID. For example, for login name sam, the **passwd** command could display this message:

   ```
   sam's current gecos:
                "Sam Smith"
   Change (yes) or no)? >
   ```

   If you type a Y for yes, the **passwd** command prompts you for the new name. The **passwd** command records the name you enter in the **/etc/passwd** file.

3. To use a different shell the next time you log in, type:

   ```
   passwd -s
   ```

   The **passwd** command lists the path names of the available shells and the shell you are currently using. The command also displays a prompt:

   ```
   Change (yes) or (no)? >
   ```

   If you type a Y for yes, the **passwd** command prompts you for the shell to use. The next time you log in, the system provides the shell that you specify here.

Files

Item	Description
/usr/bin/passwd	Contains the **passwd** command.
/etc/passwd	Contains user IDs, user names, home directories, login shell, and finger information.
/etc/security/passwd	Contains encrypted passwords and security information.

paste Command

Purpose

Joins the lines of different files.

Syntax

paste [**-s**] [**-d** *List*] *File1* ...

Description

The **paste** command reads input from the files specified on the command line. The command reads from standard input if a **-** (minus sign) appears as a file name. The command concatenates the corresponding lines of the given input files and writes the resulting lines to standard output.

By default, the **paste** command treats each file as a column and joins them horizontally with a tab character (parallel merging). You can think of the **paste** command as the counterpart of the **cat** command (which concatenates files vertically, that is, one file after another).

With the **-s** flag, the **paste** command combines subsequent lines of the same input file (serial merging). These lines are joined with the tab character by default.

Notes:

1. The **paste** command supports up to 32767 input files (the **OPEN_MAX** constant).
2. The action of the **pr -t -m** command is similar to that of the **paste** command, but creates extra spaces, tabs, and lines for a nice page layout.
3. Input files should be text files, but may contain an unlimited number of line lengths.

Flags

Item	Description
-d *List*	Changes the delimiter that separates corresponding lines in the output with one or more characters specified in the *List* parameter (the default is a tab). If more than one character is in the *List* parameter, then they are repeated in order until the end of the output. In parallel merging, the lines from the last file always end with a new-line character instead of one from the *List* parameter. The following special characters can also be used in the *List* parameter: **\n** New-line character **\t** Tab **** Backslash **\0** Empty string (not a null character) **c** An extended character You must put quotation marks around characters that have special meaning to the shell.
-s	Merges subsequent lines from the first file horizontally. With this flag, the **paste** command works through one entire file before starting on the next. When it finishes merging the lines in one file, it forces a new line and then merges the lines in the next input file, continuing in the same way through the remaining input files, one at a time. A tab separates the lines unless you use the **-d** flag. Regardless of the *List* parameter, the last character of the file is forced to be a new-line character.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To paste several columns of data together, enter:

```
paste names places dates > npd
```

This creates a file named npd that contains the data from the `names` file in one column, the `places` file in another, and the `dates` file in a third. If the `names`, `places`, and `dates` file look like:

```
names           places          dates
rachel          New York        February 5
jerry           Austin          March 13
mark            Chicago         June 21
marsha          Boca Raton      July 16
scott           Seattle         November 4
```

then the npd file contains:

```
rachel          New York        February 5
jerry           Austin          March 13
mark            Chicago         June 21
marsha          Boca Raton      July 16
scott           Seattle         November 4
```

A tab character separates the name, place, and date on each line. These columns do not always line up because the tab stops are set at every eighth column.

2. To separate the columns with a character other than a tab, enter:

```
paste  -d"!@" names places dates > npd
```

This alternates ! and @ as the column separators. If the `names`, `places`, and `dates` files are the same as in example 1, then the npd file contains:

```
rachel!New York@February 5
jerry!Austin@March 13
mark!Chicago@June 21
marsha!Boca Raton@July 16
scott!Seattle@November 4
```

3. To display the standard input in multiple columns, enter:

```
ls | paste - - - -
```

This lists the current directory in four columns. Each `-` (minus) tells the **paste** command to create a column containing data read from the standard input. The first line is put in the first column, the second line in the second column, and so on.

This is equivalent to:

```
ls | paste  -d"\t\t\t\n" -s -
```

This example fills the columns across the page with subsequent lines from the standard input. The `-d"\t\t\t\n"` defines the character to insert after each column: a tab character (`\t`) after the first three columns, and a new-line character (`\n`) after the fourth. Without the **-d** flag, the **paste -s -** command would display all of the input as one line with a tab character between each column.

Files

Item	Description
/usr/bin/paste	Contains the **paste** command.

patch Command

Purpose

Applies changes to files.

Syntax

patch [**-b** [**-B** *Prefix*]] [**-f**] [**-l**] [**-N**] [**-R**] [**-s**] [**-v**] [**-c** | **-e** | **-n** | **-u**] [**-d** *Directory*] [**-D** *Define*] [**-F** *Number*] [**-i** *PatchFile*] [**-o** *OutFile*] [**-p** *Number*] [**-r** *RejectFile*] [**-x** *Number*] [*File*]

Description

The **patch** command reads a source file's instructions on how to change a file, then applies the changes. The source file contains difference listings (or *diff* listings) produced by the **diff -c** or **-u** command, and one or more sets of **diff** command output, customarily called *hunks*.

The **patch** command skips any leading text in a patch file, applies the actual diff listing, and skips any trailing text. Thus, you could use as a patch file or message that includes a diff listing, and the **patch** command would still work. In such a case, if the entire diff listing is indented by a consistent amount, the **patch** command will also adjust for that spacing.

To change a line range within the original file, each hunk within a patch must be a separate diff listing. The line numbers for successive hunks within a patch must occur in ascending order.

File Name Determination

If no *File* parameter is specified, the **patch** command performs the following steps to obtain the name of the file to edit:

1. In the header of a context diff listing,
   - If the type of the diff is copied context, the file name is determined from lines beginning with *** (three asterisks) or —- (three dashes). A line beginning with *** indicates the name of the file from which the patches were taken, while a line beginning with —- indicates the name of the file to which the patches should be applied. The shortest name of an existing file is selected.
   - If the type of the diff is unified context, the file name is determined from lines beginning with --- (three dashes) or +++ (three pluses). A line beginning with --- indicates the name of the file from which the patches were taken, while a line beginning with +++ indicates the name of the file to which the patches should be applied. The shortest name of an existing file is selected.
2. If there is an `Index:` line in the leading text, the **patch** command tries to use the file name from that line.
3. A context diff header takes precedence over an `Index:` line.
4. If no file name can be determined from the leading text, the **patch** command prompts you for the name of the file to patch.
5. If the original file cannot be found, but a suitable SCCS or RCS file is available, the **patch** command attempts to get or check out the file.
6. If the leading text contains a `Prereq:` line, the **patch** command takes the first word from the prerequisites line (normally a version number) and checks the input file to see if that word can be found. If not, the **patch** command prompts you for confirmation before proceeding.

Patch Application

If the patch file contains more than one patch, the **patch** command tries to apply each diff listing as if it came from a separate patch file. In this case, the name of the file to patch is determined for each diff listing, and the header text before each diff listing is examined for information such as file name and revision level.

If you specify the **-c**, **-e**, **-n**, or **-u** flag, the **patch** command interprets information within each hunk as a copied context difference, an ed editor difference, a normal difference, or a unified context difference respectively. Otherwise, the **patch** command determines the type of difference based on the format of the information within the hunk.

The **patch** command searches for the place to apply each hunk by taking the first line number of the hunk and adding or subtracting any line offset caused by applying the previous hunk. If an exact match is not possible at this line location, the **patch** command scans both forward and backward for a set of lines matching the hunk's content exactly.

If no such place is found, and if the **patch** command is applying a context diff listing, the **patch** command can search for a less exact match. A *fuzz factor* specifies how many lines can be inexactly matched. If the fuzz factor is set to 1 or more, the **patch** command performs a second scan, this time ignoring the first and last line of context. If no match results, and the maximum fuzz factor is set to 2 or more, the **patch** command performs a third scan, this time ignoring the first two lines and the last two lines of the context. (The default maximum fuzz factor is 2.) If no match is found, the **patch** command places the hunk in a reject file. The reject file is created with the same name as the output file and the suffix **.rej**. This naming convention can be overridden by using the **-r** flag.

The rejected hunk is written in copied context diff listing form, regardless of the format of the patch file. If the input was a normal or ed editor style difference, the reject file may contain differences with zero lines of copied context format. The line numbers on the hunks in the reject file may be different from the line numbers in the patch file. This is because the reject file line numbers reflect the approximate locations for the failed hunks in the new file rather than the old one.

As each hunk is completed, the **patch** command tells you whether the hunk succeeded or failed. You are also informed of the new line number assumed for each hunk. If this is different from the line number specified in the diff listing, you are notified of the offset. The **patch** command also tells you if a fuzz factor was used to make the match.

> **Note:** A single large offset may be an indication that a hunk was installed in the wrong place. Use of a fuzz factor may also indicate bad placement.

Preparing Patches for Other Users

Programmers preparing patches that will be shipped to other users should consider the following additional guidelines:

- If you try to apply the same patch twice, the **patch** command assumes the second application should be a reverse patch and prompts you for confirmation of this reversal. Therefore, avoid sending out reversed patches, since this makes users wonder whether they already applied the patch.
- It is recommended that you keep a **patchlevel.h** file that is updated with the latest patch level. The patch level can then be used as the first diff listing in the patch file you send out. If your patch includes a `Prereq:` line, users cannot apply patches out of order without receiving a warning.
- Make sure you specify the file names correctly, either in a context diff listing header or with an `Index:` line. If you are patching something in a subdirectory, be sure to tell the patch user to specify a **-p** flag as needed.
- You can create a file by sending out a diff listing that compares a null file to the file you want to create. However, this only works if the file you want to create does not already exist in the target directory.
- While you may be able to put many diff listings into one file, it is advisable to group related patches into separate files.
- The **patch** command cannot tell if the line numbers are incorrect in an ed script, and can only detect bad line numbers in a normal diff listing when it finds a change or a delete command. A context diff listing using a fuzz factor of 3 may have the same line-number problem. Until a suitable interactive interface is added, use a context diff listing in such cases to check the changes for accuracy. Compilation without errors usually means that the patch worked, but it is not an infallible indicator.
- The results of the **patch** command are guaranteed only when the patch is applied to exactly the same version of the file from which the patch was generated.
- If the code has been duplicated, for example:

```
#ifdef
... NEWCODE
#else
... OLDCODE
# endif
```

  the **patch** command is incapable of patching both versions. If the **patch** command succeeds, it may have patched the wrong version and return a successful exit status.

Flags

Item	Description
-b	Saves a copy of each modified file before the differences are applied. The copied original is filed with the same name and the suffix **.orig**. If a file by that name already exists, it is overwritten. If multiple patches are applied to the same file, only one copy is made of the original file at the time of the first patch. If the **-o** *OutFile* flag is also specified, the **.orig** file is not created. But if the specified out file already exists, *OutFile***.orig** is created.
-B *Prefix*	Specifies a prefix to the backup file name. This flag only works in conjunction with the **-b** flag.
-c	Interprets the patch file as a copied context diff listing (the output of the **diff -c** or **diff -C** command). This flag cannot be used with the **-e**, **-n**, or **-u** flag.
-d *Directory*	Changes the current directory to the specified directory before processing.
-D *Define*	Marks changes with the following C preprocessor construct: `#ifdef Define` `...  (NEWCODE)` `#else` `...  (OLDCODE)` `#endif /* Define */` The *Define* variable is used as the differentiating symbol. This flag only works when the normal or context form of diff listing is used as a patch file.
-e	Interprets the patch file as an ed editor script. This flag cannot be used with the **-c**, **-n**, or **-u** flag.
-f	Suppresses queries to the user. To suppress commentary, use the **-s** flag.
-F *Number*	Sets the maximum fuzz factor. This flag applies to context diff listings only and causes the **patch** command to ignore the specified number of lines when determining where to install a hunk. If the **-F** flag is not specified, the default fuzz factor is 2. The factor may not be set to more than the number of lines of content in the context diff listing (ordinarily 3). **Note:** A larger fuzz factor increases the odds of a faulty patch.
-i *PatchFile*	Reads the patch information from the specified file, rather than from standard input.
-l	(lowercase L) Causes any sequence of blank characters in the diff listing script to match any sequence of blank characters in the input file. Other characters are matched exactly.
-n	Interprets the script as a normal diff listing. This flag cannot be used with the **-c**, **-e**, or **-u** flag.
-N	Ignores patches where the differences have already been applied to the file. By default, already-applied patches are rejected.
-o *OutFile*	Copies the files to be patched, applies the changes, then writes the modified version to the specified output file. Multiple patches for a single file are applied to the intermediate versions of the file created by any previous patches. Therefore, multiple patches result in multiple, concatenated versions of the output file.

Item	Description
-p *Number*	Sets the path name strip count, which controls how path names found in the patch file are treated. This flag is useful if you keep your files in a directory different from the specified path. The strip count specifies how many slashes are stripped from the front of the path name. Any intervening directory names are also stripped. For example, assume a patch file specified `/u/leon/src/blurf1/blurf1.c`: • **-p 0** leaves the entire path name unmodified. • **-p 1** removes the leading slash, leaving `u/leon/src/blurf1/blurf1.c`. • **-p 4** removes four slashes and three directories, leaving `blurf1/blurf1.c`. If the **-p** flag is not specified, only the base name (the final path name component) is used. This flag works only when the *File* parameter is not specified.
-r *RejectFile*	Overrides the default reject file name. The default reject file name is formed by appending the suffix **.rej** to the original file name.
-R	Reverses the sense of the patch script. For example, if the diff listing was created from new version to old version, using the **-R** flag causes the **patch** command to reverse each portion of the script before applying it. Rejected differences are saved in swapped format. The **-R** flag cannot be used with ed scripts, because there is too little information to reconstruct the reverse operation. If the **-R** flag is not specified, the **patch** command attempts to apply each portion in its reversed sense as well as in its normal sense, until a portion of the patch file is successfully applied. If the attempt is successful, the user is prompted to determine if the **-R** flag should be set. **Note:** This method cannot detect a reversed patch if used with a normal diff listing where the first command is an append (that is, would have been a delete). Appends always succeed because a null context matches anywhere. Fortunately, most patches add or change lines rather than delete lines. Therefore most reversed normal diff listings begin with a delete, causing a failure and triggering heuristics.
-s	Patches silently unless an error occurs.
-u	Interprets the patch file as a unified context difference (the output of the **diff** command when you specify the **-u** or **-U** flag). You cannot specify this flag with the **-c**, **-e**, or **-n** flag.
-v	Prints the revision header and patch level. If the **-v** flag is used with other flags, the other flags are ignored.
-x *Number*	Sets internal debugging flags. This flag is only for **patch** command developers.

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
1	An error occurred and one or more lines are written to the rejected file.
>1	An error occurred.

Examples

1. To apply diff listings in the `difflisting` file to the `prog.c` file, enter:

   ```
   patch -i difflisting prog.c
   ```

2. To save the original version of the `prog.c` file, enter:

   ```
   patch -b -i difflisting prog.c
   ```

   This applies changes to `prog.c` and saves the original contents of `prog.c` in the file `prog.c.orig`.

3. To patch the `prog.c` file without altering the original version, enter:

   ```
   patch -i difflisting -o prog.new prog.c
   ```

   This uses `prog.c` as a source file, but the changed version is written to a file named `prog.new`.

Files

Item	Description
/usr/bin/patch	Contains the **patch** command.

pathchk Command

Purpose

Checks path names.

Syntax

pathchk [-p] [-P] *pathname...*

Description

The **pathchk** command checks that one or more path names are valid and portable. By default, the **pathchk** command checks each component of each path name specified by the *pathname* parameter based on the underlying file system. An error message is sent for each path name that meets the following criteria:

- The byte length of the full path name is longer than allowed by the system.
- The byte length of a component is longer than allowed by the system.
- Search permission is not allowed for a component.
- A character in any component is not valid in its containing directory.

It is not an error if one or more components of a path name do not exist. If a file that matches the path name specified by the *pathname* parameter can be created and it must not violate any of the above criteria.

More extensive portability checks are run when the `-p` flag is specified.

Flags

Item	Description
`-p`	Checks the path name based on POSIX portability standards. An error message is sent for each path name that meets the following criteria: • The byte length of the full path name is longer than allowed by POSIX standards. • The byte length of a component is longer than allowed by POSIX standards. • A character in any component is not in the portable file name character set.
`-P`	Checks the *pathname* operand and returns an error message if the *pathname* operand meets the following criteria: • The *pathname* operand contains a component whose first character is the hyphen character. • The *pathname* operands are empty.

Exit Status

This command returns the following exit values:

Item	Description
`0`	All *pathname* operands passed all of the checks.
`>0`	An error occurred.

Examples

1. To check the validity and portability of the `/home/bob/work/tempfiles` path name on your system, enter:

```
pathchk /home/bob/work/tempfiles
```

2. To check the validity and portability of the `/home/bob/temp` path name for POSIX standards, enter:

```
pathchk -p /home/bob/temp
```

Files

Item	Description
`/usr/bin/pathchk`	Contains the **pathchk** command.

pax Command

Purpose

Extracts, writes, and lists members of archive files; copies files and directory hierarchies.

Syntax

To List Member Files of Archived Files

pax [**-c** | **-n**] [**-d**] [-U] [**-v**] [**-H** | **-L**] [**-f** *Archive*] [**-s** *ReplacementString*...] [**-x** *Format*] [**-o** *Options*] [**-Z**] [*Pattern*...]

To Extract Archive Files Using the -r Flag

pax -r [**-c** | **-n**] [**-d**] [**-i**] [**-k**] [-U] [**-u**] [**-v**] [**-H** | **-L**] [**-f** *Archive*] [**-o** *Options*] [**-p** *String*...] [**-s** *ReplacementString*...] [**-x** *Format*] [**-Z**] [*Pattern* ...]

To Write Archive Files Using the -w Flag

pax -w [**-d**] [**-i**] [**-t**] [-U] [**-u**] [**-v**] [**-X**] [**-H** | **-L**] [**-E**] [**-b** *Blocking*] [[**-a**] **-f** *Archive*] [**-o** *Options*] [**-s** *ReplacementString*...] [**-x** *Format*] [**-Z**] [*File*...]

To Copy Files Using the -r and -w Flags

pax -r -w [**-d**] [**-i**] [**-k**] [**-l**] [**-t**] [-U] [**-u**] [**-v**] [**-X**] [**-H** | **-L**] [**-p** *String*...] [**-o** *Options*] [**-s** *ReplacementString*...] [**-x** *Format*] [**-Z**] [*File* ...] *Directory*

Description

pax stands for portable archive interchange. The **pax** command extracts and writes member files of archive files; writes lists of the member files of archives; and copies directory hierarchies. The **-r** and **-w** flags specify the type of archive operation.

Note: pax actively sparse files that are being restored. If a file blocks an aligned and sized areas that are NULL populated, **pax** does not cause physical space for those file system blocks to be allocated. The file size in bytes remains the same, but the actual space that is taken within the file system is for the non-NULL areas.

Listing Member Files of Archived Files (List Mode)

When the **-r** flag or the **-w** flag is not specified, the **pax** command lists all the member files of the archive file that is read from standard input. If the *Pattern* parameter is specified, only the member files with path names that match the specified patterns are written to standard output. If a named file is a directory, the file hierarchy that is contained in the directory is also written. When the **-r** flag or the **-w** flag is not specified, the **-c**, **-d**, **-f**, **-n**, **-s**, and **-v** flags, and the *Pattern* parameter can be specified.

Extracting Archive Files Using the -r Flag (Read Mode)

When the **-r** flag is specified, but the **-w** flag is not, the **pax** command extracts all the member files of the archive files that are read from standard input. If the *Pattern* parameter is specified, only the member files with path names that match the specified patterns are written to standard output. If a named file is a directory, the file hierarchy that is contained in the directory is also extracted. The **-r** flag can be specified with the **-c**, **-d**, **-f**, **-i**, **-k**, **-n**, **-s**, **-u**, and **-v** flags, and with the *Pattern* parameter.

The access and modification times of the extracted files are the same as the archived files. The file modes of the extracted files are the same as when they were archived, unless they are affected by the user's default file creation mode (**umask**). The **S_ISUID** and **S_ISGID** bits of the extracted files are cleared.

If intermediate directories are necessary to extract an archive member, the **pax** command creates the directories with access permissions set as the bitwise inclusive OR of the values of the **S_IRWXU**, **S_IRWXG**, and **S_IRWXO** masks.

If the selected archive format supports the specification of linked files, it is an error if these files cannot be linked when the archive is extracted.

Writing Archive Files Using the -w Flag (Write Mode)

When the **-w** flag is specified and the **-r** flag is not, the **pax** command writes the contents of the files that are specified by the *File* parameter to standard output in an archive format. If no *File* parameter is specified, a list of files to copy, one per line, is read from the standard input. When the *File* parameter specifies a directory, all of the files that are contained in the directory are written. The **-w** flag can be specified with the **-a**, **-b**, **-d**, **-f**, **-i**, **-o**, **-s**, **-t**, **-u**, **-v**, **-x**, and **-X** flags and with *File* parameters.

Copying Files Using the -r and -w Flags (Copy Mode)

When both the **-r** and **-w** flags are specified, the **pax** command copies the files that are specified by the *File* parameters to the destination directory specified by the *Directory* parameter. If no files are specified, a list of files to copy, one per line, is read from the standard input. If a specified file is a directory, the file hierarchy that is contained in the directory is also copied. The **-r** and **-w** flags can be specified with the

-d, **-i**, **-k**, **-l**, **-o**, **-p**, **-s**, **-t**, **-u**, **-v**, and **-X** flags and with *File* parameters. The *Directory* parameter must be specified.

Copied files are the same as written to an archive file and are later extracted, except that there are hard links between the original and the copied files.

Modifying the Archive Algorithm Using the -o Flag

Use the **-o** flag to modify the archive algorithm according to keyword-value pairs. The keyword-value pairs must adhere to a correct archive format. A list of valid keywords and their behavior is given in the subsequent description of the **-o** flag.

Further notes

In read or copy modes, if intermediate directories are necessary to extract an archive member, the **pax** command does actions similar to the **mkdir()** subroutine with the intermediate directory used as the path argument and the value **S_IRWXU** as the mode argument.

If any specified pattern or file operands are not matched by at least one file or archive member, **pax** writes a diagnostic message to standard error for each one that did not match and exits with an error status.

In traversing directories, the **pax** command detects the infinite loops by entering a previously visited directory that is an ancestor of the last file visited. Upon detection of an infinite loop, the **pax** command writes a diagnostic message to standard error and terminates.

When **pax** command is in read mode or list mode, by using the **-x pax** archive format, a file name, link name, owner name, or any other field in an extended header record cannot be converted from the **pax** UTF8 code set format to the current code set and locale. The **pax** command writes a diagnostic message to standard error, processes the file as described for the **-o invalid=** option, and then processes the next file in the archive.

For AIX 5.3 the pax command ignores the extended attributes by default. The -U option informs pax to archive or restore extended attributes, which include ACLs. The -pe option preserves ACLs. When the -pe option is specified and if pax fails to preserve the ACLs, a diagnostic message is written to the standard error but the extracted file is not deleted. A nonzero exit code is returned. A new record type is required for extended attribute entries in the pax archive files.

Variables

Item	Description
Directory	Specifies the path of a destination directory when copying files.
File	Specifies the path of a file to be copied or archived. If no file matches the *File* parameter, the **pax** command detects the error, exits, and writes a diagnostic message.
Pattern	Specifies a pattern that matches one or more paths of archive members. A / (backslash) character is not recognized in the *Pattern* parameter and it prevents the subsequent character from having any special meaning. If no *Pattern* parameter is specified, all members are selected in the archive. If a *Pattern* parameter is specified, but no archive members are found that match the pattern that is specified, the **pax** command detects the error, exits, and writes a diagnostic message.

Flags

Item	Description
-a	Appends files to the end of an archive. **Note:** Streaming tape devices do not allow the append function.

Item	Description
-b *Blocking*	Specifies the block size for output. The *Blocking* parameter specifies a positive decimal integer value that specifies the number of bytes per block. Application conforming to POSIX2 must not specify a blocksize value greater than 32256. Devices and archive formats might impose restrictions on blocking. Blocking is automatically determined on input. Default blocking when the archives are created depends on the archive format. (See the **-x** flag definition.) The *Blocking* parameter accepts one of the following values: ***Integer* b** Specifies that the block size, in bytes, be the value of the positive decimal integer that is specified by the *Integer* parameter that is multiplied by 512. ***Integer* k** Specifies that the block size, in bytes, be the value of the positive decimal integer that is specified by the *Integer* parameter that is multiplied by 1024. ***Integer* m** Specifies that the block size, in bytes, be the value of the positive decimal integer that is specified by the *Integer* parameter that is multiplied by 1024 x 1024. ***Integer+Integer*** Specifies that the block size, in bytes, be the sum of the positive decimal integers that are specified by the *Integer* parameters.
-c	Matches all file or archive members except the files that are specified by the *Pattern* parameter.
-d	Causes directories being copied, archived, or extracted, to match the directory and not the contents of the directory.
-E	Avoids truncation of the long user and group names during addition of files to a new or existing archive.
-f *Archive*	Specifies the path of an archive file to be used instead of standard input (when the **-w** flag is not specified) or standard output (when the **-w** flag is specified but the **-r** flag is not). When specified with the **-a** flag option, any files that are written to the archive are appended to the end of the archive.
-H	If a symbolic link that refers to a directory is specified on the command line, **pax** archives the file hierarchy that is rooted in the directory that is referenced in the link, by using the name of the link as the name of the file hierarchy. By default, **pax** archives the symbolic link itself.
-i	Renames files or archives interactively. For each archive member that matches the *Pattern* parameter or file that matches a *File* parameter, a prompt is written to the display device that contains the name of a file or archive member. A line is then read from the display device. If this line is empty, the file or archive member is skipped. If this line consists of a single period, the file or archive member is processed with no modification to its name. Otherwise, its name is replaced with the contents of the line.
-k	Prevents the **pax** command from writing over existing files.
-l	Links files when copying files. Hard links are established between the source and destination file hierarchies whenever possible.

Item	Description
-L	If a symbolic link that refers to a directory is specified on the command line or encountered during the traversal of a file hierarchy, **pax** archives the file hierarchy that is rooted in the directory that is referenced in the link, by using the name of the link as the name of the file hierarchy. By default, **pax** archives the symbolic link itself.
-n	Selects the first archive member that matches each *Pattern* parameter. No more than one archive member is matched for each pattern.

Item	Description
-o *Options*	Modifies the archiving algorithm according to the keyword-value pairs specified in the *Options* parameter. The keyword-value pairs must be in the following format: *keyword:=value,keyword:=value,...* Some keywords apply only to certain file formats, as indicated with each description. Use of keywords that are inapplicable to the file format being processed will be ignored by **pax**. Keywords can be preceded with white space. The *value* field consists of zero or more characters; within *value*, any literal comma must be preceded with a backslash (\). A comma as the final character, or a comma that is followed by white space as the final character, in *Options* is ignored. Multiple **-o** options can be specified. If keywords given to these multiple **-o** options conflict, the keywords and values that appear later in the command-line sequences take precedence. The earlier values are ignored. The following keyword-value pairs are supported for the indicated file formats: **datastream**=*pathname*,**datastr_size**=*size* (Applicable to all file formats.) The **datastream** keyword indicates that the incoming archive file is not in a file format; instead, it is a DataStream from the standard input device. Consequently, the data must be archived as a regular file in a format that is recognized by the **-x** flag. The file name of the DataStream must be specified in the *pathname* parameter and must include the identification of the person who invoked the command, the group identification, and the **umask** for the file mode. **Note:** The **datastream** keyword does not have a default variable size. You must specify one. The **datastr_size** keyword denotes the size of the DataStream input in bytes by using decimal digits. If the **pax** command reaches the end of file (EOF) character before it reads the *size* parameter, it pads the archive file with null values. The null values make the archive file the same size as specified by the *size* parameter. If the data in the archive file exceeds the size that is specified, the **pax** command truncates the archive file to the size specified by the *size* parameter. The **pax** command also stops taking input and closes the archive file. **Note:** You can specify multiple instances of keyword pairs. If you assign different values to the same keyword, the **pax** command uses the last value that is assigned to the keyword to run the **-o** flag. **delete**=*pattern* (Applicable only to the **-x pax** format.) When used in write or copy mode, **pax** omits any keywords matching *pattern* from the extended header records that it produces. When used in read or list mode, **pax** ignores any keywords matching *pattern* in the extended header records. In all cases, matching is done using standard shell pattern-matching notation. For example, `-o delete=security.*` suppresses security-related information.

Item	Description
-o *Options (Continued)*	

exthdr.name=*string* (Applicable only to the **-x pax** format.)

This keyword allows user control over the name written into the **ustar** header blocks for the extended header records. The name is the contents of *string* after the following character substitutions have been made:

***string* includes:**
Replaced by:

%d
The directory name of the file, equivalent to the result of the **dirname** utility on the transformed path name

%f
The filename of the file, equivalent to the result of the **basename** utility on the transformed path name

%%
A %% character

Any other % characters in *string* produce undefined results. If this keyword-value pair is not specified in the **-o** *Options* list, the default value of the name is:

%d/PaxHeaders/%f

globexthdr.name=*string* (Applicable only to the **-x pax** format.)

When used in write or copy mode with the appropriate options, **pax** creates global extended header records with **ustar** header blocks that will be treated as regular files by previous versions of **pax**. This keyword allows user control over the name that is written into the **ustar** header blocks for global extended header records. The name is the contents of *string* after the following character substitutions have been made:

***string* includes:**
Replaced by:

%n
An integer that represents the sequence number of the global extended header record in the archive starting at 1

%%
A % character

Any other % characters in *string* produce undefined results. If this keyword-value pair is not specified in the **-o** *Options* list, the default value of the name is

$TMPDIR/GlobalHead.%n

where **$TMPDIR** is either the value of the **TMPDIR** environment variable or **/tmp** if **TMPDIR** is unset.

invalid=*action* (Applicable only to the **-x pax** format.)

This keyword allows user control over the action **pax** takes upon encountering values in an extended header record that:

- in read or copy mode, are invalid in the destination hierarchy, or
- in list mode, cannot be written in the code set and current locale.

Item	Description
-o *Options (Continued)*	**pax** recognizes these invalid values: • In read or copy mode, a file name or link name that contains character encodings invalid in the destination hierarchy. (For example, the name may contain embedded NULLs.) • In read or copy mode, a file name or link name that is longer than the maximum allowed in the destination hierarchy (for either a path name component or the entire path name). • In list mode, any character string value (file name, link name, username, and so on) that cannot be written in the code set and current locale. These mutually exclusive values of the *action* argument are supported: • **bypass** In read or copy mode, **pax** bypasses the file, causing no change to the destination hierarchy. In list mode, **pax** writes all requested valid values for the file, but its method for writing invalid values is unspecified. • **rename** In read or copy mode, **pax** acts as if the **-i** flag is in effect for each file with invalid file name or link name values, allowing the user to provide a replacement name interactively. In list mode, **pax** behaves identically to the **bypass** action. • **UTF8** When used in read, copy, or list mode and a file name, link name, owner name, or any other field in an extended header record cannot be translated from the **pax UTF8** code set format to the current code set and locale, **pax** uses the actual UTF8 encoding for the name. • **write** In read or copy mode, **pax** writes the file, translating or truncating the name, regardless of whether this may overwrite an existing file with a valid name. In list mode, **pax** behaves identically to the **bypass** action. If no **-o invalid=***action* is specified, **pax** acts as if the **bypass** action is specified. Any overwriting of existing files that may be allowed by the **-o invalid=***actions* is subject to permission (**-p**) and modification time (**-u**) restrictions, and is suppressed if the **-k** flag is also specified. **linkdata** (Applicable only to the **-x pax** format.) In write mode, the **pax** command writes the contents of a file to the archive, even when that file is a hard link to a file whose contents are written to the archive.

Item	Description
-o *Options (Continued)*	**listopt**=*format* (Applicable to all file formats.) This keyword specifies the output format of the table of contents that are produced when the **-v** option is specified in list mode. To avoid ambiguity, this keyword-value pair must be used as the only or final keyword-value pair following the **-o** flag; all characters in the remainder of the option-argument are considered part of the format string. If multiple **-o listopt=format** options are specified, the format strings are considered to be a single, concatenated string, evaluated in command-line order. See the **List-Mode Format Specifications** section for more information. **times** (Applicable only to the **-x pax** format.) When used in write or copy mode, **pax** includes atime, ctime, and mtime extended header records for each file.

Extended header keywords

(Applicable only to the **-x pax** format.)

If the **-x pax** format is specified, the keywords and values that are defined in the list below can be used as parameters to the **-o** flag, in either of two modes:

keyword=value

When used in write or copy mode, these keyword-value pairs are written into the global extended header records of the new archive. When used in read or list mode, these keyword-value pairs act as if they were present in the global extended header records of the archive that is being read. In both cases, the given value is applied to all files that do not have a value that is assigned in their individual extended header records for the specified keyword.

*keyword***:=***value*

When used in write or copy mode, these keyword-value pairs are written into the extended header records of each file in the new archive. When used in read or list mode, these keyword-value pairs act as if they were present in the extended header records of each file in the archive that is being read. In both cases, the given value overrides any value for the specified keyword that is found in global or file-specific extended header records.

atime

The file access time for the following files, equivalent to the value of the st_atime member of the stat structure for a file.

charset

The name of the character set to encode the data in the following files. The entries in this table are defined to refer to known standards:

value	**Formal Standard**
"ISO-IR 646 1990"	ISO/IEC 646 IRV
"ISO-IR 8859 1 1987"	ISO 8859-1
"ISO-IR 8859 2 1987"	ISO 8859-2
"ISO-IR 10646 1993"	ISO/IEC 10646
"ISO-IR 10646 1993 UTF8"	ISO/IEC 10646, UTF8 encoding
"BINARY"	None

The encoding is included in an extended header for information only; when **pax** is used as described, it does not translate the file data into any other encoding. The BINARY entry indicates binary data that is not encoded.

comment

A series of characters used as a comment. All characters in the value field are ignored by **pax**.

ctime

The file creation time for the following file(s), equivalent to the value of the st_ctime member of the stat structure for a file.

gid

The group ID of the group that owns the file, expressed as a decimal number by using digits from ISO/IEC 646. This record overrides the *gid* field in the following header block(s). When used in write or copy mode, **pax** includes a gid extended header record for each file whose group ID is greater than 99,999,999.

gname

The group of the following file(s), formatted as a group name in the group database. This record overrides the *gid* and *gname* fields in the following header blocks, and any *gid* extended header record. When used in read, copy, or list mode, **pax** translates the name from the UTF8 encoding in the header record to the character set appropriate for the group database on the receiving system. If any of the UTF8 characters cannot be translated, and if the **-o invalid=UTF8** option is not specified, the results are undefined. When used in write or copy mode, **pax** includes a gname extended header record for each file whose group name cannot be represented entirely with the letters and digits of the portable character set.

hdrcharset

The name of the character set that is used to encode the value field of the `gname`, `linkpath`, `path`, and `uname` extended header records. The entries in the following table are defined to refer to the known standards. Additional names might be agreed between the originator and the recipient.

value	**Formal Standard**
ISO-IR106462000UTF-8	ISO/IEC 10646, UTF-8 encoding
BINARY	None

If the `hdrcharset` extended header record is not specified, the default character set (ISO/IEC 10646-1:2000 standard UTF-8 encoding) is used to encode all values in the extended header records.

The **BINARY** entry indicates that all the values that are recorded in the extended headers for affected files are unencoded binary data from the underlying system.

linkpath

The path name of a link that is created to another file, of any type, previously archived. This record overrides the *linkname* field in the following **ustar** header block(s).

The following **ustar** header block determines the type of link that is created, whether hard or symbolic. In the latter case, the linkpath value is the contents of the symbolic link. **pax** translates the name of the link (contents of the symbolic link) from the UTF8 encoding to the character set appropriate for the local file system.

When used in write or copy mode, **pax** includes a link path extended header record for each link whose path name cannot be represented entirely with the members of the portable character set other than NULL.

mtime

The file modification time of the following file(s), equivalent to the value of the st_mtime member of the stat structure for a file. This record overrides the *mtime* field in the following header block(s). The modification time is restored if the process has the appropriate privilege to do so.

path

The pathname of the following file(s). This record overrides the *name* and *prefix* fields in the following header block(s). **pax** translates the path name of the file from the UTF8 encoding to the character set appropriate for the local file system. When used in write or copy mode, **pax** includes a path extended header record for each file whose path name cannot be represented entirely with the members of the portable character set other than NULL.

realtime.*any*

The keywords that are prefixed by real time are reserved for future POSIX real-time standardization. **pax** recognizes but silently ignores them.

security.*any*

The keywords that are prefixed by security are reserved for future POSIX security standardization. **pax** recognizes but silently ignores them.

size

The size of the file in octets, expressed as a decimal number using digits from ISO/IEC 646. This record overrides the *size* field in the following header block(s). When used in write or copy mode, **pax** includes a size of extended header record for each file with a size value greater than 999,999,999,999.

uid

The user ID of the user that owns the file, expressed as a decimal number using digits from ISO/IEC 646.. This record overrides the *uid* field in the following header block(s). When used in write or copy mode, **pax** includes a uid extended header record for each file whose owner ID is greater than 99,999,999.

uname

The owner of the following file(s), formatted as a user name in the user database. This record overrides the *uid* and *uname* fields in the following header block(s), and any *uid* extended header record. When used in read, copy, or list mode, **pax** translates the name from the UTF8 encoding in the header record to the character set appropriate for the user database on the receiving system. If any of the UTF8 characters cannot be translated, and if the **-o invalid=UTF8** option is not specified, the results are undefined. When used in write or copy mode, **pax** includes a uname extended header record for each file whose user name cannot be represented entirely with the letters and digits of the portable character set.

If the *value* field is zero length, it deletes any header block field, previously entered extended header value, or global extended header value of the same name.

If a keyword in an extended header record (or in a **-o** option-argument) overrides or deletes a corresponding field in the **ustar** header block, **pax** ignores the contents of that header block field.

Extended header keyword precedence

(Applicable only to the **-x pax** format.)

This section describes the precedence in which the various header records and fields and command-line options are selected to apply to a file in the archive. When **pax** is used in read or list modes, it determines a file attribute in this sequence:

1. If **-o delete**=*keyword-prefix* is used, the affected attribute is determined from step (7) if applicable, or ignored otherwise.
2. If **-o keyword**:=NULL is used, the affected attribute is ignored.
3. If **-o keyword**:=*value* is used, the affected attribute is assigned the value.
4. If *value* exists in a file-specific extended header record, the affected attribute is assigned the value. When extended header records conflict, the last one given in the header takes precedence.
5. If **-o keyword**=*value* is used, the affected attribute is assigned the value.
6. If a value exists in a global extended header record, the affected attribute is assigned the value. When global extended header records conflict, the last one given in the global header takes precedence.
7. Otherwise, the attribute is determined from the **ustar** header block.

Item	Description
-p *String*	Specifies one or more file characteristics to be retained or discarded on extraction. The *String* parameter consists of the characters **a**, **e**, **m**, **o**, and **p**. Multiple characteristics can be concatenated within the same string and multiple **-p** flags can be specified. The specifications have the following meanings: **a** Does not retain file-access times. **e** Retains the user ID, group ID, file mode, access time, modification time, and ACLs. **m** Does not retain file-modification times. **o** Retains the user ID and the group ID. **p** Retains the file modes. If neither the **-e** nor the **-o** flag is specified, or the user ID and group ID are not preserved for any reason, the **pax** command does not set the **S_ISUID** and **S_ISGID** bits of the file mode. If the retention of any of these items fails, the **pax** command writes a diagnostic message to standard error. Failure to retain any of the items affects the exit status, but does not cause the extracted file to be deleted. If specification flags are duplicated or conflict with each other, the last flag that is specified takes precedence. For example, if `-p eme` is specified, file-modification times are retained.
-r	Reads an archive file from the standard input.
-s *ReplacementString*	Modifies file or archive-member names that are specified by the *Pattern* or *File* parameters according to the substitution expression *ReplacementString*, by using the syntax of the **ed** command. The substitution expression has the following format: **-s /*old*/*new*/[gp]** where (as in the **ed** command), *old* is a basic regular expression and *new* can contain an **&** (ampersand), **n* (**n** is a digit) back references, or subexpression matching. The *old* string can also contain new-line characters. Any non-null character can be used as a delimiter (the / (backslash) is the delimiter in the example). Multiple **-s** flag expressions can be specified; the expressions are applied in the order specified, terminating with the first successful substitution. The optional trailing g character performs as in the **ed** command. The optional trailing p character causes successful substitutions to be written to standard error. File or archive-member names that substitute to the empty string are ignored when reading and writing archives.
-t	Causes the access times of input files to be the same as they were before being read by the **pax** command.

Item	Description
-U	Performs archival and extraction of ACL and Extended Attributes. Attributes include Access control list (ACL) also. If the ACL type is not supported on the *Target* filesystem then it is converted to the ACL type supported by the *Target* filesystem. If the EA is not supported on the filesystem then it is not copied. When listing members of the archive this option will list the names of any named extended attributes and the type of any ACLs associated with each file that are part of the archive image.
-u	Ignores files that are older than a preexisting file or archive member with the same name. • When extracting files, an archive member with the same name as a file in the file system is extracted if the archive member is newer than the file. • When writing files to an archive file, an archive member with the same name as a file in the file system is superseded if the file is newer than the archive member. If the **-a** flag is specified this is accomplished by appending to the archive. Otherwise it is unspecified if this is accomplished by actual replacement in the archive or by appending to the archive. • When copying files to a destination directory, the file in the destination hierarchy is replaced by the file in the source hierarchy or by a link to the file in the source hierarchy if the file in the source hierarchy is newer.
-v	Writes information about the process. If neither the **-r** or **-w** flags are specified, the **-v** flag produces a verbose table of contents; otherwise, archive member path names are written to standard error.
-w	Writes files to the standard output in the specified archive format.

Item	Description
-x *Format*	Specifies the output archive format with the default format being **ustar**. The **pax** command recognizes the following formats: **pax** The **pax** interchange format. The default blocking value for this format for character-special archive files is 10240. Blocking values of 512 - 32256 in increments of 512 are supported. **cpio** Extended **cpio** interchange format. The default blocking value for this format for character-special archive files is 5120. Blocking values of 512 -32256 in increments of 512 are supported. **ustar** Extended **tar** interchange format. The default blocking value for this format for character-special archive files is 10240. Blocking values of 512 -32256 in increments of 512 are supported. • **Filename**: The **pax** command supports the length of the path and the file name until the PATH_MAX limit that is defined by the system is reached. If the length of the path and the file name input exceeds the PATH_MAX limit, then the values are not archived. • **gid or uid**: The **pax** command supports the values of gid and uid until the UINTMAX limit is reached. Values greater than the UINTMAX limit are truncated. If you attempt to append an archive file with a format that is different from the existing archive format causes the **pax** command to exit immediately with a nonzero exit status. In copy mode, if no **-x** format is specified, **pax** behaves as if **-x pax** were specified.
-X	When traversing the file hierarchy specified by a pathname, the **pax** command does not descend into directories that have a different device ID.
-Z	Archives the Encrypted File System (EFS) information of encrypted files or directories. The EFS information is extracted by default. When members of the archive are listed, an **e** indicator is displayed after the file mode for encrypted files and directories that were archived with the **-Z** flag, and a hyphen (-) is displayed for other files. **Note:** Archives created with the **-Z** flag can be restored only on AIX 6.1 or later releases.

Flag Interaction and Processing Order

The flags that operate on the names of files or archive members (**-c**, **-i**, **-n**, **-s**, **-u**, and **-v**) interact as follows:

- When extracting files, archive members are selected according to the user-specified *pattern* parameters as modified by the **-c**, **-n**, and **-u** flags. Then, any **-s**, and **-i** flags modify, in that order, the names of the selected files. The **-v** flag writes the names resulting from these modifications.
- When writing files to an archive file, or when copying files, the files are selected according to the user-specified pathnames as modified by the **-n** (this option is not valid for Copy Mode) and **-u** flags. Then, any **-s**, and **-i** flags modify, in that order, the names resulting from these modifications. The **-v** flag writes the names resulting from the modification.

- If both the **-u** and **-n** flags are specified, the **pax** command does not consider a file selected unless it is newer than the file to which it is compared.

List Mode Format Specifications

In list mode with the **-o listopt=***format* option, the format argument is applied for each selected file. **pax** appends a newline character to the **listopt** output for each selected file. The format argument is used as the format string described in **printf()**, with the following exceptions:

1. The sequence *keyword* can occur before a format conversion specifier. The conversion argument is defined by the value of *keyword*. The following keywords are supported:
   - Any of the field name entries for **ustar** and **cpio** header blocks.
   - Any keyword defined for the extended header or provided as an extension within the extended header.

   For example, the sequence `%(charset)s` is the string value of the name of the character set in the extended header.

   The result of the keyword conversion argument is the value from the applicable header field or extended header, without any trailing NULLs.

   All keyword-values used as conversion arguments are translated from the UTF8 encoding to the character set appropriate for the local file system, user database, etc., as applicable.
2. An additional conversion character, **T**, specifies time formats. The **T** conversion character can be preceded by the sequence *keyword=subformat*, where *subformat* is a date format allowed by the **date** command. The default keyword is **mtime** and the default subformat is: %b %e %H:%M %Y.
3. An additional conversion character, **M**, specifies the file mode string as displayed by the **ls -l** command. If *keyword* is omitted, the **mode** keyword is used. For example, %.1M writes the single character corresponding to the *entry type* field of the **ls -l** command.
4. An additional conversion character, **D**, specifies the device for block or special files, if applicable. If not applicable and *keyword* is specified, then this conversion is equivalent to %*keyword* u. If not applicable and *keyword* is omitted, this conversion is equivalent to <space>.
5. An additional conversion character, **F**, specifies a pathname. The **F** conversion character can be preceded by a sequence of comma-separated keywords:

   keyword,keyword...

   The values for all the non-null keywords are concatenated together, each separated by a /. The default is *path* if the keyword path is defined; otherwise, the default is *prefix,name*.
6. An additional conversion character, **L**, specifies a symbolic link expansion. If the current file is a symbolic link, then %L expands to:

   "%s -> %s", *value_of_keyword*, *contents_of_link*

   Otherwise, the %L conversion character is equivalent to %F.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To copy the `olddir` directory hierarchy to `newdir`, enter:

   ```
   mkdir newdir
   ```

   ```
   pax -rw olddir newdir
   ```

2. To copy the contents of the current directory to the tape drive, enter:

   ```
   pax -wf /dev/rmt0
   ```

3. To archive the file `xxx` as XXX and display the successful substitution, enter one of the following commands:

   - ```
     pax -wvf/dev/rfd0 -s /xxx/XXX/p xxx
     ```
   - ```
     pax -wvf/dev/rfd0 -s/x/X/gp xxx
     ```

4. To read a file from a standard input and dump it to a datastream file with a specified size, enter:

   ```
   dd if=/dev/hd6 bs=36b count=480 | pax -wf /dev/rfd0 -o
   datastream=_filename_,datastr_size=_size_
   ```

5. To list the files in an archive `pax.ar` in a specified format, enter:

   ```
   pax -v -o listopt="start %F end" -f pax.ar
   ```

6. To create an archive `pax.ar` in `pax` format, enter :

   ```
   pax -wf pax.ar -x pax file1
   ```

7. To extract a file from an archive `pax.ar` in `pax` format with a new path, enter :

   ```
   pax -rvf pax.ar -x pax -o path=newfilename
   ```

8. To copy the contents of a symbolic link from source to destination, enter:

   ```
   pax -rwL srclink destdir
   ```

9. To extract files from the archive with group name as `bin`, enter:

   ```
   pax -rvf pax.ar -x pax -o gname=bin
   ```

10. To ignore the path name from the archive in `pax` format during extraction, enter:

    ```
    pax -rvf pax.ar -o delete=path
    ```

11. To avoid the truncation of long user and group names while creating the archive, enter:

    ```
    pax -wEf file.pax file
    ```

12. To copy the `olddir` directory hierarchy to `newdir` with ACL and EA associated with the files, enter:

    ```
    mkdir newdir
    pax -rUw olddir newdir
    ```

Files

Item	Description
/usr/bin/pax	Contains the **pax** command.

pcat Command

Purpose

Unpacks files and writes them to standard output.

Syntax

pcat *File* ...

Description

The **pcat** command reads the files designated by the *File* parameter, unpacks them, and writes them to standard output. Whether or not the specified file ends in the **.z** characters, the **pcat** command assumes that the file is packed and unpacks it.

The exit value of the **pcat** command is the number of files it was unable to unpack. A file cannot be unpacked if any of the following occurs:

- The file name (exclusive of **.z**) has more than 253 bytes.
- The file cannot be opened.
- The file is not a packed file.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To display compressed files, enter:

   ```
   pcat chap1.z chap2 | pg
   ```

   This command sequence displays the compressed files `chap1.z` and `chap2.z` on the screen in expanded form, a page at a time (`| pg`). Note that the **pcat** command accepts files with and without the `.z` characters.

2. To use a compressed file without expanding the copy stored on disk, enter:

   ```
   pcat chap1.z | grep 'Greece'
   ```

   This command sequence prevents the **pcat** command from displaying the contents of `chap1.z` in its expanded form and pipes it to the **grep** command.

File

Item	Description
/usr/bin/pcat	Contains the **pcat** command.

pdelay Command

Purpose

Enables or reports the availability of delayed login ports.

Syntax

pdelay [**-a**] [*Device*]

Description

The **pdelay** command enables delayed ports. Delayed ports are enabled like shared ports, except that the login herald is not displayed until you type one or more characters (usually carriage returns). If a port is directly connected to a remote system or connected to an intelligent modem, it is enabled as a delayed port to prevent the **getty** command from talking to a **getty** on the remote side or to the modem on a local connection. This action conserves system resources and is equivalent to **pdelay enabled=delay**. If you do not specify a *Device* parameter, the **pdelay** command reports the names of the currently enabled ports.

Use the *Device* parameter to specify the ports to be enabled. Permitted values include:

- Full device name, such as the **/dev/tty1** device
- Simple device name, such as the **tty1** device
- A number (for example, 1 to indicate the **/dev/tty1** device)

Note: You must have root user authority to run this command.

Flags

Item	Description
-a	Enables all ports as delayed.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

To display the names of the delayed ports that are currently enabled, enter:

```
pdelay
```

Files

Item	Description
/etc/locks	Contains **lock** files for the **pshare** and **pdelay** commands.

Item	Description
/usr/sbin/pdelay	Contains the **pdelay** command.

pdisable Command

Purpose

Disables login ports.

Syntax

pdisable [**-a**] [*Device*]

Description

The **pdisable** command disables a specific port, even if a user is logged in at that port. The system disables a port by updating an entry in the **/etc/inittab** file and then sending a signal to the **init** process**.** When the **init** process receives the signal and reads the updated status entry, it takes the appropriate action.

Use the *Device* parameter to specify the ports to be disabled. Permitted values include:

- A full device name, such as the **/dev/tty1** device
- A simple device name, such as the **tty1** device
- A number (for example, 1 to indicate the **/dev/tty1** device).

If you do not specify a *Device* parameter, the **pdisable** command reports the names of currently disabled ports in its set.

Note: You must have root user authority to run this command.

Flag

Item	Description
-a	Disables all ports that are currently enabled.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the names of all ports currently disabled, enter:

   ```
   pdisable
   ```

2. To disable all ports that are enabled, even if users are logged in, enter:

   ```
   pdisable -a
   ```

3. To disable the workstation attached to the **/dev/tty8** port, enter:

   ```
   pdisable tty8
   ```

Files

Item	Description
/etc/locks	Contains **lock** files for the **pshare** and **delay** commands.
/usr/sbin/pdisable	Contains the **pdisable** command.

pdlink Command

Purpose

Links files in partitioned sub directories.

Syntax

pdlink *dirname filename ...*

Description

The **pdlink** command allows you to make a file that exists under a partitioned subdirectory accessible to the processes running at different SLs. The file corresponds to the sensitivity label (SL) of the invoking process. The directory name that you specify using the *dirname* parameter must a partitioned directory, and the file name that you specify using the *filename* parameter must be a file name (not a path name) under that named directory. You can specify multiple file names.

The **pdlink** command creates a hard link to the file specified, with the following qualifications:

- The link is only created in the partitioned subdirectories.
- Each partitioned subdirectory must exist at the time the **pdlink** command is running.
- The link is only created in partitioned subdirectories that have an SL that is higher than the minimum SL of the file specified by the *filename* parameter.

Security

Only authorized users can run the **pdlink** command.

Item	Description
aix.mls.pdir.link	Required to create links in partitioned sub directories with this command.

Exit Status

The **pdlink** command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To create a link of the **sample.c** file, present in the partitioned directory called **partdir**, enter:

```
pdlink partdir sample.c
```

Files

Item	Description
/usr/sbin/pdlink	Contains the **pdlink** command.

pdmkdir Command

Purpose

Creates partitioned directories.

Syntax

pdmkdir [**-m** *Mode*] [**-u** *Owner*] [**-g** *Group*] *dirname* ...

Description

The **pdmkdir** command creates partitioned directories that you specify using the *dirname* parameter. Normal users can create partitioned directories if the Discretionary Access Control (DAC), the Mandatory Access Control (MAC) and the Mandatory Integrity Control (MIC) permissions allow the user to create the new directory. Users with the **aix.mls.pdir.mkdir** authorization can override the DAC, MAC and MIC permissions.

Flags

Item	Description
-g *Group*	Sets the group of the newly-created directories. You can specify either a group name or group ID. Users with the **aix.mls.pdir.mkdir** authorization can change the group of the directory to a group that they are not members of.
-m *Mode*	Sets the permission bits for the newly created directories to the value that is specified by the *Mode* variable. Specify the *Mode* variable as a numeric value.
-u *Owner*	Sets the owner of the newly created directories. You can specify either the owner name or owner ID. Users with the **aix.mls.pdir.mkdir** authorization can change the owner of the directory.

Note: The *Mode*, *Owner* or *Group* variable that is set is applied to the partitioned directory and the partitioned subdirectory created based on the processes Sensitivity Level (SL) which ran the command. If another process with a different SL accesses the partitioned directory, the partitioned subdirectory that is created cannot be governed by these flags.

Security

All users can run the **pdmkdir** command. To successfully perform specific functions, users need the following authorization:

Item	Description
aix.mls.pdir.mkdir	Required to change the owner or group using the **-u** or **-g** flag. This authorization is also required to create directories in a path that ignores the DAC, MAC and MIC permissions of the parent directory.

Exit Status

The **pdmkdir** command returns the following exit values:

Item	Description
0	The command ran successfully and made all requested changes.
>0	An error occurred.

Examples

1. To create a partitioned directory, enter:

```
pdmkdir partdir
```

2. To create a partitioned directory with the permission "755", user "joe", group "staff", enter:

```
pdmkdir –m 755 –u joe –g staff partdir
```

Files

Item	Description
/usr/sbin/pdmkdir	Contains the **pdmkdir** command.

pdmode Command

Purpose

Invokes a command in the virtual or real partitioned, directory-access mode.

Syntax

pdmode [[**-r**] *command* [*arg* ...]]

Description

The **pdmode** command allows you to invoke a command that you specify using the *command* parameter in the virtual or real partitioned directory access mode. When invoked without any argument, the **pdmode** command returns the partitioned directory access mode of the process which invoked this command.

If you run the **pdmode** command followed by the *command* parameter without any flag, the command is run in the virtual mode. A user can run a command in the real partitioned directory access mode by using the **-r** flag.

Flags

Item	Description
-r *command* [*arg*...]	Sets the new process's partitioned directory access mode to the real mode. In this mode, partitioned directories are not transparent, and you must be aware of partitioned directories to navigate the subtree at a partitioned directory. To successfully run the command with this option, users need the **aix.mls.pdir.mode** authorization.

Security

All users can run the **pdmode** command. To successfully perform specific functions, you need the following authorization:

Item	Description
aix.mls.pdir.mode	Required to use the **pdmode** command with the **-r** flag.

Exit Status

The **pdmode** command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To get the partitioned directory access mode, enter:

   ```
   pdmode
   ```

2. To run the **ls** command in the virtual mode, enter:

   ```
   pdmode ls -l
   ```

3. To run the **ls** command in the real mode, enter:

   ```
   pdmode –r ls -l
   ```

Files

Item	Description
/usr/sbin/pdmode	Contains the **pdmode** command.

pdrmdir Command

Purpose

Deletes partitioned directories.

Syntax

pdrmdir *dirname* ...

Description

The **pdrmdir** command deletes partitioned directories that you specify using the *dirname* parameter. Normal users can delete partitioned directories if the Discretionary Access Control (DAC), the Mandatory Access Control (MAC) and the Mandatory Integrity Control (MIC) permissions allow the user to delete the directory. Authorized users with the **aix.mls.pdir.rmdir** authorization can override the DAC, MAC and MIC permissions.

The **pdrmdir** command removes only empty partitioned subdirectories and does not remove files or directories within partitioned subdirectories. The partitioned directory is removed after all the partitioned subdirectories are removed and the directory is empty. The removal of partitioned directory fails if a file exists.

Security

All users can execute the **pdrmdir** command. To successfully perform specific functions, users need the following authorization:

Item	Description
aix.mls.pdir.rmdir	Required to remove directories in a path ignoring the DAC, MAC and MIC permissions.

Exit Status

The **pdrmdir** command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To delete a partitioned directory, enter:

```
pdrmdir partdir
```

Files

Item	Description
/usr/sbin/pdrmdir	Contains the **pdrmdir** command.

pdset Command

Purpose

Converts normal directories to partitioned directories.

Syntax

pdset *dirname* ...

Description

The **pdset** command converts normal directories that you specify using the *dirname* parameter to partitioned directories.

The directory names that you specify cannot be a partitioned subdirectory or a partitioned sub-subdirectory. Existing subdirectories or files under this directory can only be accessible in the real mode of the partitioned directory.

Security

Only authorized users can run the **pdset** command.

Item	Description
aix.mls.pdir.set	Required for converting normal directories to partitioned directories.

Exit Status

The **pdset** command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To convert a directory to a partitioned directory, enter:

```
pdset testdir
```

Files

Item	Description
/usr/sbin/pdset	Contains the **pdset** command.

penable Command

Purpose

Enables or reports the availability of login ports.

Syntax

penable [**-a**] [*Device*]

Description

The **penable** command enables normal ports. Normal ports are asynchronous and only allow users to log in. No outgoing use of the port is allowed while it is enabled. The system enables a port by updating an entry in the **/etc/inittab** file and then sending a signal to the **init** process. After receiving the signal and reading the updated status entry, the process takes the appropriate action.

Use the *Device* parameter to specify the ports to be enabled. Permitted values include:

- Full device name, such as the **/dev/tty1** device
- Simple device name, such as the **tty1** device
- A number (for example, 1 to indicate the **/dev/tty1** device).

If you do not specify a *Device* parameter, the **penable** command reports the names of the currently enabled normal ports.

Note: You must have root user authority to run this command.

Flags

Item	Description
-a	Enables all normal ports.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

To enable all normal ports listed in the **/etc/inittab** file, enter:

```
penable -a
```

Files

Item	Description
/etc/locks	Contains **lock** files for the **pshare** and **pdelay** commands.
/usr/sbin/penable	Contains the **penable** command.

perfwb Command

Purpose

Starts the Performance Workbench to monitor system activity

Syntax

perfwb

Note: The DISPLAY environment variable must be set.

Description

The **perfwb** command is used to start the Performance Workbench. It is a graphical interface to monitor the system activity and processes.

A panel shows the partition configuration and the CPU and memory consumptions.

Another panel lists the top processes that can be sorted on the different provided metrics. A filtering device is also provided to restrict the list to particular processes.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Location

/usr/bin/perfwb

Files

Item	Description
/usr/bin/perfwb	Contains the **perfwb** command.

Item	Description
$HOME/workspace	Contains the **perfwb** working directory that contains preferences.

pg Command

Purpose

Formats files to the display.

Syntax

pg [**-** *Number*] [**-c**] [**-e**] [**-f**] [**-n**] [**-p** *String*] [**-r**] [**-s**] [**+***LineNumber*] [**+**/*Pattern***/**] [*File ...*]

Description

The **pg** command reads a file name from the *File* parameter and writes the file to standard output one screen at a time. If you specify a **-** (dash) as the *File* parameter, or run the **pg** command without options, the **pg** command reads standard input. Each screen is followed by a prompt. If you press the Enter key, another page is displayed. Subcommands used with the **pg** command let you review or search in the file.

To determine workstation attributes, the **pg** command scans the file for the workstation type specified by the **TERM** environment variable. The default type is **dumb**.

When the **pg** command pauses and issues a prompt, you can issue a subcommand. Some of these subcommands change the display to a particular place in the file, some search for specific patterns in the text, and others change the environment in which the **pg** command works.

Changing Location Within the File

The following subcommands display a selected place in the file:

Item	Description
Page	Displays the page specified by the *Page* parameter.
+*Number*	Displays the page obtained by adding the *Number* value to the current page.
-*Number*	Displays the page as specified by the *Number* value before the current page.
l	(Lowercase L) Scrolls the display one line forward.
*Number***l**	Displays at the top of the screen the line specified by the *Number* parameter.
+*Number***l**	Scrolls the display forward for the specified number of lines.
-*Number***l**	Scrolls the display backward for the specified number of lines.
d	Scrolls half a screen forward. Pressing the Ctrl-D key sequence functions the same as the **d** subcommand.
-d	Scrolls half a screen backward. Pressing the **-**Ctrl-D key sequence functions the same as the **-d** subcommand.
Ctrl-L	Displays the current page again. A single . (dot) functions the same as the **Ctrl-L** key sequence subcommand.
$	Displays the last page in the file. Do not use this when the input is from a pipeline.

Searching for Text Patterns

The following subcommands search for text patterns in the text. (You can also use the patterns described in the **ed** command.) They must always end with a new-line character, even if the **-n** flag is used.

In an expression such as `[k.a-z]k.`, the minus implies a range, as in a through z, according to the current collating sequence. A collating sequence defines equivalence classes for use in character ranges.

Item	Description
[*Number***]/***Pattern***/**	Searches for the occurrence of the *Pattern* value as specified by the *Number* variable. The search begins immediately after the current page and continues to the end of the current file, without wraparound. The default for the *Number* variable is 1.
*Number***?***Pattern***?**	

Item	Description
*Number***^***Pattern***^**	Searches backward for the occurrence of the *Pattern* value as specified by the *Number* variable. The searching begins immediately before the current page and continues to the beginning of the current file, without wraparound. The default for the *Number* variable is 1. The ^ notation is useful for Adds 100 terminals which will not properly handle the ? notation.

After searching, the **pg** command displays the line with the matching pattern at the top of the screen. You can change the position of the display by adding the **m** or **b** suffix to the search command. The **m** suffix displays the line with the matching pattern in the middle of the screen for all succeeding subcommands. The **b** suffix displays the line with the matching pattern at the bottom of the screen for all succeeding subcommands. The **t** suffix displays the line with the matching pattern at the top of the screen again.

Changing the pg Environment

You can change the **pg** command environment with the following subcommands:

Item	Description
[*Number***]n**	Begins examining the next file in the command line, as specified by the *Number* variable. The default for the *Number* variable is first.
[*Number***]p**	Begins examining the previous file on the command line, as specified by the *Number* variable. The default for the *Number* variable is first.
[*Number***]w**	Displays another window of text. If the *Number* variable is specified, sets the window size to the number of lines it specifies. This subcommand is the same as the **[***Number***]z** subcommand.
[*Number***]z**	Displays another window of text. If the *Number* variable is specified, sets the window size to the number of lines it specifies. This subcommand is the same as the **[***Number***]w** subcommand.
s *File*	Saves the input in the specified file. Only the current file being examined is saved. This command must always end with a new-line character, even if you specify the **-n** flag.
h	Displays an abbreviated summary of available subcommands.
q or **Q**	Quits the **pg** command.
!*Command*	Sends the specified command to the shell named in the **SHELL** environment variable. If this is not available, the default shell is used. This command must always end with a new-line character, even if the **-n** flag is used.

Attention:

1. Some output is lost when you press the QUIT WITH DUMP (Ctrl-\) or INTERRUPT (Ctrl-C) key sequence because any characters waiting in the output queue are purged when the **QUIT** signal is received.
2. If workstation tabs are not set every eight positions, unpredictable results can occur.

At any time output is being sent to the workstation, you can press the QUIT WITH DUMP or INTERRUPT key sequence. This causes the **pg** command to stop sending output and displays the prompt. Then you can enter one of the preceding subcommands at the command prompt.

If standard output is not a workstation, the **pg** command acts like the **cat** command, except that a header is displayed before each file.

While waiting for workstation input, the **pg** command stops running when you press the INTERRUPT key sequence. Between prompts these signals interrupt the current task and place you in the prompt mode.

Flags

Item	Description
-c	Moves the cursor to the home position and clears the screen before each page. This flag is ignored if the `clear_screen` field is not defined for your workstation type in the **terminfo** file.
-e	Does not pause at the end of each file.
-f	Does not split lines. Normally, the **pg** command splits lines longer than the screen width.
-n	Stops processing when a **pg** command letter is entered. Normally, commands must end with a new-line character.
-p *String*	Uses the specified string as the prompt. If the *String* contains a **%d** value, that value is replaced by the current page number in the prompt. The default prompt is **:** (colon). If the specified string contains spaces, you must enclose the string in quotation marks.
-r	Prevents shell escape when the **"!"** subcommand is used.
-s	Highlights all messages and prompts.
+*LineNumber*	Starts at the specified line number.
-*Number*	Specifies the number of lines in the window. On workstations that contain 24 lines, the default is 23.
+/*Pattern***/**	Starts at the first line that contains the specified pattern.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Example

To look at the contents of a file one page at a time, enter:

```
pg filename
```

Files

Item	Description
/usr/bin/pg	Contains the **pg** command.
/usr/share/lib/terminfo/*	Contains the **terminfo** file that defines terminal types.

Item	Description
/tmp/pg*	Contains the temporary file created when using **pg** command.

phold Command

Purpose

Disables or reports the availability of login ports on hold.

Syntax

phold [**-a**] [*Device*]

Description

The **phold** command disables a set of login ports. The **phold** command allows logged-in users to continue, but does not allow any more users to log in. A user cannot log in on a disabled port. The system disables a port by updating an entry in the **/etc/inittab** file and then sending a signal to the **init** process. When the **init** process receives the signal and reads the updated status entry, it takes the appropriate action.

Use the *Device* parameter to specify the ports to be disabled. Permitted values include:

- A full device name, such as the **/dev/tty1** device
- A simple device name, such as the **tty1** device
- A number (*e.g.*, 1 to indicate the **/dev/tty1** device)

If you do not specify a *Device* parameter, the **phold** command reports the names of currently disabled ports in its set.

Note: You must have root user authority to run this command.

Flags

Item	Description
-a	Holds all ports that are currently enabled.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

To list the ports that are currently on hold, enter:

```
phold
```

Files

Item	Description
/etc/locks	Contains **lock** files for the **pshare** and **pdelay** commands.
/etc/phold	Contains the **phold** command.

pic Command

Purpose

Preprocesses **troff** command input for the purpose of drawing pictures.

Syntax

pic [**-T** *Name*] [**-** | *File* ...]

Description

The **pic** command is a **troff** command preprocessor for drawing simple figures on a typesetter. The basic objects are a box, circle, ellipse, line, spline, arrow, arc, and the text specified by the *Text* variable. The top-level object is the picture.

Item	Description
File	Specifies the output from a **troff** command that is processed by the **pic** command to draw pictures.

Pictures

The top-level object in the **pic** command is the picture.

.PS *OptionalWidth OptionalHeight*

ElementList

.PE

If the **.PF** macro is used instead of the **.PE** macro, the position after printing is restored to what it was upon entry.

Item	Description
OptionalWidth	Specifies the width of the picture (in inches), if present, regardless of any dimensions used internally. The maximum value is 8.5.
OptionalHeight	Specifies a height value, in inches, different from the default, which is scaled to the same proportion. The maximum value is 14.

Item	Description
ElementList	Represents the following list of elements: ***Shape AttributeList*** *For Statement* ***Placename: Element*** *If Statement* ***Placename: Position*** *Copy Statement* ***Variable = Expression*** Print Statement ***Direction*** Plot Statement **{ *List of Elements* }** sh *X Commandline X* **[*List of Elements*]** *troff-command*

Variable names begin with a lowercase letter, followed by zero or more letters or numbers. Place names begin with an uppercase letter, followed by zero or more letters or numbers. Place and variable names retain their values from one picture to the next.

Elements in a list must be separated by new-line characters or ; (semicolon); a long element can be continued by ending the line with a \ (backslash). Comments are introduced by a # character and ended by a new-line character.

Primitives

The primitive objects are as follows:

box

circle

ellipse

arc

line

arrow

spline

move

Text-List

The **arrow** object is the same as the **line** object with the **->** attribute.

Attributes

An *AttributeList* element is a sequence of zero or more attributes; each attribute consists of a keyword, perhaps followed by a value.

Attribute	Attribute
h(eigh)t *Expression*	**wid(th)** *Expression*
rad(ius) *Expression*	**diam(eter)** *Expression*
up *OptionalExpression*	**down** *OptionalExpression*
right *OptionalExpression*	**left** *OptionalExpression*
from *Position*	**to** *Position*

Attribute	**Attribute**
at *Position*	**with** *Corner*
by *Expression, Expression*	**then**
dotted *OptionalExpression*	**dashed** *OptionalExpression*
chop *OptionalExpression*	**-> <- <->**
invis	**same**
Text-list	

Missing attributes and values are filled in from defaults. Not all attributes make sense for all primitives; irrelevant ones are not processed. The following are the currently meaningful attributes:

Item	**Description**
Primitives	**Attributes**
box	**h(eigh)t**, **wid(th)**, **at**, **same**, **dotted**, **dashed**, **invis**, *Text*
circle, **ellipse**	**rad(ius)**, **diam(eter)**, **h(eigh)t**, **wid(th)**, **at**, **same**, **invis**, *Text*
arc	**up**, **down**, **left**, **right**, **h(eigh)t**, **wid(th)**, **from**, **to**, **at**, **rad(ius)**, **invis**, **ccw**, **cw**, **<-**, **->**, **<->**, *Text*
line, **arrow**	**up**, **down**, **left**, **right**, **h(eigh)t**, **wid(th)**, **from**, **to**, **by**, **then**, **at**, **same**, **dotted**, **dashed**, **invis**, **<-**, **->**, **<->**, *Text*
spline	**up**, **down**, **left**, **right**, **h(eigh)t**, **wid(th)**, **from**, **to**, **by**, **then**, **at**, **same**, **invis**, **<-**, **->**, **<->**, *Text*
move	**up**, **down**, **left**, **right**, **to**, **by**, **same**, *Text*
Text-list	**at**, *Text-item*

The **at** attribute implies placing the geometrical center at the specified place. For lines, splines, and arcs, the **h(eigh)t** and **wid(th)** attributes refer to arrowhead size.

The *Text-item* variable is usually an attribute of some primitive; by default, it is placed at the geometrical center of the object. Stand-alone text is also permitted. A *Text-list* primitive is a list of text items; a text item is a quoted string optionally followed by a positioning request, as follows:

"..."

"..." **center**

"..." **ljust**

"..." **rjust**

"..." **above**

"..." **below**

If there are multiple text items for some primitives, they are centered vertically except as qualified. Positioning requests apply to each item independently.

Text items can contain **troff** commands that control, for example, size and font changes and local motions. Make sure these commands are balanced so that the entering state is restored before exiting.

Item	Description
Positions/Places	A position is ultimately an *X,Y* coordinate pair, but it can also be specified in the following ways: *Place* (*Position*) *Expression, Expression* (*Position*) [**+/-** (*Expression, Expression*)] (*Position*) [**+/-** *Expression, Expression*] (*Place1, Place2*) (*Place1.X, Place2.Y*) *Expression* **<** *Position, Position* **>** *Expression* [**of the way**] **between** *Position* **and** *Position* *Placename* [*Corner*] *Corner Placename* **Here** *Corner* **of** *Nth Shape* *Nth shape* [*Corner*] **Note:** A *Corner* variable designates one of the eight compass points or the center, beginning, or end of a primitive, as follows: **.n .e .w .s .ne .se .nw .sw** **.t .b .r .l** **c .start .end**

Each object in a picture has an ordinal number; *Nth* refers to this, as follows:

- *Nth*
- *Nth* last

The **pic** command is flexible enough to accept names like **1***th* and **3***th*. Usage like **1***st* and **3***st* are accepted as well.

Variables

The built-in variables and their default values are as follows:

Item	Description
boxwid	0.75
boxht	0.5
circlerad	0.25
arcrad	0.25
ellipsewid	0.75
ellipseht	0.5
linewid	0.5
lineht	0.5
movewid	0.5

Item	Description
moveht	0.5
arrowwid	0.05
arrowht	0.1
textwid	0
textht	0
dashwid	0.5
scale	1

These default values can be changed at any time, and the new values remain in force from picture to picture until changed again.

The **textht** and **textwid** variables can be set to any value to control positioning. The width and height of the generated picture can be set independently from the **.PS** macro line. Variables changed within the [(left bracket) delimiter and the] (right bracket) delimiter revert to their previous value upon exit from the block. Dimensions are divided by **scale** during output.

Note: The **pic** command has an eight inch by eight inch limitation on picture sizes generated and sent to the **troff** command, even when the **.ps** (size) line specifies a size greater than eight inches.

Expressions

The following **pic** command expressions are evaluated in floating point. All numbers representing dimensions are taken to be in inches.

Expression + *Expression*

Expression - *Expression*

Expression * *Expression*

Expression / *Expression*

Expression % *Expression* (modulus)

- *Expression*

(*Expression*)

variable

number

Place **.x**

Place **.y**

Place **.ht**

Place **.wid**

Place **.rad**

sin(*Expression*) **cos**(*Expression*) **atan2**(*Expression, Expression*) **log**(*Expression*) **sqrt**(*Expression*) **int**(*Expression*) **max**(*Expression, Expression*) **min**(*Expression,Expression*) **rand**(*Expression*)

Logical Operators

The **pic** command provides the following operators for logical evaluation:

Item	Description
!	Not
>	Greater than

Item	Description
<	Less than
>/=	Greater than or equal to
</=	Less than or equal to
&&	And
|	Or
==	Equal to
!=	Not equal to

Definitions

The following **define** statement is not part of the grammar:

define *Name* **X** *Replacement text* **X**

Occurrences of values such as **$1** and **$2** in the *Replacement text* variable are replaced by the corresponding options if the *Name* variable is called, as follows:

Name(*Option1, Option2*, ...)

Non-existent options are replaced by null strings. The *Replacement text* variable can contain newline characters.

copy and copy thru Statements

The **copy** statement includes data from a file or values that immediately follow, such as:

```
copy File
```

```
copy thru Macro
```

```
copy File thru Macro
```

```
copy File thru Macro until String
```

The *Macro* parameter value can be either the name of a defined macro or the body of a macro enclosed in some character not part of the body. If no file name is given, the **copy** statement copies the input until the next **.PE** macro line.

for Loops and if Statements

The **for** and **if** statements provide for loops and decision-making, as follows:

```
Variable=Expression to Expression by Expression do X anything X
```

```
if Expression then X anything X else X anything X
```

The **by** and **else** clauses are optional. The *Expression* variable in an **if** statement can use the usual relational operators or the *String1* **==** (or **!=**) *String2* string tests.

Miscellaneous Information

The **sh** command runs a command line, as follows:

```
sh X Commandline X
```

It is possible to plot the value of an expression, as follows:

```
plot Expression OptionalFormat Attributes
```

The *Expression* variable value is evaluated and converted to a string (using the format specification, if provided).

The state of fill or no-fill mode is preserved with respect to pictures.

Input numbers can be expressed in **E** (exponential) notation.

Flags

Item	Description
-T*Name*	Prepares the output for the specified printing device. Possible values for *Name* are: **ibm3812** 3812 Pageprinter. **ibm3816** 3816 Pageprinter. **hplj** Hewlett-Packard LaserJet II. **ibm5587G** 5587-G01 Kanji Printer multi-byte language support. **psc** PostScript printer. **X100** AIXwindows display. **X100K** AIXwindows display for multi-byte character support. The default is **ibm3816**. **Note:** It is possible to set the **TYPESETTER** environment variable to one of the preceding values instead of using the **-T** *Name* flag of the **troff** command.
-	Reverts to standard input.

pick Command

Purpose

Selects messages by content and creates and modifies sequences.

Syntax

pick [**+***Folder*] [*Messages*] [**-datefield** *Field*] [**-not**] [**-lbrace**] [**-after** *Date*] [**-before** *Date*] [**-cc** *"Pattern"*] [**-date** *"Pattern"*] [**-from** *"Pattern"*] [**-search** *"Pattern"*] [**-to***"Pattern"*] [**—***Component "Pattern"*] [**-rbrace**] [**-and**] [**-or**] [**-sequence** *Name* [**-zero** | **-nozero**] [**-public** | **-nopublic**] [**-list** | **-nolist**]

Description

The **pick** command selects messages containing particular character patterns or particular dates. You can use the **-and**, **-or**, **-not**, **-lbrace**, and **-rbrace** flags to construct compound conditions for selecting messages.

Flags

Item	Description
-after *Date*	Selects messages with dates later than that specified by the *Date* variable. Use the following specifications for the *Date* variable: **yesterday** **today** **tomorrow** **sunday** **monday** **tuesday** **wednesday** **thursday** **friday** **saturday** **-**Days SystemDate The **pick** command treats the days of the week as days in the past. For example, **monday** means last Monday, not today or next Monday. You can use the **-***Days* argument to specify a number of days in the past. For example, -31 means 31 days ago. For the *SystemDate* argument, you can specify any valid format defined for your system.
-and	Forms a logical AND operation between two message-selecting flags; for example, `pick -after Sunday -and -from mark`. The **-and** flag has precedence over the **-or** flag, but the **-not** flag has precedence over the **-and** flag. Use the **-lbrace** and **-rbrace** flags to override this precedence.
-before *Date*	Selects messages with dates earlier than the specified date. See the **-after** flag on how to specify *Date*.
-cc "*Pattern***"**	Selects messages that contain the character string specified by the **"***Pattern***"** variable in the `cc:` field.
-date "*Pattern***"**	Selects messages that contain the character string specified by the **"***Pattern***"** variable in the `Date:` field.
-datefield *Field*	Specifies which dated field is parsed when the **-after** and **-before** flags are given. By default, the **pick** command uses the `Date:` field.
+*Folder*	Identifies the folder that contains the messages you wish to pick. By default, the system uses the current folder.
-from "*Pattern***"**	Selects messages that contain the character string specified by the **"***Pattern***"** variable in the `From:` field.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-lbrace	Groups **-and**, **-or**, and **-not** operations. Operations between the **-lbrace** and **-rbrace** flags are evaluated as one operation. You can nest the **-lbrace** and **-rbrace** flags.
-list	Sends a list of selected message numbers to standard output. This allows you to use the **pick** command to generate message numbers to use as input for other commands. For example, to scan all messages in the current folder that were sent after Tuesday, you would enter the following: `scan 'pick -after tuesday -list'` If you do not specify a sequence, the **-list** flag is the default.

Item	Description
Messages	Specifies the messages to search. You can specify several messages, a range of messages, or a single message. Use the following to specify messages: ***Number*** Number of the message. ***Sequence*** A group of messages specified by the user. Recognized values include: **all** All of the messages in the folder. This is the default. **cur or . (period)** Current message. **first** First message in a folder. **last** Last message in a folder. **new** New message that is created. **next** Message following the current message. **prev** Message preceding the current message.
-nolist	Prevents the **pick** command from generating a list of the selected message numbers. If a sequence is specified, the **-nolist** flag is the default.
-nopublic	Restricts a sequence to your usage. The **-nopublic** flag does not restrict the messages in a sequence, only the sequence itself. This option is the default if the folder is write-protected from other users.
-not	Forms a logical NOT operation on a message-selecting flag; for example, `pick -not -from george`. This construction evaluates all messages not chosen by the message-selecting flag. The **-not** flag has precedence over the **-and** flag, and the **-and** flag has precedence over the **-or** flag. Use the **-lbrace** and **-rbrace** flags to override this precedence.
-nozero	Appends the selected messages to the specified sequence.
-or	Forms a logical OR operation on two message-selecting flags; for example, `pick -from amy -or -from mark`. The **-not** flag has precedence over the **-and** flag, and the **-and** flag has precedence over the **-or** flag. Use the **-lbrace** and **-rbrace** flags to override this precedence.
-public	Allows other users access to a sequence. The **-public** flag does not make protected messages available, only the sequence itself. This option is the default if the folder is not write-protected from other users.
-rbrace	Groups **-and**, **-or**, and **-not** operations. Operations between the **-lbrace** and **-rbrace** flags are evaluated as one operation. You can nest the **-lbrace** and **-rbrace** flags.
-search "*Pattern***"**	Selects messages that contain the character string specified by the **"***Pattern***"** variable anywhere in the message.

Item	Description
-sequence *Name*	Stores the messages selected by the **pick** command in the sequence specified by the *Name* variable.
-to "*Pattern***"**	Selects messages that contain the character string specified by the **"***Pattern***"** variable in the `To:` field.
-zero	Clears the specified sequence before placing the selected messages into the sequence. This flag is the default.
—*Component* **"***Pattern***"**	Selects messages that contain the character string specified by the **"***Pattern***"** variable in the heading field specified by the *Component* variable; for example, `pick —reply-to amy`.

Profile Entries

The following profile entries are part of the *UserMHDirectory*/**.mh_profile** file:

Item	Description
`Current-Folder:`	Sets the default current folder.
`Path:`	Specifies the user's MH directory.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To get a list of message numbers in the current folder that are from user `jones`, enter:

   ```
   pick  -from jones
   ```

   The system responds with a message similar to the following:

   ```
   12
   15
   19
   ```

2. To see a list of message numbers in the `schedule` folder received within the last 30 days, enter:

   ```
   pick  + schedule  -after -30
   ```

   The system responds with a message similar to the following:

   ```
   5
   8
   21
   30
   ```

Files

Item	Description
$HOME/.mh_profile	Contains the user's MH profile.
/usr/bin/pick	Contains the **pick** command.

ping Command

Purpose

Sends an echo request to a network host.

Syntax

ping [**-d**] [**-D**] [**-n**] [**-q**] [**-r**] [**-v**] [**-R**] [**-a** *addr_family*] [**-c** *Count*] [**-w** *timeout*] [**-f** | **-i** *Wait*] [**-l** *Preload*] [**-p** *Pattern*] [**-s** *PacketSize*] [**-S** *hostname/IP addr*] [**-L**] [**- I** *a.b.c.d.*] [**-o** *interface*] [**-T** *ttl*] *Host* [*PacketSize*] [*Count*]

Description

The **/usr/sbin/ping** command sends an Internet Control Message Protocol (ICMP) ECHO_REQUEST to obtain an ICMP ECHO_RESPONSE from a host or gateway. The **ping** command is useful for:

- Determining the status of the network and various foreign hosts.
- Tracking and isolating hardware and software problems.
- Testing, measuring, and managing networks.

If the host is operational and on the network, it responds to the echo. Each echo request contains an Internet Protocol (IP) and ICMP header, followed by a ping PID and a **timeval** structure, and enough bytes to fill out the packet. The default is to continuously send echo requests until an Interrupt is received (Ctrl-C).

The **ping** command sends one datagram per second and prints one line of output for every response received. The **ping** command calculates round-trip times and packet loss statistics, and displays a brief summary on completion. The **ping** command completes when the program times out or on receipt of a **SIGINT** signal. The *Host* parameter is either a valid host name or Internet address.

By default, the **ping** command will continue to send echo requests to the display until an Interrupt is received (Ctrl-C). The Interrupt key can be changed by using the **stty** command.

Because of the load that continuous echo requests can place on the system, repeated requests should be used primarily for problem isolation.

Flags

Item	Description
-c *Count*	Specifies the number of echo requests, as indicated by the *Count* variable, to be sent (and received).
-w *timeout*	This option works only with the -c option. It causes ping to wait for a maximum of 'timeout' seconds for a reply (after sending the last packet).
-d	Starts socket-level debugging.
-D	This option causes a hex dump to standard output of ICMP ECHO_REPLY packets.
-f	Specifies flood-ping option. The **-f** flag "floods" or outputs packets as fast as they come back or one hundred times per second, whichever is more. For every ECHO_REQUEST sent, a . (period) is printed, while for every ECHO_REPLY received, a backspace is printed. This provides a rapid display of how many packets are being dropped. Only the root user may use this option. **Note:** This can be very hard on a network and should be used with caution. Flood pinging is only permitted by the root user. The **-f** flag is incompatible with the **-i** *Wait* flag.

Item	Description
-I *a.b.c.d*	Specifies that the interface specified by *a.b.c.d* is to be used for outgoing IPv4 multicasts. The **-I** flag is an uppercase i.
-o *interface*	Specifies that *interface* is to be used for outgoing IPv6 multicasts. The interface is specified in the form 'en0', 'tr0' etc.
-i *Wait*	Waits the number of seconds specified by the *Wait* variable between the sending of each packet. The default is to wait for one second between each packet. This option is incompatible with the **-f** flag.
-L	Disables local loopback for multicast pings.
-l *Preload*	Sends the number of packets specified by the *Preload* variable as fast as possible before falling into normal mode of behavior (one per second). The **-l** flag is a lowercase l.
-n	Specifies numeric output only. No attempt is made to look up symbolic names for host addresses.
-p *Pattern*	Specifies up to 16 'pad' bytes to fill out the packet you send. This is useful for diagnosing data-dependent problems in a network. For example, **-p ff** fills the packet with all 1's.
-q	Specifies quiet output. Nothing is displayed except the summary lines at startup time and when finished.
-r	Bypasses the routing tables and sends directly to a host on an attached network. If the *Host* is not on a directly connected network, the **ping** command generates an error message. This option can be used to ping a local host through an interface that no longer has a route through it.
-R	Specifies record route option. The **-R** flag includes the RECORD_ROUTE option in the ECHO_REQUEST packet and displays the route buffer on returned packets. **Note:** The IP header is only large enough for nine such routes. Also, many hosts and gateways ignore this option.
-a *addr_family*	Maps the destination address of the ICMP packets to IPv6 format if addr_family is equal to "inet6".
-s *PacketSize*	Specifies the number of data bytes to be sent. The default is 56, which translates into 64 ICMP data bytes when combined with the 8 bytes of ICMP header data.
-S *hostname/IP addr*	Uses the IP address as the source address in outgoing ping packets. On hosts with more than one IP address, the **-S** flag can be used to force the source address to be something other than the IP address of the interface on which the packet is sent. If the IP address is not one of the machine's interface addresses, an error is returned and nothing is sent.
-T *ttl*	Specifies that the time-to-live for a multicast packet is *ttl* seconds.
-v	Requests verbose output, which lists ICMP packets that are received in addition to echo responses.

Parameters

Item	Description
PacketSize	Specifies the number of data bytes to be sent. The default is 56, which translates into 64 ICMP data bytes when combined with the 8 bytes of ICMP header data. This parameter is included for compatibility with previous versions of the **ping** command.

Item	Description
Count	Specifies the number of echo requests to be sent (and received). This parameter is included for compatibility with previous versions of the **ping** command.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To check the network connection to host canopus and specify the number of echo requests to send, enter:

   ```
   ping  -c 5 canopus
   ```

   OR

   ```
   ping canopus  56  5
   ```

   Information similar to the following is displayed:

   ```
   PING canopus.austin.century.com: (128.116.1.5): 56 data bytes
   64 bytes from 128.116.1.5: icmp_seq=0 ttl=255 time=2 ms
   64 bytes from 128.116.1.5: icmp_seq=1 ttl=255 time=2 ms
   64 bytes from 128.116.1.5: icmp_seq=2 ttl=255 time=3 ms
   64 bytes from 128.116.1.5: icmp_seq=3 ttl=255 time=2 ms
   64 bytes from 128.116.1.5: icmp_seq=4 ttl=255 time=2 ms

   ----canopus.austin.century.com PING Statistics----
   5 packets transmitted, 5 packets received, 0% packet loss
   round-trip min/avg/max = 2/2/3 ms
   ```

2. To get information about host `lear` and start socket-level debugging, enter:

   ```
   ping  -d lear
   ```

   Information similar to the following is displayed:

   ```
   PING lear.austin.century.com: (128.114.4.18) 56 data bytes
   64 bytes from 128.114.4.18: icmp_seq=0 ttl=255 time=6 ms
   64 bytes from 128.114.4.18: icmp_seq=1 ttl=255 time=17 ms
   64 bytes from 128.114.4.18: icmp_seq=2 ttl=255 time=6 ms
   64 bytes from 128.114.4.18: icmp_seq=3 ttl=255 time=6 ms
   64 bytes from 128.114.4.18: icmp_seq=4 ttl=255 time=6 ms
   ^C
   ----lear.austin.century.com PING Statistics ----
   5 packets transmitted, 5 packets received, 0% packet loss
   round-trip min/avg/max = 6/8/17 ms
   ```

   Note: The output is repeated until an Interrupt (Ctrl-C) is received.

3. To obtain information about host opus and specify the number of data bytes to be sent, enter:

   ```
   ping  -s 2000 opus
   ```

   OR

   ```
   ping opus  2000
   ```

   Information similar to the following is displayed:

   ```
   PING opus.austin.century.com: (129.35.34.234): 2000 data bytes
   2008 bytes from 129.35.34.234: icmp_seq=0 ttl=255 time=20 ms
   2008 bytes from 129.35.34.234: icmp_seq=1 ttl=255 time=19 ms
   2008 bytes from 129.35.34.234: icmp_seq=2 ttl=255 time=20 ms
   ```

```
2008 bytes from 129.35.34.234: icmp_seq=3 ttl=255 time=20 ms
2008 bytes from 129.35.34.234: icmp_seq=4 ttl=255 time=20 ms
2008 bytes from 129.35.34.234: icmp_seq=5 ttl=255 time=19 ms
2008 bytes from 129.35.34.234: icmp_seq=6 ttl=255 time=19 ms
^C
----opus.austin.century.com PING Statistics----
7 packets transmitted, 7 packets received, 0% packet loss
round-trip min/avg/max = 19/19/20 ms
```

Note: The output is repeated until an Interrupt (Ctrl-C) is received.

4. To invoke the flood-ping option to host `stlopnor`, enter:

   ```
   ping  -f stlopnor
   ```

   Information similar to the following is displayed:

   ```
   Ping stlopnor.austin.century.com: (129.35.34.234): 56 data bytes
   .^C
   ----stlopnor.austin.century.com PING Statistics ----
   1098 packets transmitted, 1097 packets received, 0% packet loss
   round-trip min/avg/max = 4/4/11
   ```

   Note: The flood-ping output continues until an Interrupt (Ctrl-C) is received.

5. To specify an interval of five seconds between packets sent to host `opus`, enter:

   ```
   ping  -i5 opus
   ```

   Information similar to the following is displayed:

   ```
   PING opus.austin.century.com: (129.35.34.234): 56 data bytes
   64 bytes from 129.35.34.234: icmp_seq=0 ttl=255 time=5 ms
   64 bytes from 129.35.34.234: icmp_seq=1 ttl=255 time=5 ms
   64 bytes from 129.35.34.234: icmp_seq=2 ttl=255 time=6 ms
   ^C
   ----opus.austin.century.com PING Statistics----
   3 packets transmitted, 3 packets received, 0% packet loss
   round-trip min/avg/max = 5/5/6 ms
   ```

   Note: The output is repeated until an Interrupt (Ctrl-C) is received.

6. To send the number of packets specified by the *Preload* variable as fast as possible before falling into normal mode of behavior to host `opus`, enter:

   ```
   ping  -l 10 opus
   ```

   Information similar to the following is displayed:

   ```
   PING opus.austin.century.com: (129.35.34.234): 56 data bytes
   64 bytes from 129.35.34.234: icmp_seq=0 ttl=255 time=9 ms
   64 bytes from 129.35.34.234: icmp_seq=1 ttl=255 time=11 ms
   64 bytes from 129.35.34.234: icmp_seq=2 ttl=255 time=16 ms
   64 bytes from 129.35.34.234: icmp_seq=3 ttl=255 time=22 ms
   64 bytes from 129.35.34.234: icmp_seq=4 ttl=255 time=26 ms
   64 bytes from 129.35.34.234: icmp_seq=5 ttl=255 time=27 ms
   64 bytes from 129.35.34.234: icmp_seq=6 ttl=255 time=30 ms
   64 bytes from 129.35.34.234: icmp_seq=7 ttl=255 time=31 ms
   64 bytes from 129.35.34.234: icmp_seq=8 ttl=255 time=33 ms
   64 bytes from 129.35.34.234: icmp_seq=9 ttl=255 time=35 ms
   64 bytes from 129.35.34.234: icmp_seq=10 ttl=255 time=36 ms
   64 bytes from 129.35.34.234: icmp_seq=11 ttl=255 time=6 ms
   64 bytes from 129.35.34.234: icmp_seq=12 ttl=255 time=6 ms
   64 bytes from 129.35.34.234: icmp_seq=13 ttl=255 time=6 ms
   64 bytes from 129.35.34.234: icmp_seq=14 ttl=255 time=7 ms
   64 bytes from 129.35.34.234: icmp_seq=15 ttl=255 time=6 ms
   ^C
   ----opus.austin.century.com PING Statistics----
   16 packets transmitted, 16 packets received, 0% packet loss
   round-trip min/avg/max = 6/19/36 ms
   ```

   Note: The output is repeated until an Interrupt (Ctrl-C) is received.

7. To diagnose data-dependent problems in a network, enter:

```
ping  -p ff opus
```

This command sends packets with a pad-pattern of all 1's to host opus. Information similar to the following is displayed:

```
PATTERN: 0xff
PING opus.austin.century.com: (129.35.34.234): 56 data bytes
64 bytes from 129.35.34.234: icmp_seq=0 ttl=255 time=5 ms
64 bytes from 129.35.34.234: icmp_seq=1 ttl=255 time=5 ms
64 bytes from 129.35.34.234: icmp_seq=2 ttl=255 time=5 ms
64 bytes from 129.35.34.234: icmp_seq=3 ttl=255 time=6 ms
64 bytes from 129.35.34.234: icmp_seq=4 ttl=255 time=5 ms
^C
----opus.austin.century.com PING Statistics----
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 5/5/6 ms
```

Note: The output is repeated until an Interrupt (Ctrl-C) is received.

8. To specify quiet output, enter:

```
ping  -q bach
```

Only summary information similar to the following is displayed:

```
PING bach.austin.century.com: (129.35.34.234): 56 data bytes
^C
----bach.austin.century.com PING Statistics----
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 5/5/8 ms
```

Note: Although not displayed, the output of packets continues until an Interrupt (Ctrl-C) is received.

pioattred Command

Purpose

Provides a way to format and edit attributes in a virtual printer.

Syntax

pioattred **-q** *PrintQueueName* **-d** *QueueDeviceName* [**-o** *Action*] [**-a** *Attribute*]

Description

The **pioattred** command provides a way to format virtual printer attributes and to edit the attributes. Specifically, attributes in the printer definition file can be formatted and/or edited according to the action specified with the **-o** flag. Formatted attributes are written to standard output **stdout**. Attributes are edited with the editor specified in the **VISUAL** environment variable. The virtual printer definition file is assumed to be in the **/var/spool/lpd/pio/@local/custom/*** directory.

Flags

Item	Description
-a *Attribute*	Specifies the name of the attribute in the virtual printer definition file to format or edit. This flag may be specified many times.
-d *QueueDeviceName*	Specifies the *QueueDeviceName* spooler of the virtual printer definition to format or edit.

Item	Description
-o *Action*	Specifies the action that the **pioattred** command should take on the virtual printer definition. If this flag is omitted, the **pioattred** command assumes a value of 0 (zero). **0** Format the attributes specified. The result goes to **stdout**. **1** Format and edit the attribute(s) specified; use the editor specified in the **VISUAL** environment variable. If no editor is specified in the **VISUAL** environment variable, use the vi editor. If an error is made in editing the attributes, save the erroneous attributes in a temporary file, and return a return code indicating an error. The following values are used in the event that an error return code was returned after editing the attributes. **2** Edit the attributes again. The virtual printer definition will be the state it was left in when the error occurred. **3** Ignore the error and save the edited attributes in the virtual printer definition. **4** Clean up and leave things in the state they were before the **pioattred** command was started.
-q *PrintQueueName*	Specifies the *PrintQueueName* spooler of the virtual printer definition to format or edit.

Examples

1. To format the **ci** and **sh** attributes in the queue: quedev virtual printer definition, enter:

   ```
   pioattred -q queue -d quedev -o 0 -a ci -a sh
   ```

   OR

   ```
   pioattred -q queue -d quedev -a ci -a sh
   ```

2. To format all attributes in the queue: quedev virtual printer definition, enter:

   ```
   pioattred -q queue -d quedev -o 0
   ```

   OR

   ```
   pioattred -q queue -d quedev
   ```

3. To edit the **st** attribute in the queue: quedev virtual printer definition, enter:

   ```
   pioattred -q queue -d quedev -o 1 -a st
   ```

Files

Item	Description
/usr/sbin/pioattred	Contains the **pioattred** command.

piobe Command

Purpose

Print job manager for the printer backend.

Syntax

/usr/lpd/piobe [**-a** *PreviewLevel*] [**-A** *DiagnosticLevel*] [**-d** *InputDataStream*] [**-f** *FilterName*] [*FormatterFlags*] [*File* ...]

Description

The **piobe** command is a spooler backend program called by the **qdaemon** program to process a print job. The **piobe** command serves as the print job manager.

Based on the argument of the **-d** flag (or its default value in the database), which specifies the data stream type of the print files, the **piobe** command retrieves a pipeline from the database and passes it to a shell. The pipeline contains a string of filters to convert the print files as necessary and send them to a printer. If requested, the **piobe** command also retrieves and runs pipelines from the database to generate header and trailer pages.

The *FormatterFlags* argument (flags other than the flags listed in this topic) is assumed to be referenced by the filter commands in the pipelines. If a flag is specified but not referenced by the pipelines, an error message is issued and the print job ended.

> **Note:** The **piobe** command should not be typed directly on the command line. This command is invoked by the **qdaemon** process and is dependent on the various services provided by the **qdaemon** process.

Flags

Item	Description
-a *PreviewOption*	Provides a way to preview parameter values that would be used for a print job without actually printing any files. Values that can be specified for the *PreviewOption* variable are: **0** Specifies normal print processing **1** Returns a list of flag values and the pipeline of filters that would be used to convert the input data type to the data type expected by the printer, but does not actually invoke the pipeline of filters or send the file to the printer. The list of flag values returned are the default command line flag values from the configuration database. These values are overridden by any flag arguments specified on the command line. Please note that: • Only flags that are valid for the *InputDataTyp*e variable specified (or defaulted) for the **-d** flag are shown. • Flag values related only to the spooling of your print job, instead of the actual printing, are not shown. The default values for the spooling flags are included with the descriptions of the flags for the **qprt** command. • The flag values may not have been checked to verify that they are valid. The pipeline of filters shows the filter commands (and the flag values passed to the filter commands) that would process the data from your print file before it is sent to the printer. You can review the description for each of the filter commands to determine the type of filtering that would be performed.
-A *Value*	Specifies the level of diagnostic output. Diagnostic output is useful for diagnosing errors encountered by a pipeline of filters that is processing a print file, a header page, or a trailer page. Diagnostic output is mailed to the user who submitted the print job. The *Value* variable can be one of the following: **0** Discards any standard error output that is produced by the header, trailer, or print file pipelines. **1** If any standard error output is produced, returns the standard error output and the pipeline that produced it and ends the print job. **2** Returns the flag values, standard error output (if any), and completes pipelines, regardless of whether an error is detected. If an error is detected, the print job is ended. **3** Similar to a value of **2**, except that the file is not printed. A value of **1** is recommended. A value of **0** is used if a filter in a pipeline produces output to standard error, even if no error is encountered, such as for status information. A value of **2** or **3** is used for diagnosing a problem even if the problem does not cause any output to standard error.

Item	Description
-d *InputDataType*	Specifies the type of data that is in the file to be printed. This flag is a one-character identifier. Based on the data type for the print file and the data type expected by the printer, the print files are passed through filters (if necessary) before being sent to the printer. Examples of data type identifiers are: **a** IBM extended ASCII **p** Pass-through (sent to the printer unmodified) **s** PostScript **c** PCL **d** Diablo 630 **k** Kanji. If the printer you select does not support the *InputDataType* variable and filters are not available to convert the data type of your print file to a data type supported by the printer, the print job will be ended with an error message.
-f *FilterType*	Specifies a type of filter through which your print file is passed before being sent to the printer. This flag is a one-character identifier. The identifiers are similar to the filter flags available with the html

pioburst Command

Purpose

Generates burst pages (header and trailer pages) for printer output.

Syntax

/usr/lpd/pio/etc/pioburst [**-H** *HostName*] *TextFile*

Description

The **pioburst** command retrieves prototype text for a burst page from the file specified by the *TextFile* variable, fills in the variable fields identified by **%** escape sequences in the prototype text, and writes the constructed text to standard output. It is invoked as a filter in a pipeline by the print job manager, the **piobe** command.

The **%** escape sequences, which are replaced by corresponding values, are:

Item	Description
%A	Specifies the formatting flag values.
%D	Specifies the user to whom the print output is to be delivered.
%H	Specifies the name of the host machine printing the job.
%P	Specifies the time the print job was printed.
%Q	Specifies the time the print job was queued.

Item	Description
%S	Specifies the user who submitted the print job.
%T	Specifies the title of the print job.
%%	Specifies the % (percent sign).

Labels (20 characters long) for each of the variable fields can be specified by using the same escape sequence as for the variable field, except using lowercase letters. For example, to generate a label for the variable field specifying the print job was queued (**%Q**), use **%q**. The **%e** variable represents the label `END OF OUTPUT FOR:`.

The **pioburst** command requires the following environment variables to be initialized:

Item	Description
PIOTITLE	Title of the print job (for **%T**)
PIOQDATE	Time the print job was queued (for **%Q**)
PIOFROM	User who submitted the print job (for **%S**)
PIOTO	User to whom the print output is to be delivered (for **%D**)
PIOFLAGS	Flag values (for **%A**).

Flags

Item	Description
-H *HostName*	Specifies that the host name designated by the *HostName* variable override the default host name (the name of the host machine printing the job).

Example

To generate a header page and send it to standard output, enter:

```
pioburst /usr/lpd/pio/burst/H.ascii
```

Files

Item	Description
/usr/lpd/pio/etc/pioburst	Contains the **pioburst** command.

piocnvt Command

Purpose

Expands or contracts a predefined printer definition or a virtual printer definition.

Syntax

piocnvt [**-s** *State*] **-i** *SourceFile* [**-o** *TargetFile*]

Description

The **piocnvt** command takes either a predefined printer definition or a virtual printer definition and expands or contracts the file. An expanded printer definition file contains all the attributes associated with

that printer definition. A contracted printer definition contains only the printer specific attributes for that printer definition.

Printer definition files are arranged in a hierarchical parent-child relationship. For example the predefined printer definition `4201-3.asc` has the parent `master`. An expanded printer definition for `4201-3.asc` would contain all the attributes from `4201-3.asc` as well as those from `master`. A contracted printer definition for `4201-3.asc` would contain only the attributes not found in `master`. The **piocnvt** command simply provides a way to move back and forth between the expanded and contracted states of a printer definition file.

Flags

Item	Description
-i *SourceFile*	Specifies the complete path and name of the input file.
-o *TargetFile*	Specifies the complete path and name of the output file. If the **-o** flag is omitted, the *SourceFile* will be used for output.
-s *State*	Specifies whether the state of the *TargetFile* parameter should be expanded or contracted. If the **-s** flag is omitted, the **piocnvt** command attempts to determine the state by examining the **zD** attribute in the *SourceFile*. If a determination cannot be made the *TargetFile* parameter will be left in an expanded state. **+** Indicates that the state of the *TargetFile* parameter should be expanded. **!** Indicates that the state of the *TargetFile* parameter should be contracted.

Examples

1. To expand the virtual printer definition `lp0:lp0` into the file `new:lp0`; enter:

```
piocnvt -s+ -i lp0:lp0 -o new:lp0
```

2. To contract the virtual printer definition `lp0:lp0` in place; enter:

```
piocnvt -s! -i lp0:lp0
```

Files

Item	Description
/usr/sbin/piocnvt	Contains the **piocnvt** command.

piodigest Command

Purpose

Digests attribute values for a virtual printer definition into a memory image and stores the memory image in a file.

Syntax

/usr/lpd/pio/etc/piodigest [**-s** *DataStreamType*] [**-n** *DeviceName*] [**-p** *DirectoryPath*] [**-q** *PrintQueueName*] [**-t** *PrinterType*] [**-d** *QueueDeviceName*] { *ColonFileName* | **-** }

Description

The **piodigest** command retrieves virtual printer attribute values from a colon file, builds a memory image of the attribute values and their lookup tables, and writes the constructed memory image to a file. The memory image in the file is then available for access by a print formatter and by the **piobe** command (the print job manager). The command also creates Object Data Manager (ODM) stanzas for the specified queue and queue devices. The ODM stanzas are used in System Management Interface Tool (SMIT) dialogs. If an attribute called **zV** is specified and the attribute contains a value of **+**, the **piodigest** command performs syntax, reference, and limits validation on all attributes specified in the colon file.

The **piodidgest** command should be invoked whenever a customized version of a virtual printer definition is initially generated or is later modified. Each invocation of the **piodigest** command digests the attribute values for one virtual printer definition.

The *ColonFileName* parameter is the name of the input file in colon format. A colon file contains the attribute values for one virtual printer. A value of **-** (dash) for the *ColonFileName* parameter indicates that the colon file should be read from standard input.

The name of the output file that is generated will be of the form:

```
PrinterType.DataStreamType.DeviceName.PrintQueueName:QueueDeviceName
```

Flags

Item	Description
-d *QueueDeviceName*	Specifies the name of the virtual printer (queue device). If this flag is not specified, the virtual printer name specified by the **mv** attribute from the input colon file is assumed.
-n *DeviceName*	Specifies the name of the printer device, such as `lp0` for line printer 0, or `lp1` for line printer 1. If this flag is not specified, the device name specified by the **mn** attribute from the input colon file is assumed.
-p *DirectoryPath*	Specifies the path name of the directory where the output file is to be generated. If this flag is not specified, the **/var/spool/lpd/pio/@local/ddi** directory is assumed.
-q *PrintQueueName*	Specifies the name of the print queue to which the virtual printer is assigned. If this flag is not specified, the print queue name specified by the **mq** attribute from the input colon file is assumed.
-s *DataStreamType*	Specifies the printer data stream type. Example data stream types are **asc** (IBM extended ASCII), **ps** (PostScript), **pcl** (HP PCL), and **630** (Diablo 630). If this flag is not specified, the data stream type specified by the **md** attribute from the input colon file is assumed.
-t *PrinterType*	Specifies the printer type. Examples are `4201-3` and `ti2115`. If this flag is not specified, the printer type specified by the **mt** attribute from the input colon file is assumed.

Example

To generate a digested virtual printer definition, enter:

```
piodigest -d mypro -n lp0 -q proq -s asc -t 4201-3
```

The attribute values for the virtual printer assigned to the `mypro` queue device on the `proq` print queue are digested and stored in the file named **4201-3.asc.lp0.proq:mypro** in the **/var/spool/lpd/pio/@local/ddi** directory.

Files

Item	Description
/var/spool/lpd/pio/@local/ddi/*	Contains the digested, virtual printer definitions.
/usr/lpd/pio/etc/piodigest	Contains the **piodigest** command.

piodmgr Command

Purpose

Compacts the Object Data Manager (ODM) database in the **/var/spool/lpd/pio/@local/smit** directory.

Syntax

piodmgr { **-c** | **-h** }

Description

The **piodmgr** command extracts existing printer definitions from the ODM database in the **/var/spool/lpd/pio/@local/smit** directory, recreates the ODM database, compacts the database, and reloads the compacted database.

The **-c** and **-h** flags are mutually exclusive. The **-h** flag only compacts the database when the host name has been changed. The **-c** flag always compacts the database.

Note: Root user authority is needed to run this command.

Flags

Item	Description
-c	Extracts existing printer definitions from the ODM database, recreates the database, compacts the information, and replaces the database.
-h	Performs exactly like the **-c** flag, but the **-h** flag compacts the information only if the host name has been changed. If the host name has been changed, the **-h** flag extracts the new name and updates the host name information in the database. If the host name has not been changed, the **-h** flag does not compact the information. This flag is an optional compactor rather than an automatic compactor as with the **-c** flag.

Examples

1. To compact and update the ODM printer definition database, enter:

```
piodmgr -c
```

2. To perform compaction of the information depending on whether the host name has been changed or not, enter:

```
piodmgr -h
```

Files

Item	Description
/usr/lib/lpd/pio/etc/piodmgr	Contains the **piodmgr** command.
/var/spool/lpd/pio/@local/smit/*	Contains predefined printer definitions used by the command.

piofontin Command

Purpose

Copies fonts from a multilingual font diskette.

Syntax

piofontin **-t** *PrinterType* **-c** *Codepage* [**-d** *Device*]

Description

The **piofontin** command copies font files from a multilingual font diskette to a directory one level beneath the **/usr/lib/lpd/pio/fonts** label. The directory to which the font files are copied has the name specified by the *PrinterType* parameter. The font files are named according to the naming convention for files. Names are of the form:

```
codepage.typeface.pitch*10.quality
```

Only the root user can use the **piofontin** command.

Flags

Item	Description
-c *Codepage*	Specifies the code page for the fonts. For Greek fonts the value is 851, and for Turkish fonts the value is 853.
-d *Device*	Specifies the diskette-drive device name. This defaults to the **-d/dev/fd0** label, the standard 3.5-inch diskette drive.
-t *PrinterType*	Specifies the type of printer for the fonts. Supported printer types are 4201-3, 4202-3, 4207-2, 4208-2, 2380, 2381, 2390, and 2391.

Example

To read a diskette containing 4201-3 fonts in code page 851 from diskette drive `/dev/fd1;` enter:

```
piofontin 4201-3 851 /dev/fd1
```

The font files are copied to the **/usr/lib/lpd/pio/fonts/4201-3** directory.

File

Item	Description
/usr/sbin/piofontin	Contains the **piofontin** command.

pioformat Command

Purpose

Drives a printer formatter.

Syntax

/usr/lpd/pio/etc/pioformat -@ *DataBaseFile* [**-!** *FormatterName*] [**-# +** *PassThroughOption*]

Description

The **pioformat** command initiates the printer formatter driver. The formatter driver establishes access to the database values, loads and links a printer formatter, and then drives the formatter by calling its **setup** function, **initialize** function, **lineout** function, **passthru** function, and **restore** function as appropriate. The formatter driver also provides the **piogetopt** subroutine, **piogetstr** subroutine, **pioexit** subroutine used by the formatter.

The flags listed below are processed by the formatter driver and are not passed on to the formatter. However, all flags NOT listed below are assumed to be formatting flags and are passed on to the formatter.

Flags

Item	Description
-@ *DataBaseFile*	Specifies either of the following: • The full path name of the (digested) database file to be accessed • The print queue and queue device names, separated by a colon If the argument string begins with a **/** (slash) character, it is assumed to be a full path name. The combination of the queue name and the queue device name results in a unique string that is a part of the database file name and is used to search for the database file name in the **/var/spool/lpd/pio/@local/ddi** directory. This short form alternative is provided as a convenience when the formatter driver and formatter are run as standalone devices, instead of by the spooler.
-! *FormatterName*	Specifies the full path name of the formatter to be loaded, linked, and driven. If the **-!** flag is not specified, the default formatter name defined by the **mf** attribute name in the database is used. A default formatter name is provided as a convenience when the formatter driver and formatter are run as standalone devices, instead of by the spooler.

Item	Description
-# + *PassThroughOption*	Specifies that the print file should be passed through unmodified. If the **-# +** flag is not specified, the print file will be formatted. The parameter that is passed to the formatter's **setup** routine contains a value of 1 instead of 0, indicating that the file should be passed through instead of being formatted.

Examples

1. To format the `myfile` file according to the database file (virtual printer description) for the queue device named `std` associated with the print queue named `pro`, overriding the page width to 132 characters, and using the **pioformat** command and a formatter as a standalone filter, enter:

   ```
   cat myfile | pioformat  -@ pro:std -w 132 >/dev/lp0
   ```

2. To use the **pioformat** command and a formatter in a pipeline running under the spooler, enter:

   ```
   %Ide/pioformat  -@ %Idd/%Imm  -! %Idf/piof420x %Fbb %Fee ...
   ```

   For this example, assume that:

   - The printer is a 4207 Model 2 Proprinter.
   - The print queue name is `pro`.
   - There is only one queue device (virtual printer) defined for the print queue and its name is `std` and its output data stream type is `asc` (extended ASCII).
   - The printer device name is `/dev/lp0`.
   - The print job submitter specified the flag and argument `-i 5`.

   Before the print job manager (the **piobe** command) passes the pipeline to a shell to format the file, it resolves the pipeline's embedded references to attribute values. Based on the assumptions listed above for this example, the attribute references would be resolved as:

Item	Description
`%Ide -> /usr/lpd/pio/etc`	Directory where the **pioformat** command resides
`%Idd -> /var/spool/lpd/pio/@local/ddi`	Directory for database files
`%Imm -> 4207-2.asc.lp0.pro:std`	Database file name
`%Idf -> /usr/lpd/pio/fmtrs`	Directory for formatters
`%Fbb ->`	Null string, since submitter did not specify the **-b** flag
`%Fee -> -i 5`	Submitter specified this flag and argument.

   The resulting pipeline shown below would be passed to a shell to format the file (shown on multiple lines for readability):

```
/usr/lpd/pio/etc/pioformat        # initiate the formatter driver
-@/usr/lpd/pio/ddi/4207-2.asc.lp0.pro:std
                                  # (digested) database file
-!/usr/lpd/pio/fmtrs/piof420x     # loadable formatter
-i5                               # formatting option
                                  # (indent 5 characters)
```

Files

Item	Description
/usr/lpd/pio/etc/pioformat	Contains the formatter driver.
/usr/lpd/pio/fmtrs/*	Contains the formatters.
/var/spool/lpd/pio/@local/ddi/*	Contains the digested database files.

piofquote Command

Purpose

Converts certain control characters destined for PostScript printers.

Syntax

/usr/lpd/pio/etc/piofquote

Description

The **piofquote** command is a filter that converts certain control characters destined for PostScript printers that can emulate other printers. The command reads data from standard input, checks for control characters, and modifies them as needed. It then writes the data to standard output.

If a least 1 byte of data appears on standard input, the **piofquote** command writes a hex 04 control character to standard output before the first input data byte is written to standard output. The command also writes a hex 04 to standard output when end-of-file is recognized on standard input.

If a hex 01, 03, 04, 05, 11, 13, 14, or 1c control character is found in the input data read from standard input, the hex 40 bit in the control character is turned on and a hex 01 character is prefixed to the control character before it is written to standard output.

Files

Item	Description
standard input	Input data stream to be processed.
standard output	Output data stream containing converted control characters.

piolsvp Command

Purpose

Lists virtual printers on a system.

Syntax

piolsvp { **-q** | **-v** | **-Q** | **-p** | **-A** } [**-n***AttachmentField*]

piolsvp -P *Queue* [**:** *QueueDevice*] **-n***AttachmentField*

piolsvp -P *Queue* **-d**

piolsvp -N *AttachmentType* **-n***AttachmentField*

Description

The **piolsvp** command lists the virtual printers and attachment types on the system. The **piolsvp** command displays either the queues or the queues plus the queue-device pairs for virtual printers.

The order of the list of queues and queue-device pairs is the same as the order used by the **/etc/qconfig** file.

Flags

Item	Description
-A	Displays all attachment types and descriptions for the attachment types. The **.attach** and **.config** files in the **/usr/lib/lpd/pio/etc** directory define all attachment types.
-d	Displays the queue devices associated with a given queue.
-n*AttachmentField*	Specifies a field name for an attachment. The field name is typically a SMIT selector name. Possible values for the *AttachmentField* variable are: **submit_job** **add_queue** **add_printer** **remove_queue** **printer_conn** **change_queue** **change_filters** When the **-n** and **-A** flags are specified, only the attachment types that have a value for the specified attachment field in their attachment files are displayed. Attachment definitions are kept in the files with the *AttachmentType***.attach** naming convention. The **.attach** files reside in the **/usr/lib/lpd/pio/etc** directory. When the **-n** flag is specified with either the **-q** or **-v** flags, only queues and queue-device pairs that belong to defined attachment types are displayed. A defined attachment type has an assigned field value in the definition files. When the **-n** flag is specified with the **-P** flag, the SMIT selector name is displayed. The **-n** and **-P** flag combination also displays the queue device name and attachment type. When the **-n** flag is specified with the **-N** flag, the SMIT selector name is displayed for the specified attachment field and attachment type.
-N	Specifies an attachment type. The SMIT selector name associated with a given attachment field is displayed.
-p	Displays all the queue and queue-device pairs on the system and provides a description of each queue and queue-device pair. Only the queue name for the first queue-queue is displayed if there are queues with multiple queue devices.
-P	Specifies the queue name or queue device name for which information is displayed. The information consists of queue device name, attachment type, and SMIT selector value name.
-q	Displays all queues on the system. The **-q** flag also displays the queue-device pairs for queues that have more than one device.

Item	Description
-Q	Displays all the queues on the system. The **-Q** flag does not list queue-device pairs. Use the **-q** flag to list queue-device pairs.
-v	Displays all queue-device pairs for the queues that have virtual printers.

Examples

1. To display all the print queues on the system, enter:

```
piolsvp -q
```

   The output of this command is:

```
e4019a                  4019 (IBM ASCII)
d3816                   IBM 3816 Page Printer
ena_asc                 4029 (IBM ASCII)
ena_gl                  4029 (Plotter Emulation)
ena_pcl                 4029 (HP LaserJet II Emulation)
ena_ps                  4029 (PostScript)
hplj2                   Hewlett-Packard LaserJet II
tstx                    4216-31 (Proprinter XL Emulation)
e4019ps                 4019 (PostScript)
4019lxxa                4029 (PostScript)
4019lxxa:lxx            4029 (PostScript)
40191xxa:rkmlxx         4019 (IBM ASCII)
4019lxxa:rkmlxxl        4019 (IBM ASCII)
```

2. To display all the virtual printers in the system, enter:

```
piolsvp -v
```

   The output of this command is:

```
#QUEUE           DEVICE         DESCRIPTION
e4019a           e4019          4019 (IBM ASCII)
d3816            ena3816        IBM 3816 Page Printer
ena_asc          ena            4029 (IBM ASCII)
ena_gl           ena            4029 (Plotter Emulation)
ena_pcl          ena            4029 (HP LaserJet II Emulation)
ena_ps           ena            4029 (PostScript)
hplj2            lxx            Hewlett-Packard LaserJet II
tstx             lxx            4216-31 (Proprinter XL Emulation)
e4019ps          e4019          4019 (PostScript)
4019lxxa         lxx            4029 (PostScript)
4019lxxa         rkmlxx         4019 (IBM ASCII)
40191xxa         rkmlxx         4019 (IBM ASCII)
```

3. To list all the queues on the system, enter:

```
piolsvp -Q
```

   The output of this command is:

```
e4019a          4019 (IBM ASCII)
d3816           IBM 3816 Page Printer
ena_asc         4029 (IBM ASCII)
ena_gl          4029 (Plotter Emulation)
ena_pcl         4029 (HP LaserJet II Emulation)
ena_ps          4019 (PostScript)
hplj2           Hewlett-Packard LaserJet II
tstx            4216-31 (Proprinter XL Emulation)
e4019ps         4019 (PostScript)
4019lxxa        4029 (PostScript)
```

4. To list all the attachment types that have a SMIT selector value specified for the add_queue SMIT selector, enter:

```
piolsvp -A -nadd_queue
```

The output from this command is:

```
#ATTACHMENT TYPE        DESCRIPTION
local                   Local Attached
remote                  Remote Attached
ascii                   ASCII Terminal Attached
other                   Generic Backend Attached
```

5. To list information for the 4019lxxa queue, enter:

```
piolsvp -P4019lxxa -n add_queue
```

The output from this command is:

```
lxx     xsta    sm_xsta_addq_sel
```

6. To list the SMIT selector value for the remote attachment, enter:

```
piolsvp -Axst -nadd_queue
```

The output from this command is:

```
sm_xsta_addq_sel
```

Files

Item	Description
/usr/lib/lpd/pio/etc/piolsvp	Contains the **piolsvp** command.
/etc/qconfig	Contains the configuration files.
/var/spool/lpd/pio/@local/custom/*	Contains the customized virtual printer attribute files.
/usr/lib/lpd/pio/etc/*.attach	Contains the attachment type files

piomgpdev Command

Purpose

Manages printer pseudo-devices.

Syntax

piomgpdev **-p** *PseudoDevice* **-t** *AttachmentType* { **-A** | **-C** | **-R** | **-D** } [**-a** *Clause* ...]

Description

The **piomgpdev** command changes and removes pseudo-devices for printer attachments. The **piomgpdev** command stores information about the pseudo-devices in files in the **/var/spool/lpd/pio/@local/dev** directory. The file contains stanzas in the following form:

```
key_word = value
```

The information stored in these files pertains to connection characteristics for a given attachment and a printer.

Flags

Item	Description
-a *Clause*	Specifies a clause to be added or changed in the file for a pseudo-device. The clause is in the following form: `key_word = value` If the **-D** flag is specified, the clause can contain only the keyword.
-A	Adds a pseudo-device.
-C	Changes a pseudo-device.
-D	Displays information for a specified clause of a pseudo-device definition.
-p *PseudoDevice*	Specifies the name of a pseudo-device for a printer attachment.
-R	Removes a pseudo-device.

Files

Item	Description
/usr/lib/lpd/pio/etc/piomgpdev	Contains the **piomgpdev** command.
/var/spool/lpd/pio/@local/dev/*	Contains the printer pseudo-device files.

piomkapqd Command

Purpose

Builds a SMIT dialog to create print queues and printers.

Syntax

To Create a Print Queue for an Existing Printer

piomkapqd **-A** *AttachmentType* **-p** *Printer* **-d** *DeviceName* **-h** *Header* [**-e**]

To Create a Printer and a Print Queue

piomkapqd **-A** *AttachmentType* **-p** *Printer* **-v** *Device* **-s** *Subclass* **-r** *Adapter* **-h** *Header* [**-e**]

To Create a Printer Attached to a TTY or to Assign Printer Output to a File and Create a New Queue

piomkapqd **-A** *AttachmentType* **-p** *Printer* { **-T** *TTYName* | **-f** *FileName* } **-h** *Header* [**-e**]

To Use a User-Defined Attachment for a New Printer and Print Queue

piomkapqd **-A** *AttachmentType* **-p** *Printer* [**-d** *DeviceName*] **-c** *CmdExec* **-i** *DiscCmd* **-o** *ObjectID* **-h** *Header* [**-e**]

Description

The **piomkapqd** command creates a System Management Interface Tool (SMIT) dialog that allows the user to create new printers and print queues. The **piomkapqd** command also allows users to add their user-defined attachment types to a SMIT printer or queue definition dialog.

Flags

Item	Description
-A *AttachmentType*	Specifies the type of attachment used to connect the printer to the data source. Common values for the *AttachmentType* variable are: **local** Specifies a local attachment type. **ascii** Specifies an ASCII attachment type. **file** Specifies a file where the data is stored.
-c *CmdExec*	Specifies the value for the **cmd_to_execute** SMIT command. This flag is used when creating a user-defined attachment dialog. If this flag is not included, the **piomkpq** command is used as the default.
-d *DeviceName*	Specifies the name of the device, pseudo-device, or file where the output is directed, for example `lp0` or `tty1`.
-e	Specifies that an existing print queue is to be used for printer output. The **-e** prevents the **piomkapqd** command from creating a new queue.
-f *FileName*	Indicates the name of the file where output is stored.
-h *Header*	Specifies the title or header of the SMIT dialog that is being created.
-i *DiscCmd*	Specifies the value of the **cmd_to_discover** SMIT command. This flag is used when creating a user-defined attachment dialog. If this flag is not included, the **piomkapqd** command default value is used to create the dialog.
-o *ObjectID*	Specifies the SMIT object whose ID matches the value of the *ObjectID* variable.
-p *Printer*	Specifies the printer type as defined in the **/usr/lib/lpd/pio/predef** directory, for example `ibm4019`.
-r *ParentAdapter*	Specifies the parent adapter for the printer.
-s *Subclass*	Specifies the subclass type to which the printer belongs. The possible values for the *Subclass* variable are: • **parallel** • **rs232** • **rs422**
-T *TTYName*	Specifies the name of the TTY attached to the new printer or queue.
-v *Device*	Specifies the device type as defined in the ODM database. The **-v** flag retrieves printer definitions that are not stored in the **/usr/lib/lpd/pio/predef** directory.

Examples

1. To create a SMIT dialog that adds a print queue to an existing local printer, enter:

```
piomkapqd -A local -p ibm4019 -d lp0 -h 'Add a New Queue'
```

2. To create a SMIT dialog that adds a new printer named `lp2` and new print queue attached locally, enter:

```
piomkapqd -A local -p ibm4019 -v ibm4019 -s rs232 -r sa0 -h 'Add New Printer'
```

3. To create a SMIT dialog that adds a printer attached to a TTY and create a new queue for the printer, enter:

```
piomkapqd -A tty -p ibm4039 -T tty12 -h 'Add TTY Printer'
```

4. To create a SMIT dialog that directs output to a file name `stuff` and to create a new queue, enter:

```
piomkapqd -A file -p ibm4039 -f stuff -h 'Add Output File' -e
```

5. To create a SMIT dialog that adds a user-defined printer attachment type and creates a new queue, enter:

```
piomkapqd -A hpJetDirect -p hplj-4 [-d lp0] -c /usr/sbin/mkjetd -i /usr/bin/lsjd -o
JetDirect -h
'Add New Attachment Type'
```

File

Item	Description
/usr/lib/lpd/pio/etc/piomkapqd	Contains the **piomkapqd** command.

piomkpq Command

Purpose

Creates a print queue.

Syntax

To add a new printer

piomkpq **-A** *AttachmentType* **-p** *PrinterType* **-Q** *QueueName* **-D** *DataStream* **-v** *DeviceType* **-s** *Subclass* **-r** *ParentAdapter* **-w** *PortNumber* [**-a** { *interface* | *ptop* | *autoconfig* | *speed* | *parity* | *bpc* | *stops* | *xon* | *dtr* | *tbc*=*DescValue* }] ...

To create a new print queue

piomkpq **-A** *AttachmentType* **-p** *PrinterType* { **-D** *DataStream* | **-q** *QueueName* } **-s** *Subclass* **-r** *ParentAdapter* **-w** *PortNumber* **-v** *DeviceType* [**-a** { *interface* | *ptop* | *autoconfig* | *speed* | *parity* | *bpc* | *stops* | *xon* | *dtr* | *tbc*=*DescValue* }] ...

To create print queues for an existing printer

piomkpq **-A** *AttachmentType* **-p** *PrinterType* **-d** *DeviceName* { **-D** *DataStream* | **-q** *QueueName* }

To add an existing printer to an existing print queue

piomkpq **-A** *AttachmentType* **-p** *PrinterType* **-d** *DeviceName* **-D** *DataStream* **-q** *QueueName*

Description

The **piomkpq** command creates print queues and printers. This command is used by SMIT dialogs created with the **piomkapqd** command. The **piomkpq** command performs the following functions:

- Creates printer devices with various attachment types.
- Creates print queues.
- Creates queue devices.
- Creates virtual printers.

- Creates pseudo-devices.

Flags

Item	Description
-a	Specifies a device attribute. This takes the form *Attribute=Value*, for example: `-a speed=9600`. The valid attributes are: **Interface** **ptop** **autoconfic** **speed** **parity** **bpc** **stops** **xon** **dtr** **tbc**
-A *AttachmentType*	Specifies the type of attachment used to connect the printer to the data source. Common values for the *AttachmentType* variable are: **local** Specifies a local attachment type. **ascii** Specifies an ASCII attachment type. **file** Specifies a file where the data is stored.
-d *DeviceName*	Specifies the name of the device, pseudo-device, or file where the output is directed, for example `lp0` or `tty1`.
-D *DataStream*	Specifies the datastream of a print queue to be created or an existing print queue.
-p *PrinterType*	Specifies the printer type as defined in the **/usr/lib/lpd/pio/predef** directory, for example `ibm4019`.
-q *QueueName*	Specifies a new queue name. The **-q** and **-Q** flags are exclusive.
-Q *QueueName*	Specifies an existing queue name. The **-q** and **-Q** flags are exclusive.
-s *Subclass*	Specifies the subclass type to which the printer belongs. The possible values for the *Subclass* variable are: • **parallel** • **rs232** • **rs422**
-r *ParentAdapter*	Specifies the parent adapter for the printer.
-w *PortNumber*	Specifies the port number for the printer attachment.
-v *DeviceType*	Specifies the device type as defined in the ODM database.

Examples

1. To create a local print queue named `castor` of datastream ASCII for an existing IBM 4019 printer named `lp0`, enter:

```
piomkpq -A local -p ibm4019 -d lp0 -D asc -q castor
```

2. To add an existing local printer to an existing local print queue called `pyrite` for the datastream PostScript, enter:

```
piomkpq -A local -p ibm4019 -d lp0 -Q pyrite -D ps
```

3. To create local print queue called `baker` for a new printer, enter:

```
piomkpq -A local -p ibm4019 -D asc -Q baker -s parallel -r ppa0
-w p -v ibm4019 [-a ptop=120]
```

4. To create the **clues** file print queue, enter:

```
piomkpq -A file -p ibm4019 -d clues -D asc -q baker
```

Files

Item	Description
/usr/lib/lpd/pio/etc/piomkpq	Contains the **piomkpq** command.
/usr/lib/lpd/pio/etc/piomgpdev	Creates a pseudo-device.
/usr/sbin/mkdev	Creates a device.
/usr/bin/mkque	Creates a queue.
/usr/bin/mkquedv	Creates a queue device.
/usr/sbin/mkvirprt	Creates a virtual printer.

piomsg Command

Purpose

Sends a printer backend message to the user.

Syntax

piomsg [**-u** *UserList*] [**-c** *MsgCatalog* [**-s** *MsgSet*] **-n** *MsgNumber*] [**-a** *MsgArg*] ... [*MsgText*]

Description

The **piomsg** command either retrieves a printer backend message from a message catalog or sends a specified message text to one or more users. The **piomsg** command runs when a print job is executed. Typically, the **piomsg** command is used in printer colon files to send a message to the user submitting a print job while the print job is processed by the **piobe** command.

When the **-c**, **-s**, or **-n** flags are specified, the **piomsg** command retrieves a message from a message catalog. The command searches for the message in the directory specified in the **NLSPATH** environment variable. If the **NLSPATH** environment variable does not contain a directory path, the **piomsg** command searches the **/usr/lib/lpd/pio/etc** default directory. If no message is found in the **/usr/lib/lpd/pio/etc** directory, the command supplies the text specified in the *MessageText* variable. When the **-c**, **-s**, or **-n** flags are not specified, the **piomsg** command returns the value (if any) of the *MessageText* variable.

Each message is parsed for the **%s** or **%n$s printf** subroutine conversion specifications. The **printf** conversion specifications are replaced with supplied message strings, if any, before the message is sent to the user. The **piomsg** command processes escape sequences, such as, linefeed **/n** or horizontal tab **/t**, that are embedded in the message.

Flags

Item	Description
-a *MsgArg*	Specifies the message argument string. The value of the *MsgArg* variable is substituted into the message, if it contains the **%s** or **%n$s printf** subroutine conversion specifications. The **-a** flag can be specified up to 10 times to specify multiple arguments. If there are any errors while parsing conversion specifications, the original message is sent.
-c *MsgCatalog*	Specifies the message catalog that contains the message to be retrieved. The **-c** flag must be specified with the **-n** flag.
-n *MsgNumber*	Specifies the message number. The **-n** flag must be specified with the **-c** flag.
-s *MsgSet*	Specifies an optional message set. The default value for the *MsgSet* variable is 1. The **-s** flag must be specified with both the **-c** and **-n** flags.
-u *UserList*	Specifies the list of users who receive the message. The names of users or nodes in the *UserList* variable are separated by commas. To include a node name in the user list specify the @ character followed by a node name or address. If the **-u** flag is omitted, the message returns to the user who initiated the print job.

Examples

1. To retrieve message number `100` in message set number 1 from the `piobe.cat` message catalog and send the message to user `joe` on the same node as the print server and `tom` on node `foobar`, enter:

   ```
   piomsg -u joe,tom@foobar -c piobe.cat -n 100
   ```

2. To send a message with a message argument string to the user who submitted the print job, enter:

   ```
   piomsg -a "/usr/bin/troff" "The specified filter %s is not found\n"
   ```

3. To retrieve message number 5 in set number2 from the `xyz.cat`, use a dummy message in the event of a failure, and send the message to the printer, enter:

   ```
   piomsg -cxyz.cat -s2 -n5 "xyz.cat is not installed.\n"
   ```

   Note: When the **piomsg** command cannot retrieve messages from the catalog specified with the **NLSPATH** environment variable or the default directory, the supplied message text is sent to the users.

File

Item	Description
/usr/lib/lpd/pio/etc/piomsg	Contains the **piomsg** command.

pioout Command

Purpose

Printer backend's device driver interface program.

Syntax

/usr/lpd/pio/etc/pioout [**-A** *BytesPrinted*] [**-B** *TotalBytes*] [**-C** *NumberCancelStrings*] [**-D** *CancelString*] [**-E** *Mask*] [**-F** *FormFeedString*] [**-I** *InterventionRequiredUser*] [**-K** *TextString*] [**-L** *TextString*] [**-N** *NumberFormFeedStrings*] [**-O** *OutFile*] [**-P** *PrefixFile*] [**-R** *ParseRoutine*] [**-S** *SuffixFile*] [**-W+**]

Description

The **pioout** command is at the end of pipelines invoked by the **piobe** command (the print job manager) to print a file or a burst page on a printer. It reads input data from standard input, the prefix file (if the **-P** flag is specified), and the suffix file (if the **-S** flag is specified), and then writes the data to the printer (or *OutFile*, if the **-O** flag is specified). Error conditions and situations where intervention is required (unless the **-I** flag is specified) are reported to the user who submitted the print job.

The values specified with the **-A** flag and the **-B** flag are used to periodically report to the **qdaemon** process the percentage of the print job that has completed. The **-C** flag and the **-D** flag specify the data string sent to the printer if the print job is canceled.

The **-O** flag is used to generate a header page and store it in a temporary file. The **-P** flag is then used to print the header page (that was saved in a temporary file) just prior to printing the print file.

The **pioout** command requires the following environment variables to be initialized:

Item	Description
PIOTITLE	Title of the print job
PIODEVNAME	Device name
PIOQNAME	Print queue name
PIOQDNAME	Queue device name
PIOFROM	User who submitted the print job
PIOMAILONLY	If nonzero, message to user should always be mailed, not displayed.
PIOTERM	Overrides the terminal type assumed from the tty definition. This variable is only used for print jobs submitted to terminal-attached terminals.

Flags

Item	Description
-A *BytesPrinted*	Specifies the number of bytes already printed for the print job.
-B *TotalBytes*	Specifies the total number of bytes to be printed for the print job.
-C *NumberCancelStrings*	Specifies the number of times the string specified by the **-D** flag is to be sent to the printer when a print job is canceled. If this flag is not specified, the value is assumed to be 3168.
-D *CancelString*	Specifies the string to be sent to the printer when a print job is canceled. If the **-D** flag is not specified, the string is assumed to consist of 1 null character.
-E *Mask*	Specifies, as *Mask*, one or more device-driver error-flag names, separated by commas. If the mask is one returned by the **ioctl** subroutine with an **LPQUERY** command, the error condition indicated by the mask is ignored. Flag names can include **LPST_ERROR, LPST_NOSLCT**, and **LPST_SOFT**, and are defined in the **/usr/include/sys/lpio.h** file.

Item	Description
-F *FormFeed String*	Specifies the string to be sent to the printer to cause a form feed. If the **-F** flag is not specified, the string is assumed to be `\014`.
-I *InterventionRequiredUser*	Specifies the user to whom a message is to be sent when the printer requires intervention. If this flag is not specified, the message is sent to the user who submitted the print job. The *InterventionRequiredUser* parameter can be one or more user names, separated by commas. A null string represents the print job submitter. For example, the string `,jim@server02` causes intervention required messages to be sent to both the print job submitter and to user `jim` at node `server02`.
-K *TextString*	Specifies that messages sent by a PostScript printer will be discarded if they contain the specified text string. For example, if the *TextString* variable is `warming up`, messages that include the text `warming up` will be discarded.
-L *TextString*	Specifies that if a message received from a PostScript printer includes the specified text string, the text following this text string in the message will be sent to the intervention-required user specified by the **-I** flag.
-N *NumberFormFeedStrings*	Specifies the number of form-feed strings to be sent to the printer at the end of the input data stream. If this flag is not specified, the value is assumed to be zero. This flag is normally used only to align continuous forms after the printer has been idle, or to feed forms when the printer goes idle.
-O *OutFile*	Specifies that the output is sent to the specified file instead of being sent to the printer.
-P *PrefixFile*	Specifies the file sent to the printer before the first byte of the print file is sent. If the print job terminates before the first byte of the print file arrives, the prefix file is not sent.
-R *ParseRoutine*	Specifies the full path name of a routine to parse data read from the printer. An example of a parse routine is contained in the **/usr/include/piostruct.h** file. If the **-R** flag is not specified, a default parse routine is used.
-S *SuffixFile*	Specifies the file sent to the printer after the print file has been sent. If the print job terminates before the first byte of the print file arrives, the suffix file is not sent.
-W +	Specifies that EOF (hex 04) must be received from the printer in order to exit.

piopredef Command

Purpose

Creates a predefined printer data-stream definition.

Syntax

piopredef [**-r**] **-d** *QueueDeviceName* **-q** *PrintQueueName* **-s** *DataStreamType* **-t** *PrinterType*

Description

The **piopredef** command creates a predefined printer data-stream definition from a virtual printer definition. It can be thought of as the inverse of the **mkvirprt** command, displayed with the **chvirprt** command, and then specified with the **piopredef** command to create a predefined definition for the unsupported printer.

The new predefined printer definition can then be specified with a **mkvirprt** command to generate additional virtual printers for the unsupported printer type on the same computer, or transported to other computers and used there.

Flags

Item	Description
-d *QueueDeviceName*	Specifies with the *QueueDeviceName* variable the spooler of the customized virtual printer definition to be used to create the predefined printer definition.
-q *PrintQueueName*	Specifies with the *PrintQueueName* variable the spooler of the virtual printer definition to be used to create the predefined printer definition.
-r	Specifies that if the **-s** flag and the **-t** flag specify a predefined printer definition that already exists, the existing one should be replaced.
-s *DataStreamType*	Specifies with the *DataStreamType* variable the printer for the predefined printer definition to be created. Example data stream types are: **asc** IBM extended ASCII **gl** Hewlett-Packard GL **pcl** Hewlett-Packard PCL **ps** PostScript **630** Diablo 630 **855** Texas Instruments 855.
-t *PrinterType*	Specifies the printer type for the predefined printer definition to be created. Examples of existing printer types are: 4201-3, hplj-2, ti2115, and so on.

Note: If no flags are specified, the command syntax is displayed.

Example

To create a new predefined printer definition from an existing virtual printer definition for the virtual printer, enter:

```
piopredef -d mypro -q proq -s asc -t 9234-2
```

The attributes for the virtual printer assigned to the `mypro` queue device on the `proq` print queue are copied to create a new predefined printer definition for the `9234-2` printer (`asc` data stream).

Files

Item	Description
/etc/piopredef	Contains the **piopredef** command.
/usr/lpd/pio/predef/*	Predefined printer data stream attribute files. File names are in the format: `PrinterType.DataStreamType`.
/var/spool/lpd/pio/@local/custom/*	Customized virtual printer attribute files. File names are in the format: `PrintQueueName:QueueDeviceName`.

pkgadd Command

Purpose

Transfers a software package or set to the system.

Syntax

To Install a Software Package

pkgadd [**-d** *Device*] [**-r** *Response*] [**-n**] [**-a** *Admin*] [-P *Path*] [*Pkginst1* [*Pkginst2* [. . .]]]

To Copy a Software Package to the Specified Spool Directory

pkgadd -s *Spool* [**-d** *Device*] [*Pkginst1* [*Pkginst2* [. . .]]]

Description

pkgadd transfers the contents of a software package or set from the distribution medium or directory to install it onto the system. A package is a collection of related files and executables that can be independently installed. A set is made up of a special-purpose package, referred to as a Set Installation Package (SIP), and a collection of one or more packages that are members of the set. The SIP controls the installation of the set.

pkgadd checks that all packages listed on the command line are on the installation medium. If any of the packages listed does not exist, no changes are made to the system, that is, none of the listed packages are installed.

Note: Nonroot users must meet the following conditions to run the pkgadd command successfully:

1. Users must have write permission to the paths specified in the pkgmap file.
2. The current *user:group* must match the *user:group* specified in the pkgmap file.
3. Users must have write permissions on the `/var/sadm/install` and `/var/sadm/pkg` directories.

Used without the **-d** flag, **pkgadd** looks in the default spool directory for the package (**/var/spool/pkg**). Used with the **-s** flag, it writes the package to a spool directory instead of installing it.

Error messages are always logged. In addition, when **pkgadd** terminates, it sends mail (by default, to "root") with all the error messages and a summary of which packages installed completely, partially, or not at all.

Flags

Item	Description
-d *Device*	Installs or copies a package/set from *Device*. *Device* can be the full pathname to a directory, file or named pipe, or "-" which specifies packages in datastream format read from standard input. The default device is the installation spool directory (**/var/spool/pkg**).
-r *Response*	Identifies a file or directory, *Response*, which contains the answers to questions posed by a "request script" during a previous **pkgask** session conducted in interactive mode [see the **pkgask** command]. When *Pkginst* is a package, *Response* can be a full pathname or a directory; when *Pkginst* is a SIP, *Response* must be a directory.
-n	Specifies that installation runs in non-interactive mode. The default mode is interactive.
-a *Admin*	Defines an installation administration file, *Admin*, to be used in place of the default administration file to specify whether installation checks (such as the check on the amount of space, the system state, and so on) are done. The token "none" overrides the use of any **admin** file, and thus forces interaction with the user. Unless a full pathname is given, **pkgadd** looks in the **/var/sadm/install/admin** directory for the file. By default, the file **default** in that directory is used. **default** specifies that no checking is done, except to see if there is enough room to install the package and if there are dependencies on other packages. The **-a** flag cannot be used if *Pkginst* is a SIP.
-P *Path*	Specifies an alternative root directory path for installation. Files will be installed under this location.
Pkginst	Defines a short string used to designate an abbreviation for the package/set name. (The term "package instance" is used loosely: it refers to all instantiations of *Pkginst*.) See the **pkginfo** command and the **pkginfo** file format. If *Pkginst* is a SIP, the SIP controls installation of the set by using request scripts and pre-install scripts. The SIP request script, not the package installation tools, is responsible for prompting the user for responses and taking the appropriate actions. If the request script fails, only the SIP is processed. To indicate all instances of a package, specify '*Pkginst***.***', enclosing the command line in single quotes, as shown, to prevent the shell from interpreting the "*" character. Use the token "all" to refer to all packages available on the source medium.
-s *Spool*	Reads the package into the directory *Spool* instead of installing it.

Special Notes

The **-r** flag can be used to indicate a directory name as well as a filename. The directory can contain numerous *Response* files, each sharing the name of the package with which it should be associated. This would be used, for example, when adding multiple interactive packages with one invocation of **pkgadd**. Each package that had a request script would need a *Response* file. If you create response files with the same name as the package (for example, *Package1* and *Package2*) then, after the **-r** flag, name the directory in which these files reside.

The **-n** flag causes the installation to halt if any interaction is needed to complete it.

When invoked with no *Pkginst* specified on the command line, **pkgadd** only displays the names of sets if at least one SIP exists on the media. Because of this, you shouldn't include packages on the same media if some are members of sets and some are not. If you do, the packages which are not members of sets can be installed only if their **pkginst** names are provided on the command line.

The **pkgadd** command checks to see if any of the files in *Pkginst* are already installed on the system and, if any are, saves this fact before continuing with installation. Later, **pkgadd** does not reinstall these files

on the system. If one of the packages installation scripts removes such a file, the result is that the file will no longer be on the system when package installation completes.

The **pkgadd** command does not uncompress any files that were already compressed (that is, only those in "**.Z**" form) before being processed by **pkgmk**.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion of script.
1	Fatal error. Installation process is terminated at this point.
2	Warning or possible error condition. Installation continues. A warning message is displayed at the time of completion.
3	Script was interrupted and possibly left unfinished. Installation terminates at this point.
4	Script was suspended (administration). Installation terminates at this point.
5	Script was suspended (interaction was required). Installation terminates at this point.
10	System should be rebooted when installation of all selected packages is completed. (This value should be added to one of the single-digit exit codes described above.)
20	The system should be rebooted immediately upon completing installation of the current package. (This value should be added to one of the single-digit exit codes described above.)
77	No package was selected for the set.
99	Internal error.

Files

Item	Description
/var/sadm/install/admin/default	default package administration file
/var/sadm/install/logs/*pkginst*.log	error message log
/var/spool/pkg	default spool directory

pkgask Command

Purpose

Stores answers to a request script.

Syntax

pkgask [**-d** *Device*] **-r** *Response* [*Pkginst* [*Pkginst* [. . .]]

Description

pkgask enables an administrator to store answers to an interactive package (one with a request script) or a set of packages. A set is made up of a special-purpose package, referred to as a Set Installation Package (SIP), and a collection of one or more packages that are members of the set. The SIP controls the installation of the set.

Invoking **pkgask** generates a *Response* file that is then used as input at installation time. The use of this *Response* file prevents any interaction from occurring during installation since the file already contains all of the information the package needs.

When **pkgask** runs, it creates the response file as well as the following directories:

Item	Description
/ptfvars	Contains variables pertaining to the package.
/fileinfo	Contains checksum information about the package.
/oldfiles	Contains backups of previous versions of the package.

To install the package on another system non-interactively, you must copy all of these files and directories to the target system.

Note: If you overwrite any of these directories, for example, to install another package non-interactively, you will not be able to successfully remove the first package unless you restore the original directory contents first.

You can use the **-r** flag to indicate a directory name as well as a filename. The directory name is used to create numerous *Response* files, each sharing the name of the package with which it should be associated. This is useful, for example, when you add multiple interactive packages with one invocation of **pkgadd**. Each package needs a *Response* file. To create multiple response files with the same name as the package instance, name the directory in which the files should be created and supply multiple instance names with the **pkgask** command. When installing the packages, you can identify this directory to the **pkgadd** command.

Flags

Item	Description
-d *Device*	Runs the request script for a package on *Device*. *Device* can be the full pathname to a directory (such as **/var/tmp**), or "-" which specifies packages in datastream format read from standard input. The default device is the installation spool directory (**/var/spool/pkg**).
-r *Response*	Identifies a file or directory, *Response*, which should be created to contain the responses to interactions with the packages request script. The file, or directory of files, can later be used as input to the **pkgadd** command [see the **pkgadd** command]. When *Pkginst* is a package, *Response* can be a full pathname or a directory; when *Pkginst* is a SIP, *Response* must be a directory.

Item	Description
Pkginst	Defines a short string used to designate an abbreviated package/set name. (The term "package instance" is used loosely: it refers to all instantiations of *Pkginst*, even those that do not include instance identifiers.) To create a package name abbreviation, assign it with the "PKG" parameter. For example, to assign the abbreviation "cmds" to the Advanced Commands package, enter **PKG=cmds**. If *Pkginst* specifies a SIP, all request scripts for packages which are members of that set are run (if any) and the resulting response files are placed in the directory provided to the **-r** flag. To indicate all instances of a package, specify '*Pkginst*.*', enclosing the command line in single quotes, as shown, to prevent the shell from interpreting the "*" character. Use the token "all" to refer to all packages available on the source medium. **Note:** When invoked with no *Pkginst* specified on the command line, **pkgask** only displays the names of sets if at least one SIP exists on the device. Thus, if you have packages which are not members of sets, they can be referenced only if their *Pkginst* names are provided on the command line.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion of script.
1	Fatal error. Installation process is terminated at this point.
2	Warning or possible error condition. Installation continues. A warning message is displayed at the time of completion.
3	Script was interrupted and possibly left unfinished. Installation terminates at this point.
4	Script was suspended (administration). Installation terminates at this point.
5	Script was suspended (interaction was required). Installation terminates at this point.
10	System should be rebooted when installation of all selected packages is completed. (This value should be added to one of the single-digit exit codes described above.)
20	The system should be rebooted immediately upon completing installation of the current package. (This value should be added to one of the single-digit exit codes described above.)
77	No package was selected for the set.
99	Internal error.

Files

Item	Description
/var/spool/pkg	default spool directory

pkgchk Command

Purpose

Checks the accuracy of an installation.

Syntax

To Check the Contents of Installed Objects

pkgchk [**-l** | **-a** **-c** **-f** **-q** **-v**] [**-n** **-x**] [-P *path*] [**-p** *Path1*[,*Path2* . . .] [**-i** *File*] [*Pkginst* . . .]

To Check the Contents of a Package Spooled on a Specified Device

pkgchk **-d** *Device* [**-l** | **-v**] [**-p** *Path1*[,*Path2* . . .] [**-i** *File*] [*Pkginst* . . .]

To Check the Contents of a Package Described in the Specified pkgmap

pkgchk **-m** *Pkgmap* [**-e** *Envfile*] [**-l** | **-a** **-c** **-f** **-q** **-v**] [**-n** **-x**] [**-i** *File*] [**-p** *Path1*[,*Path2* . . .]]

Description

pkgchk checks the accuracy of installed files or, by use of the **-l** flag, displays information about package files. The command checks the integrity of directory structures and the files. Discrepancies are reported on **stderr** along with a detailed explanation of the problem.

The first synopsis defined above is used to list or check the contents and/or attributes of objects that are currently installed on the system. Package names can be listed on the command line, or by default the entire contents of a machine is checked. If packages are installed in an alternative root directory path using the pkgadd command with the -P option, contents and attributes can be checked or listed using the same alternative root directory path specified with the -P option.

The second synopsis is used to list or check the contents of a package which has been spooled on the specified device, but not installed. Note that attributes cannot be checked for spooled packages.

The third synopsis is used to list or check the contents and/or attributes of objects which are described in the indicated *Pkgmap*.

Flags

Item	Description
-l	Lists information on the selected files that make up a package. It is not compatible with the **a**, **c**, **f**, **g**, and **v** flags.
-a	Audits the file attributes only, does not check file contents. Default is to check both.
-c	Audits the file contents only, does not check file attributes. Default is to check both.
-f	Corrects file attributes if possible. If used with the **-x** flag, it removes hidden files. When **pkgchk** is invoked with this flag it creates directories, named pipes, links, and special devices if they do not already exist.
-q	Enables quiet mode. Does not give messages about missing files.
-v	Enables verbose mode. Files are listed as processed.
-n	Ignores volatile or editable files. This should be used for most post-installation checking.

Item	Description
-x	Searches exclusive directories only, looking for files that exist that are not in the installation software database or the indicated *Pkgmap* file. (An exclusive directory is a directory created by and for a package; it should contain only files delivered with a package. If any non-package files are found in an exclusive directory, **pkgchk** reports an error.) If **-x** is used with the **-f** flag, hidden files are removed; no other checking is done. **Note:** To remove hidden files only, use the **-f** and **-x** flags together. To remove hidden files and check attributes and contents of files, use the **-f**, **-x**, **-c**, and **-a** flags together.
-p	Only checks the accuracy of the pathname or pathnames listed. "pathname" can be one or more pathnames separated by commas (or by white space, if the list is quoted).
-i	Reads a list of pathnames from *File* and compares this list against the installation software database or the indicated *Pkgmap* file. Pathnames that are not contained in "inputfile" are not checked.
-d	Specifies the device on which a spooled package resides. *Device* can be a directory pathname, or "-" which specifies packages in datastream format read from standard input.
-m	Requests that the package be checked against the pkgmap file *Pkgmap*.
-e	Requests that the pkginfo file named as *Envfile* be used to resolve parameters noted in the specified pkgmap file.
Pkginst	Defines a short string used to designate an abbreviation for the package name. (The term "package instance" is used loosely: it refers to all instantiations of *Pkginst*, even those that do not include instance identifiers.) To indicate all instances of a package, specify '*Pkginst*.*****', enclosing the command line in single quotes, as shown, to prevent the shell from interpreting the "*" character. Use the token "all" to refer to all packages available on the source medium.
-P *path*	Requests that the package in the alternate root directory path be checked.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion of script.
1	Fatal error. Installation process is terminated at this point.

Files

Item	Description
/usr/sbin/pkgchk	Contains the **pkgchk** command.

pkginfo Command

Purpose

Displays software package and/or set information.

Syntax

To Display Information on Installed Packages

pkginfo [**-q**] [**-x** | **-l**] [**-r**] [**-p** | **-i**] [**-a** *Arch*] [-P *Path*] [**-v** *Version*] [**-c** *Category1*,[*Category2*[, . . .]]] [*Pkginst* [, *Pkginst* [, . . .]]]

To Display Information on Packages Contained in the Specified Device

pkginfo [**-d** *Device*] [**-q**] [**-x** | **-l**] [**-a** *Arch*] [-P *Path*] [**-v** *Version*] [**-c** *Category1* [,*Category2*[, . . .]]] [*PkginstPkginst* [, *Pkginst* [, . . .]]]

Description

pkginfo displays information about software packages or sets that are installed on the system (as requested in the first synopsis) or that reside on a directory (as requested in the second synopsis). A package is a collection of related files and executable that can be independently installed. A set is made up of a special-purpose package, referred to as a Set Installation Package (SIP), and a collection of one or more packages that are members of the set. The SIP controls the installation of the set.

When run without flags, **pkginfo** displays one line of information about every installed package (whether installed completely or partially) whose category is not the value "set". The information displayed includes the primary category, package instance, and name of the package. For UNIX software packages produced before UNIX System V Release 4, **pkginfo** displays only the package name and abbreviation.

The **-p** and **-i** flags are meaningless if used in conjunction with the **-d** flag. The **-p** and **-i** flags are mutually exclusive. The **-x** and **-l** flags are mutually exclusive.

Flags

Item	Description
-q	Enables quite mode - no information is displayed. This flag overrides the **-x**, **-l**, **-p**, and **-i** flags. (Can be invoked by a program to query whether or not a package has been installed.)
-x	Extracts and displays the following information about the specified package: abbreviation, name, and, if available, architecture and version.
-l	Displays a "long format" report (that is, one that includes all available information) about the specified package(s).
-r	Lists the installation base for the specified package if the package is relocatable.
-p	Displays information only for partially installed packages.
-i	Displays information only for fully installed packages.
-a *Arch*	Specifies the architecture of the package as *Arch*.
-P *Path*	Displays information for packages installed in the alternative root directory path.
-v *Version*	Specifies the version of the package as *Version*. All compatible versions can be requested by preceding the version name with a tilde "~".

Item	Description
-c *Category . . .*	Displays information about packages that belong to category *Category*. (Categories are defined in the category field of the **pkginfo** file; see the **pkginfo** file format for details.) More than one category may be specified in a comma-separated list. A package is required to belong to only one category, even when multiple categories are specified. The package-to-category match is not case-sensitive. If the category specified is "set", **pkginfo** displays information about Set Installation Packages (SIPs).
Pkginst	Defines a short string used to designate an abbreviation for the package/set name. (The term "package instance" is used loosely: it refers to all instantiations of *Pkginst*, even those that do not include instance identifiers.) To indicate all instances of a package, specify '*Pkginst***.***', enclosing the command line in single quotes, as shown, to prevent the shell from interpreting the "*" character. Use the token "all" to refer to all packages available on the source medium. If *Pkginst* is a SIP, information about the packages with which the SIP is associated is displayed.
-d *Device*	Displays information from packages/sets that reside on *Device*. *Device* can be the full pathname to a directory (such as **/var/tmp**), or "**-**" which specifies packages in datastream format read from standard input. The default device is the installation spool directory (**/var/spool/pkg**).

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion of script.
1	Fatal error. Installation process is terminated at this point.

Files

Item	Description
/var/spool/pkg	Default spool directory

pkgmk Command

Purpose

Produces an installable package.

Syntax

pkgmk [**-c**] [**-o**] [**-a** *Arch*] [**-b** *BaseDir*] [**-d** *Directory*] [**-f** *Prototype*] [**-l** *Limit*] [**-p** *PStamp*] [**-r** *RootPath*] [**-v** *Version*] [*Variable*=*Value* ...] [*PkgInst*]

Description

pkgmk produces an installable package to be used as input to the **pkgadd** command. A package is a collection of related files and executables that can be independently installed. The package contents will be in directory structure format.

The **pkgmk** command uses the package prototype file as input and creates a **pkgmap** file. The contents for each entry in the prototype file is copied to the appropriate output location. Information concerning the contents (checksum, file size, modification date) is computed and stored in the **pkgmap** file, along with attribute information specified in the prototype file.

Flags

Item	Description
-a *Arch*	Overrides the architecture information provided in the **pkginfo** file with *Arch*.
-b *BaseDir*	Prepends the indicated *BaseDir* to locate relocatable objects on the source machine.
-c	Compresses non-information files. You must also specify the **-r** option when using **-c**. Entries in the *Prototype* file that reference relative paths above the *RootPath* specification will not be compressed. Any files that were already compressed (that is, only those in ".Z" form) before being processed by **pkgmk** will not be uncompressed by the **pkgadd** command.
-d *Directory*	Creates the package in *Directory*. The directory named must already exist.
-f *Prototype*	Uses the file *Prototype* as input to the command. The default name for this file is either **Prototype** or **prototype**. You can use **pkgproto** to create the *Prototype* file. In this case, you must manually add in the entries for any installation scripts and files you are using in the package. You only need entries for those files and scripts that you use. However, you must always add an entry for the **pkginfo** file for the package. See **pkgproto** for more information.
-l *Limit*	Specifies the maximum size in 512-byte blocks of the output device as *Limit*. By default, if the output file is a directory or a mountable device, **pkgmk** will employ the **df** command to dynamically calculate the amount of available space on the output device. Useful in conjunction with **pkgtrans** to create a package with datastream format.
-o	Overwrites the same instance. The package instance will be overwritten if it already exists.
-p *PStamp*	Overrides the production stamp definition in the **pkginfo** file with *PStamp*.
-r *RootPath*	Appends the source pathname in the *Prototype* file to the indicated *RootPath* to locate objects on the source machine.

Item	Description
-v *Version*	Overrides version information provided in the **pkginfo** file with *Version*.
Variable=*Value*	Places the indicated variable in the packaging environment.
PkgInst	A short string used to designate an abbreviation for the package name. **pkgmk** will automatically create a new instance if the version and/or architecture is different. A user should specify only a package abbreviation; a particular instance should not be specified unless the user is overwriting it.

Examples

1. If you want to create a package named mypkgA containing the **lsps** and **lsuser** commands, you must first create the contents of the package. For example:

```
mkdir -p /home/myuser/example/pkgmk/sbin
cp /usr/sbin/lsps /home/myuser/example/pkgmk/sbin
cp /usr/sbin/lsuser /home/myuser/example/pkgmk/sbin
```

   Then, create the **pkginfo** file. In this example the **pkginfo** file is /home/myuser/example/pkgmk/pkginfo, which contains the following:

```
PKG="mypkgA"
NAME="My Package A"
ARCH="PPC"
RELEASE="1.0"
VERSION="2"
CATEGORY="Application"
PSTAMP="AIX  2001/02/05"
```

   Then, create the *Prototype* file, /home/myuser/example/pkgmk/prototype file which contains the following:

```
!search /home/myuser/example/pkgmk/sbin
i pkginfo=/home/myuser/example/pkgmk/pkginfo
d example /example 1777 bin bin
d example /example/pkgmk 1777 bin bin
d example /example/pkgmk/sbin 1777 bin bin
f example /example/pkgmk/sbin/lsps 555 bin bin
f example /example/pkgmk/sbin/lsuser 555 bin bin
```

   Then, create the package with the above *Prototype* and **pkginfo** files using the **pkgmk** command:

```
pkgmk -d /tmp -f /home/myuser/example/pkgmk/prototype
```

   This produces the following output:

```
Building pkgmap from package prototype file
## Processing pkginfo file
    WARNING:parameter <CLASSES> set to "example"

## Attempting to volumize 5 entries in pkgmap
Part  1 -- 218 blocks, 10 entries
/tmp/mypkgA/pkgmap
/tmp/mypkgA/pkginfo
/tmp/mypkgA/root/example/pkgmk/sbin/lsps
/tmp/mypkgA/root/example/pkgmk/sbin/lsuser
## Packaging complete
```

   The newly created package named mypkgA now exists in /tmp/mypkgA.

Exit Status

Item	Description
0	Successful completion of script.
1	Fatal error. Installation process is terminated at this point.
99	Internal error.

Files

Item	Description
/usr/sbin/pkgmk	Contains the **pkgmk** command.

pkgparam Command

Purpose

Displays package parameter values.

Syntax

To Display the Value of a Parameter Contained in pkginfo

pkgparam [**-v**] [**-d** *Device*] [-P *path*] *Pkginst* [*Param* **...**]

To Display the Value of a Parameter Contained in a Device

pkgparam -d *Device* [**-v**] [*Param* **...**]

To Display the Value of a Parameter Contained in a File

pkgparam -f *File* [**-v**] [*Param* **...**]

Description

pkgparam displays the value associated with the parameter or parameters requested on the command line. The values are located in one of the following places: in the **pkginfo** file for *Pkginst*, on the *Device* named with the **-d** flag, or on the specific file named with the **-f** flag. When a *Device* is given, but a *Pkginst* is not (as shown in the second synopsis), parameter information for all packages residing on *Device* is shown.

If packages are installed in an alternative root directory path using the pkgadd command with the -P option, package parameters can be requested using the same alternative root directory path specified with the -P option.

One parameter value is shown per line. Only the value of a parameter is given unless the **-v** flag is used. With this flag, the output of the command is in this format:

```
Parameter1='Value1'
Parameter2='Value2'
Parameter3='Value3'
```

If no parameters are specified on the command line, values for all parameters associated with the package are shown.

Flags

Item	Description
-v	Specifies verbose mode. Displays name of parameter and its value.

Item	Description
-d *Device*	Specifies the *Device* on which a *Pkginst* is stored. *Device* can be the full pathname to a directory (such as **/var/tmp**), or "**-**" which specifies packages in datastream format read from standard input.
-f	Requests that the command read *File* for parameter values. This file should be in the same format as a **pkginfo** file. As an example, such a file might be created during package development and used while testing software during this stage.
Pkginst	Defines a specific package for which parameter values should be displayed. The format *Pkginst*.* can be used to indicate all instances of a package. When using this format, enclose the command line in single quotes to prevent the shell from interpreting the "*" character.
Param	Defines a specific parameter whose value should be displayed.
-P *path*	Searches for the `pkginfo` file in the alternate root directory path.

Exit Status

If parameter information is not available for the indicated package, the command exits with a non-zero status.

Item	Description
0	Successful completion of script.
1	Fatal error. Installation process is terminated at this point.

Files

Item	Description
/var/spool/pkg	default spool directory
/usr/sbin/pkgparam	Contains the **pkgparam** command.

pkgproto Command

Purpose

Generates a prototype file.

Syntax

pkgproto [**-i**] [**-c** *Class*] [*Path1* [**=***Path2*] ...]

Description

The **pkgproto** commands scans the indicated paths and generates a prototype file that may be used as input to the **pkgmk** command. To do this, the standard output of this command must be redirected to a file. The file can then be used when invoking **pkgmk**.

If no *Paths* are specified on the command line, standard input is assumed to be a list of *Paths*. If the *Path* listed on the command line is a directory, the contents of the directory are searched. However, if input is read from stdin, a directory specified as a path is not searched.

The prototype file attributes *mac*, *fixed*, and *inherited*, cannot be determined by **pkgproto** and must be manually added to the file.

By default, **pkgproto** creates symbolic link entries for any symbolic link encountered (ftype=s). When you use the **-i** flag, **pkgproto** creates a file entry for symbolic links (ftype=f). The prototype file must be edited to assign file types such as v (volatile), e (editable), or x (exclusive directory). **pkgproto** detects linked files. If multiple files are linked together, the first path encountered is considered the source of the link.

The output from this command is sent to standard output. You must redirect standard output to a file if you wish to use the result as a prototype file when invoking **pkgmk**. Since **pkgmk** uses prototype as the default filename for the prototype file, we suggest you direct the output of **pkgproto** to the file name prototype.

You must add entries to the prototype file produced by this command for any installation scripts and files your package may need. At minimum, you will need an entry for the **pkginfo** file. You may also need entries for any of the following files you use in your package: **copyright**, **compver**, **depend**, **setinfo**, **space**, any installation or removal scripts you define for the package, and/or any classes you define.

Note:

1. By default, **pkgproto** creates symbolic link entries for any symbolic link encountered (ftype=s). When you use the `-i` option, **pkgproto** creates a file entry for symbolic links (ftype=f). The prototype file must be edited to assign file types such as v (volatile), e (editable), or x (exclusive directory). **pkgproto** detects linked files. If multiple files are linked together, the first path encountered is considered the source of the link.
2. The output from this command is sent to standard output. You must redirect standard output to a file if you wish to use the result as a prototype file when invoking **pkgmk**. Since **pkgmk** uses prototype as the default filename for the prototype file, we suggest you direct the output of **pkgproto** to the file name **prototype**.
3. Note that you must add entries to the **prototype** file produced by this command for any installation scripts and files your package may need. At minimum, you will need an entry for the **pkginfo** file; see **pkginfo** for more information. You may also need entries for any of the following files you use in your package: **copyright**, **compver**, **depend**, **setinfo**, **space**, any installation or removal scripts you define for the package, and/or any classes you define, (e.g., postinstall).

Flags

Item	Description
-i	Ignores symbolic links and records the paths as ftype=f (a file) versus ftype=s (symbolic link).
-c *Class*	Maps the class of all paths to *Class*.
Path1	Path of directory where objects are located.
Path2	Path that should be substituted on output for *Path1*.

Examples

The following examples show uses of **pkgproto** and a partial listing of the output produced.

1.
```
$ pkgproto /usr/bin=bin /usr/usr/bin=usrbin /etc=etc
f none bin/sed=/bin/sed 0775 bin bin
f none bin/sh=/bin/sh 0755 bin daemon
f none bin/sort=/bin/sort 0755 bin bin
d none etc/master.d 0755 root daemon
f none etc/master.d/kernel=/etc/master.d/kernel 0644 root daemon
f none etc/rc=/etc/rc 0744 root daemon
```

2.
```
   $ find / -type d -print | pkgproto
   d none / 755 root root
   d none /usr/bin 755 bin bin
   d none /usr 755 root root
   d none /usr/bin 775 bin bin
   d none /etc 755 root root
   d none /tmp 777 root root
```

3. Identical to the previous example, but with the output captured in a file for later processing with **pkgmk**. Entries added for the required **pkginfo** file, and, for instance, a postinstall script that might be executed after the files are copied into the correct locations.

```
$ find / -type d -print | pkgproto >prototype
$ (edit the file to add entries for pkginfo and postinstall)
$ cat prototype
i pkginfo
i postinstall
d none / 755 root root
d none /usr/bin 755 bin bin
d none /usr 755 root root
d none /usr/bin 775 bin bin
d none /etc 755 root root
d none /tmp 777 root root
```

Return Codes

Item	Description
0	Successful completion of script.
1	Fatal error. Installation process is terminated at this point.

Files

Item	Description
/usr/sbin/pkgproto	Contains the **pkgproto** command

pkgrm Command

Purpose

Removes a package or set from the system.

Syntax

To Remove an Installed Software Package

pkgrm [**-n**] [**-a** *Admin*] [-P *Path*] [*Pkginst1* [*Pkginst2* [. . .]]]

To Remove a Software Package from a Spool Device

pkgrm -s *Spool* [*Pkginst*]

Description

pkgrm removes a previously installed or partially installed package/set from the system. A package is a collection of related files and executables that can be independently installed. A set is made up of a special-purpose package, referred to as a Set Installation Package (SIP), and a collection of one or more packages that are members of the set.

pkgrm checks that all packages listed on the command line are on the system. If any of the packages listed does not exist, no changes are made to the system, that is, none of the listed packages are removed.

A check is also made to determine if any other packages depend on the one being removed. The action taken if a dependency exists is defined in the *Admin* file (see the **-a** flag, below).

The default state for the command is interactive mode, meaning that prompt messages are given during processing to allow the administrator to confirm the actions being taken. Non-interactive mode can be requested with the **-n** flag.

The **-s** flag can be used to specify the directory from which spooled packages should be removed.

Flags

Item	Description
-n	Enables non-interactive mode. If there is a need for interaction, the command exits. Use of this flag requires that at least one package instance be named upon invocation of the command.
-a *Admin*	Defines an installation administration file, *Admin*, to be used in place of the default administration file. [For a description of the format of an *Admin* file, see the **admin** file format.] The token "none" overrides the use of any *Admin* file, and thus forces interaction with the user. Unless a full pathname is given, **pkgrm** looks in the **/var/sadm/install/admin** directory for the file. By default, the file **default** in that directory is used.
-P *Path*	Removes the specified packages from the alternative root directory path.
-s *Spool*	Removes the specified package(s) from the directory *Spool*.
Pkginst	Defines a short string used to designate an abbreviation for the package/set name. (The term "package instance" is used loosely: it refers to all instantiations of *Pkginst*, even those that do not include instance identifiers.) If *Pkginst* specifies a SIP, all installed packages which are members of the set, and the SIP itself, are removed in reverse dependency order. To indicate all instances of a package, specify '*Pkginst*.*', enclosing the command line in single quotes, as shown, to prevent the shell from interpreting the "*" character. Use the token "all" to refer to all packages available on the source medium.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion of script.
1	Fatal error. Installation process is terminated at this point.
99	Internal error.

Files

Item	Description
/usr/sbin/pkgrm	Contains the **pkgrm** command

pkgtrans Command

Purpose

Translates package format.

Syntax

pkgtrans [**-i** **-o** **-n** **-s**] [**-z** *Blocksize*] *Device1* *Device2* [*Pkginst1* [*Pkginst2* [...]]]

Description

The **pkgtrans** command translates an installable package from one format to another. It translates the following:

- A file system format to a datastream
- A datastream to a file system format

You cannot run **pkgtrans** from **csh**.

Flags

Item	Description
-i	Copies the **pkginfo** and *Pkgmap* files. If the packages category is defined as "set" for Set Installation Packages (SIPs) (see **setinfo** file format), then that packages' **setinfo** file is also copied.
-o	Overwrites the same instance on the destination device. The package instance is overwritten if it already exists.
-n	Creates a new instance of the package on the destination device. If the package instance already exists on the destination device, it is left unchanged and a new instance is created. The new instance has a sequence number attached to distinguish it from the existing instance. For example, assume the destination device already contained an instance of package *X*. If you use **pkgtrans** with the **-n** flag to write a new instance of package *X* to the device, the existing instance of package *X* remains on the destination device, and a new instance, called *X.2*, would be created on the device. If you executed **pkgtrans** again with the **-n** flag, a third instance, called *X.3*, would be created.
-s	Indicates that the package should be written to *Device2* as a datastream rather than as a file system. The default behavior is to write to *Device2* in the file system format.
-z *Blocksize*	Indicates the blocksize to be used when transferring to cartridge tape. Packages that have been written to tape using the **-z** flag and a value not equal to 512 are always read using a blocksize of 32768. Thus, the **-z** flag is not applicable when reading from cartridge tape.
Device1	Indicates the source device. Can be - (hyphen) which specifies packages in datastream format read from standard input. The package or packages on this device are translated and placed on *Device2*. If *Device1* is a regular file or directory, you must use the absolute pathname, rather than a relative pathname.
Device2	Indicates the destination device. Can be - (hyphen) which specifies packages written to standard output in datastream format. Translated packages are placed on this device. If *Device2* is a regular file or directory, you must specify it as an absolute pathname, rather than a relative pathname.
Pkginst	Specifies which package on *Device1* should be translated. The token "all" may be used to indicate all packages. *Pkginst.** can be used to indicate all instances of a package. If no packages are defined, a prompt shows all packages on the device and asks which to translate. If a set is being transferred to datastream format, the *Pkginst* arguments should begin with the SIP and be followed by the packages listed in the SIP's **setinfo** file, in the order in which they appear in that file.

Note: By default, **pkgtrans** does not transfer any instance of a package if any instance of that package already exists on the destination device. Use of the **-n** flag creates a new instance if an instance of this package already exists. Use of the **-o** flag overwrites the same instance if it already exists. Neither of these flags are useful if the destination device is a datastream, because the entire datastream is overwritten anyway.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion of script.
1	Fatal error. Installation process is terminated at this point.

Files

Item	Description
/usr/sbin/pkgtrans	Contains the **pkgtrans** command.

Examples

1. To translate all packages located on drive *Device* and place the translations in **/tmp**, type:

```
pkgtrans Device /tmp all
```

2. To translate "pkg1" and "pkg2" in **tmp** and place them on *Device* in a datastream format, type:

```
pkgtrans -s /tmp Device pkg1 pkg2
```

platform_dump Command

Purpose

Perform platform (Hardware & Firmware) dump related actions.

Syntax

`platform_dump` { `-c` | `-d` | `-e` | `-f` *fstype* | `-F` *flag* | `-l` | `-q` | `-S` | `-s` *seq_no* } [`-L`]

Description

The `platform_dump` command was introduced in AIX to support hardware and firmware problem determination for POWER5 platforms. You can use this command to help the operating system save firmware-related and hardware-related dumps. This command is supported only on partitions which have service authority enabled, except for Hardware Management Console (HMC) managed systems. On an HMC managed system, the dumps go to the HMC. The `platform_dump` command is normally run by operating system functions such as base installation and dumpcheck. Platform dumps contain:

- The hardware state
- The hypervisor state
- The FSP (Flexible Service Processor) state information

The disk space for platform dump files is reserved using the `platform_dump` command. A dedicated logical volume, **/dev/fwdump**, is created in the rootvg volume group and mounted on the **/var/adm/ras/platform** directory. The **fwdump_dev** device and **fwdump_dir** mount point are both saved in ODM in the SWservAt object class. During installation, AIX utilizes the `platform_dump` command to reserve the necessary disk space. The disk space is reserved only if the partition is designated to be a service partition. The maximum possible size for the platform dumps is indicated to AIX so that enough space can be allocated beforehand for the platform dumps. Note that this size can change dynamically. The operating system detects this and informs you about the extra requirements and automatically expands the logical volume if possible.

Note: If you assign service partition authority to an AIX partition after the partition was installed, run the `platform_dump -f <fstype>` command to create the **/dev/fwdump** rootvg logical volume. The *fstype* argument can have the **jfs2** or **jfs** value.

The `-L` flag is provided to record command output to the error log.

Flags

Item	Description
`-c`	Performs a check on the estimated platform dump size (as indicated by the firmware) and the disk space allocated for the platform dumps. It will report the following: If estimated size is less than or equal to allocated space, will return 0. If estimated size is greater than allocated space, will return 1.
`-d`	Deletes the file system space reserved for platform dumps and free up the same for other uses. Any existing dump files on the reserved disk space will be lost.
`-e`	Provides an estimate of disk space required to save the platform dumps when they occur. This option will interact with the firmware to provide this estimate. It is expected that, based on this space information, the user will have enough disk space allocated for platform dumps to be saved. The value output will be the required size in bytes.
`-f` *fstype*	Reserves enough disk space on the system for platforming dumps. The `-f` option will create a file system (if one does not exist) exclusively for platform dumps. If a file system already exists and the size is not enough, the file system size will be increased. The *fstype* must be a valid file system type. If the file system already exists, `any` may be specified.
`-F` *flag*	Enables or disables platform dumps. If flag is 0, platform dumps are disabled, if 1, platform dumps are enabled.
`-l`	Lists the current configuration of platform dump.
`-L`	Tells `platform_dump` to log its output as well as displaying it. This does not apply to the size output by the `-e` option.
`-q`	Checks whether the platform supports platform dumps or not. Will return 0 if platform dump is supported.
`-s` *seq_no*	Saves the platform dump from the firmware as identified in the dump notification event. *seq_no* indicates the sequence number of the dump notification event as stored in the AIX error log file. This sequence number will be used by this command to parse the detailed data area and obtain dump tag and dump type information needed to obtain the dump data from firmware.
`-S`	Saves the scan dumps on systems which support scan data. When this option is specified, the command will check for the existence of a scan dump, and if so will read and save the `scandump` data from firmware using the existing scan dump interface.

Exit Status

0
On successful completion.

1
Returned if `-c` was specified, and there is insufficient space to save platform dumps.

255
Returned if platform dumps are not supported on the system.

3
Returned if platform dumps has been disabled.

2
Returned in an error is encountered.

Security

The `platform_dump` may only be executed by the root user.

Example

1. To get an estimate of the platform dumps size, type the following:

   ```
   platform_dump -e
   ```

   This will report the estimated platform dump size in bytes.

plotgbe Command

Purpose

Plots HP-GL files to a plotter device.

Syntax

/usr/lpd/plotgbe [**-fr=***X*] [**-noin**] *File*

Description

The **plotgbe** command is a backend program which plots HP-GL files to a plotter device. The plotter device must be attached to a 5085/5086 workstation via the 5080 Attachment Adapter. To use the **plotgbe** command, you must define a print queue for the **plotgbe** backend program. See **enq** command, use the **-o** flag to pass options to the **plotgbe** backend for processing.

The **plotgbe** backend command also generates the appropriate HP-GL commands for plotter initialization and plot scaling. This data is sent to the plotter before the user-specified HP-GL file is sent. Thus, any scaling or initialization commands included in the HP-GL file override those generated by the **plotgbe** backend command.

> **Note:** The user must have read access to the file sent to the **plotgbe** command with the print request command.

Flags

Item	Description
-fr=*X*	Provides for plotting multi-frame drawings. This option causes *X* number of frames to be plotted, where *X* is a number in the range 1 through 9. For example, plotting a 20' drawing on E-size role media may require 5 frames. Thus, the option `fr=5` would be passed to the **plotgbe** backend.
-noin	Allows plotter front panel settings to remain in effect for the current plot without being reset to default values. Normally, the P1 and P2 positions which define the plot page on the plotter are set by the **plotgbe** command to their default location. Use the **-noin** no-initialization option to override the default locations.

Examples

1. To send the file `longaxis.gl` to the `plt` plotter queue and specify to the backend that the file requires five frames to print, enter:

```
enq -Pplt -o -fr=5 longaxis.gl
```

2. To send the file `plotdata.gl` to the `plt` plotter queue, specifying that the plot page positions are not to be reset to default for this file, enter:

```
enq -Pplt -o -noin plotdata.gl
```

3. To send the file `twoplot.gl` to the `plt` plotter queue, specifying no plot page initialization and that the plotter print the drawing in two frames, enter:

```
enq -Pplt -o -noin -o fr=2 twoplot.gl
```

Files

Item	Description
/usr/lpd/plotgbe	Contains the **plotgbe** command.

plotlbe Command

Purpose

Plots HP-GL files to a plotter device.

Syntax

/usr/lpd/plotlbe [**-fr=***X*] [**-noin**] *File*

Description

The **plotlbe** command is a backend program which plots HP-GL files to a plotter attached to a serial port defined as a TTY device. To use the **plotlbe** command, you must define a TTY device for the serial port and define a print queue for the **plotlbe** backend program.

When configuring the TTY serial port, set the baud-rate, parity, and stop bits to the appropriate settings for your plotter. You must also set XON/XOFF to FALSE for your TTY port.

The **plotlbe** command is called by the **qdaemon** process. It should not be entered on the command line. Any options needed for a specific print request to a plotter should be passed to the **plotlbe** command with the command used to request a print job (usually the **enq** command). With the **enq** command, use the **-o** flag to pass options to the **plotlbe** backend for processing.

The **plotlbe** backend command supports the following plotters: 7731, 7372, 7374, 7375-1, 7375-2, 6180, 6182, 6184, 6186-1, and 6186-2.

The **plotlbe** command supports ENQ/ACK handshaking. Refer to your plotter programming manual for more information on handshaking.

The **plotlbe** backend command also generates the appropriate HP-GL commands for plotter initialization and plot scaling. This data is sent to the plotter before the user-specified HP-GL file is sent. Thus, any scaling or initialization commands included in the HP-GL file override those generated by the **plotlbe** backend command.

> **Note:** The user must have read access to the file sent to the **plotlbe** command with the print request command.

Flags

Item	Description
-fr=*X*	Provides for plotting multi-frame drawings. This option causes *X* number of frames to be plotted, where *X* is a number in the range 1 through 9. For example, plotting a 20' drawing on E-size roll media may require 5 frames. Thus, the option `-fr=5` would be passed to the **plotlbe** backend.
-noin	Allows plotter front panel settings to remain in effect for the current plot without being reset to default values. Normally, the P1 and P2 positions which define the plot page on the plotter are set by the **plotlbe** command to their default locations. Use the **-noin** no-initialization option to override the default locations.

Examples

1. To send the file `longaxis.gl` to the `plt` plotter queue and specify to the backend that the file requires five frames to plot, enter:

```
enq -Pplt -o -fr=5 longaxis.gl
```

2. To send the file `plotdata.gl` to the `plt` plotter queue, specifying that the plot page positions are not to be reset to default for this file, enter:

```
enq -Pplt -o -noin plotdata.gl
```

3. To send the file `twoplot.gl` to the `plt` plotter queue, specifying no plot page initialization and that the plotter print the drawing in two frames, enter:

```
enq -Pplt -o -noin -o fr=2 twoplot.gl
```

Files

Item	Description
/usr/lpd/plotlbe	Contains the **plotlbe** command.

pmctl Command

Purpose

Starts, resets, or stops generating Performance Monitor events.

Syntax

pmctl [{ [**-E** [*mode*]] [**-f** *interval*] {[**-y** *command*]} } | [**-h**] | [**-r**] | [**-S**]] [**-s**] [{ **-a** **-y** *command* [**-f** *interval*] }]

Description

The **pmctl** command starts, stops, or resets the generation of Performance Monitor events in the PMAPI subsystem to support manual offline mode with the **tprof -E** command. It also reports the current status of the PMAPI subsystem.

Flags

Item	Description
-a	Turns on large page analysis.

Item	Description
-E [*mode*]	Enables event-based profiling. You can specify one of the following modes: **PM_*event*** Specifies the hardware event to profile. If no mode is specified for the **-E** flag, the default event is processor cycles (**PM_CYC**). **EMULATION** Enables the emulation profiling mode. **ALIGNMENT** Enables the alignment profiling mode. **ISLBMISS** Enables the Instruction Segment Lookaside Buffer miss profiling mode. **DSLBMISS** Enables the Data Segment Lookaside Buffer miss profiling mode.
-f *interval*	Specifies the sampling interval to use. • For processor cycle, **EMULATION**, **ALIGNMENT**, **ISLBMISS**, and **DSLBMISS** events, specify 1 to 500 milliseconds (default = 10). • For other Performance Monitor events, specify 10000 up to MAXINT occurrences (default = 10000). If you use the **-f** flag with the **-y** flag, specify 1 up to MAXINT occurrences for other Performance Monitor events (default = 10000).
-h	Prints man page information.
-r	Releases and resets the PMAPI subsystem.
-S	Stops generating Performance Monitor events.
-s	Prints the current status of the PMAPI subsystem.
-y *command*	Turns on the event-based profiling only for the specified command and its descendents.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To stop generating Performance Monitoring events, enter the following command:

```
pmctl –S
```

2. To reset generating Performance Monitoring events, enter the following command:

```
pmctl –r
```

3. To report the current status of the PMAPI subsystem, enter the following command:

```
pmctl –s
```

4. To start generating Performance Monitoring events, enter the following command:

```
pmctl –E
```

5. To start generating Performance Monitoring events only for the specified **workload** command and its descendents, enter the following command:

```
pmctl –E –y workload
```

6. To support the **tprof -E** command in manual offline mode, enter the following command:

```
trace -adf -o mydata.trc
trcon
pmctl –E
sleep 10; trcstop
gensyms > mydata.syms
tprof –suker mydata
```

7. To support the **tprof -E** command in manual offline mode profiling for the specified **workload** command and its descendents, enter the following command:

```
trace -adf -o mydata.trc
trcon
pmctl –E –y workload
trcstop
gensyms > mydata.syms
tprof –suker mydata
```

pmcycles Command

Purpose

Measures processor clock speed.

Syntax

pmcycles [**-d**] [**-m**]

Description

The **pmcycles** command displays the *nominal processor speed* for the system in MHz. The nominal processor speed is the maximum frequency at which the system can run across all environments and workload conditions. Depending on system conditions, the nominal processor frequency might not represent the minimum or maximum achievable processor speed.

The **lparstat -E 1 1** and **mpstat -E 1 1** commands must be used to determine the current processor speed. The **pmcycles** command might be deprecated in the future.

Flags

Item	Description
-d	Displays the decrementer in MHz and nanoseconds per tick.
-m	Displays the speed of each of the processors.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the processor speed, type:

```
pmcycles
```

   Output similar to the following appears:

```
This machine runs at 133 MHz
```

2. To display each processor speed, type:

```
pmcycles -m
```

   Output similar to the following appears:

```
Cpu 0 runs at 200 MHz
CPU 1 runs at 200 MHz
```

pmlist Command

Purpose

Lists information about supported processors.

Syntax

pmlist [**-h**]

pmlist -l [**-o t** | **c** | **x**]

pmlist { **-s** | **-e** *ShortName* | **-c** *Counter* [,*event*] | **-g** *Group* | **-S** *Set* | **-D** *DerivedMetricID* | **-m** *MetricGroup* | **-V** *Variable* } [**-p** *ProcessorType*] [**-s**] [**-d**] [**-o t** | **c** | **x**] [**-f** *Filter*]

Description

The **pmlist** command performs the following functions:

- List the supported processors.
- List the information summary for a specified processor.
- List the event table for a specified processor.
- List any existing event groups for a specified processor.
- List any existing event sets for a specified processor.
- List the event set and formula for a specified derived metric.
- List the variables in the derived metric files.

Flags

Item	Description
-c -1	Lists all events for all counters.
-c *Counter*	Lists all events for the specified *Counter*.
-c *Counter*,*Event*	Lists the specified *Event* for the specified *Counter*.
-d	Displays event detailed description.
-D -1	Displays all the derived metrics supported.
-D *DerivedMetricID*	Displays the specified *DerivedMetricID*.

Item	Description
-e *ShortName*	Lists the description of the specified *ShortName* for all Counters.
-f v,**u**,**c**	Specifies the event filter as a comma-separated list of filters. The valid filters are: **v** (verified) , **u** (unverified) , and **c** (caveat). These filters represent the testing status of an event. The default filter is **v,u,c**.
-g -1	Lists all event groups.
-g *Group*	Lists the specified event *Group*.
-h	displays help information for the **pmlist** command.
-l	Lists all supported processor types.
-m -1	Lists all derived metrics by metric group.
-m *MetricGroup*	Displays all the derived metrics that pertain to the specified *MetricGroup*.
-o t | **c** | **x**	Specifies the output format for the **pmlist** command. The valid output formats are specified as one of: **t** (text format), **c** (CSV format) and **x** (XML format). The default output format is text.
-p *ProcessorType*	Specifies the processor type.
-s	Displays the processor information summary.
-S -1	Displays all the event sets supported.
-S *Set*	Displays the specified event *Set*.
-V -1	Displays all the variables that are used to calculate derived metrics.
-V *Variable*	Displays the specified variable.

Examples

1. To display the list of all supported processors, type:

```
pmlist -l
```

2. To display a summary information for the current processor, type:

```
pmlist -s
```

3. To display a summary information for the current processor in CSV format, type:

```
pmlist -s -o c
```

4. To display group number 62 for the current processor (if the current processor supports event groups), type:

```
pmlist  -g 62
```

5. To display detailed information for event 3 of counter 1 of POWER4 processor, type:

```
pmlist -p POWER4 -c 1,3 -d
```

6. To display set number 2 for the current processor (if the current processor supports event sets), type:

```
pmlist -S 2
```

pmtu Command

Purpose

Displays and deletes Path MTU discovery related information.

Syntax

pmtu [**-inet6**] **display**/[**delete** [**-dst** *destination*] [**-gw** *gateway*]]

Description

The **pmtu** command is provided to manage the Path MTU information. The command can be used to display the Path MTU table. By default the Ipv4 pmtu entries are displayed. Ipv6 pmtu entries can be displayed using the **–inet6** flag. This command also enables a root user to delete a pmtu entry with the **pmtu delete** command. The delete can be based on destination, gateway, or both.

A pmtu entry gets added into the PMTU table when a route add occurs with an MTU value.

A network option, **pmtu_expire**, is provided to expire unused pmtu entries. The default value of **pmtu_expire** is 10 minutes.

Flags

Item	Description
-dst	Specifies the destination of the pmtu entry to be deleted.
-gw	Specifies the gateway of the pmtu entry to be deleted.
-inet6	Specifies to display or delete Ipv6 pmtu entry.

Exit Status

Item	Description
0	The command completed successfully.
1	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display Ipv4 pmtu entries, type:

   ```
   pmtu display
   ```

   The output will look similar to the following:

   ```
   dst           gw               If    pmtu     refcnt   redisc_t    exp
    ------------------------------------------------------------------------
   ```

```
192.168.5.5   192.168.10.33    en2    1500        1          0       0
```

The reference count signifies the number of current TCP and UDP applications using this pmtu entry.

The `redisc_t` entry signifies the amount of time that is elapsed since the last Path MTU discovery attempt. The PMTU is rediscovered after every *pmtu_rediscover_interval* minutes. Its default value is 30 minutes and can be changed using the **no** command.

The PMTU entry expiry is controlled by the network option *pmtu_expire*. Its default value is 10 minutes. This value can be changed using the **no** command. A value of 0 does not expire any entries. The `exp` entry signifies the expiry time. PMTU entries having more than zero `refcnt` have `exp` of 0. When the `refcnt` becomes zero, the `exp` time increases every minute and the entry gets deleted when the **exp** variable becomes equal to *pmtu_expire*.

2. To delete an entry based on destination, type:

```
pmtu delete -dst 192.168.5.5
```

3. To display Ipv6 , type:

```
pmtu -inet6 display
```

Output will look similar to the following:

```
dst                       gw      If    pmtu     refcnt   redisc_t   exp
 ------------------------------------------------------------------------
fe80::204:acff:fee4:ab3b  ::    lo0   16896        2          2        0
```

Location

/usr/sbin/pmtu

Files

Item	Description
/usr/sbin/pmtu	Contains the **pmtu** command.

pop3d Daemon

Purpose

Starts the Post Office Protocol Version 3 (POP3) server process.

Syntax

pop3d [**-c**]

Description

The **pop3d** command is a POP3 server. It supports the POP3 remote mail access protocol. Also, it accepts commands on its standard input and responds on its standard output. You normally invoke the **pop3d** command with the **inetd** daemon with those descriptors attached to a remote client connection.

The **pop3d** command works with the existing mail infrastructure consisting of **sendmail** and **bellmail.**

Flags

Item	Description
-c	Suppresses the reverse host name lookup.

Parameters

Item	Description
None	

Exit Status

All error and status information is written to a logfile if **syslogd** is configured for logging.

Security

The **pop3d** daemon is a PAM-enabled application with a service name of *imap*. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the **usw** stanza of **/etc/security/login.cfg**, to PAM_AUTH as the root user.

The authentication mechanisms used when PAM is enabled depend on the configuration for the **imap** service in **/etc/pam.conf**. The **pop3d** daemon requires **/etc/pam.conf** entries for the **auth** and **session** module types. Listed below is a recommended configuration in **/etc/pam.conf** for the **imap** service:

```
#
# AIX imap configuration
#
imap auth      required     /usr/lib/security/pam_aix

imap session   required     /usr/lib/security/pam_aix
```

Note: Because the **pop3d** daemon uses the **imap** library for authentication, the **imap** service is used for both the **imapd** and **pop3d** daemons.

Files

Item	Description
/usr/sbin/pop3d	Contains the **pop3d** command.
html	

pop3ds Daemon

Purpose

Starts the Post Office Protocol Version 3 (POP3) server process over TLS/SSL.

Syntax

pop3ds [**-c**]

Description

The **pop3ds** command is a POP3 server. It supports the POP3 remote mail access protocol. Also, it accepts commands on its standard input and responds on its standard output. You normally invoke the **pop3d3** command with the **inetd** daemon with those descriptors attached to a remote client connection.

The **pop3ds** command works with the existing mail infrastructure consisting of **sendmail** and **bellmail.**

Flags

Item	Description
-c	Suppresses the reverse host name lookup.

Parameters

Item	Description
None	

Exit Status

All error and status information is written to a logfile if **syslogd** is configured for logging.

Security

The **pop3ds** daemon is a PAM-enabled application with a service name of *imap*. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the **usw** stanza of **/etc/security/login.cfg**, to PAM_AUTH as the root user.

The authentication mechanisms used when PAM is enabled depend on the configuration for the **imap** service in **/etc/pam.conf**. The **pop3ds** daemon requires **/etc/pam.conf** entries for the **auth** and **session** module types. Listed below is a recommended configuration in **/etc/pam.conf** for the **imap** service:

```
#
# AIX imap configuration
#
imap auth      required     /usr/lib/security/pam_aix

imap session   required     /usr/lib/security/pam_aix
```

Note: Because the **pop3ds** daemon uses the **imap** library for authentication, the **imap** service is used for both the **imapds** and **pop3ds** daemons.

Files

Item	Description
/usr/sbin/pop3ds	Contains the **pop3ds** command.
html	

portmap Daemon

Purpose

Converts RPC program numbers into Internet port numbers.

Syntax

/usr/sbin/portmap

Description

The **portmap** daemon converts RPC program numbers into Internet port numbers.

When an RPC server starts up, it registers with the **portmap** daemon. The server tells the daemon which port number it is listening to and which RPC program numbers it serves. Thus, the **portmap** daemon

knows the location of every registered port on the host and which programs are available on each of these ports.

A client consults the **portmap** daemon only once for each program the client tries to call. The **portmap** daemon tells the client which port to send the call to. The client stores this information for future reference.

Since standard RPC servers are normally started by the **inetd** daemon, the **portmap** daemon must be started before the **inetd** daemon is invoked.

> **Note:** If the **portmap** daemon is stopped or comes to an abnormal end, all RPC servers on the host must be restarted.

Flags

None

Examples

1. To start the **portmap** daemon, enter the following command:

```
startsrc -s portmap
```

2. To stop the **portmap** daemon enter the following command:

```
stopsrc -s portmap
```

Files

Item	Description
inetd.conf	Starts RPC daemons and other TCP/IP daemons.
/etc/rpc	Contains a list of server names and their corresponding **rpc** program numbers and aliases.

portmir Command

Purpose

Allows one TTY stream (monitor) to attach to another TTY stream (target) and monitor the user session that is taking place on that stream.

Syntax

portmir { **-d** *mir_modem* **-t** *target* [**-m** *monitor*] | **-t** *target* [**-m** *monitor*] | { **-o** | **-c** *monitor* | **-q** }

Description

The **portmir** command allows one TTY stream (monitor) to attach to another TTY stream (target) and monitor the user session that is taking place on that stream. This is accomplished by pushing a special "mirror" module into both the target and monitor TTY streams.

Both the target and monitor TTYs receive a printed message on their respective displays when a monitoring session begins. The monitoring session can be terminated from either the target TTY, monitor TTY, or a third TTY not involved in the monitoring session. When the monitor is used in a non-service mode, both streams must be in the open state (that is, either a getty or active session must be taking place on each TTY) in order for the command to work. This is necessary to allow the pushing of the "mirror" streams module. The **portmir** command is supported for use with TTY devices only (PTS, TTY, LFT).

The terminal type, as defined in the TERM environment variable, must be the same for both the monitor and target TTY. The value of this environment variable must correspond to a valid entry in the **terminfo** database. An example terminal type would be ibm3151 or vt100. The LFT is similar to the vt100. Terminal emulators such as `aixterm` are usually similar in function to vt100.

Although the console can be used as either the target TTY or the monitor TTY, using the console as the monitor TTY is not recommended. However, if the console is used as the monitor TTY, note that the console is first automatically redirected to the target TTY for the duration of the monitoring session. When the monitoring session is terminated, the console is redirected back to the device specified in the CuAt ODM database attribute **syscons**. If the console had been previously redirected, the redirection is not preserved.

Async devices that provide offloading of character processing may have problems if they are mirroring devices that rely on the line discipline (**ldterm**) to provide this function. An example of this would be the 128-port async adapter. Use the **chdev** command to disable the fastcook attribute if a port of a dissimilar adapter is monitored. Run the command as follows:

```
chdev -l tty1 -a fastcook -disable
```

Flags

Item	Description
-c *monitor*	Configures port for service boot by creating CuAt ODM database attribute **portmir_monitor**, which contains the **device** parameter as the value field. This device is used later as the default monitoring device when the **portmir** command is invoked in service mode (**-s** flag). Mirroring must be configured by the system administrator to execute at service boot time using the **-c** option. The target defaults to the device defined in the **portmir_monitor** attribute.
-d *mir_modem*	Sets monitoring port for dial-in purposes. Only the root user can issue the command with this flag. Ensure that **/usr/share/mir_modem** is linked to the correct modem setup file. **/usr/share/mir_modem** contains example files; you may need to create your own, depending on your type of modem.
-m *monitor*	Specifies monitoring device. If neither the **-m** option nor the **-s** option are specified, then the monitoring device defaults to the port on which the **portmir** command was run.
-o	Turns off monitoring and terminates the command.
-q	Queries the value set with the **-c** option.
-t *target*	Specifies target device to be monitored.

Security

Only a single mirror session may be running at any one time.

To mirror a port in the nonservice mode, place a list of users who may monitor them in a **.mir** file in your home directory (not required for the root user). When the **mirror** daemon begins running, the daemon checks to see who is on that port. It then checks to see if the user of the monitoring port is authorized to monitor that port.

The **.mir** file must have the format of a single user ID per line.

Attention: Running the **su** command to change to root user during a mirror session gives root authority to *both* users.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. After **user1** has placed **user2**'s login ID into **/u/user2/.mir** file, to mirror **user1** on target **tty1** from **user2** on monitor **tty2**, enter:

```
portmir -t tty1 -m tty2
```

2. To mirror target **tty1** to user on monitor **tty2** who is dialing in, enter:

```
portmir -t tty1 -m tty2 -d mir_modem
```

3. To set up mirroring for service boot, specifying the monitoring device during the service boot, enter:

```
portmir -c tty
```

4. To disable mirroring during the service boot, enter:

```
portmir -c off
```

5. To query the service boot mirroring device, enter:

```
portmir -q
```

Files

Item	Description
/usr/share/modems/mir_modem	Modem configuration file examples for setting up dial-in.
/usr/sbin/portmir	Contains the command file.

post Command

Purpose

Routes a message.

Syntax

post [**-alias** *File ...*] [**-format** | **-noformat**] [**-msgid** | **-nomsgid**] [**-filter** *File* | **-nofilter**] [**-width** *Number*] [**-verbose** | **-noverbose**] [**-watch** | **-nowatch**] *File*

Description

The **post** command routes messages to the correct destinations. The **post** command cannot be started by the user. The **post** command can be called only by other programs.

The **post** command searches a message for all components that specify a recipient's address and parses each address to check for the proper format. The **post** command then puts addresses into the standard format and calls the **sendmail**command. The **post** command also performs header operations, such as appending the `Date:` and `From:` components and processing the `Bcc:` component. The **post** command uses the *File* parameter to specify the name of the file to be posted.

> **Note:** The **post** command may report errors when parsing complex addresses (for example, `@A:harold@B.UUCP`). If you use complex addresses, use the **spost** command instead of the **post** command.

Flags

Item	Description
-alias *File*	Searches the specified mail alias file for addresses. This flag may be repeated to specify multiple mail alias files. The **post** command automatically searches the **/etc/mh/MailAliases** file.
-filter *File*	Uses the header components in the specified file to copy messages sent to `Bcc:` recipients.
-format	Puts all recipient addresses into a standard format for the delivery transport system. This flag is the default.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.
-msgid	Adds a message-identification component (such as `Message-ID:`) to the message.
-nofilter	Strips the `Bcc:` header from the message for the `To:` and `cc:` recipients. Sends the message with minimal headers to the `Bcc:` recipients. This flag is the default.
-noformat	Does not alter the format of the recipient addresses.
-nomsgid	Does not add a message-identification component to the message. This flag is the default.
-noverbose	Does not display information during the delivery of the message to the **sendmail** command. This flag is the default.
-nowatch	Does not display information during delivery by the **sendmail** command. This flag is the default.
-verbose	Displays information during the delivery of the message to the **sendmail** command. This information allows you to monitor the steps involved.
-watch	Displays information during the delivery of the message by the **sendmail** command. This information allows you to monitor the steps involved.
-width *Number*	Sets the width of components that contain addresses. The default is 72 columns.

Files

Item	Description
/etc/mh/MailAliases	Contains the default mail aliases.
/etc/mh/mtstailor	Contains MH command definitions.

pppattachd Daemon

Purpose

Attaches an asynchronous device stream to the PPP (Point to Point Protocol) subsystem. Can be invoked as a daemon or a normal process.

Syntax

To Use a Specific tty Port as a Connection (Runs as a Daemon):

pppattachd /dev/tty*PortNumber* { **client** | **server** | **demand** } { **ip** | **ipv6** | **ip ipv6** } [**multilink**] [**connect** "*ConnectorProgram*"] [**inactive** *Seconds*] [**authenticate pap** | **chap**] [**peer pap** | **chap**] [**user** *Name*] [**remote** *HostName*] [**nodaemon**]

To Use Standard In and Standard Out as the tty Device (Runs as a Process):

pppattachd { **client** | **server** | **demand** } { **ip** | **ipv6** | **ip ipv6** } [**multilink**] [**inactive** *Seconds*] [**authenticate pap** | **chap**] [**peer pap** | **chap**] [**user** *Name*] [**remote** *HostName*] [**nodaemon**]

Description

The **pppattachd** daemon provides the mechanism to bind an asynchronous stream to the PPP subsystem. When placing an out going connection on a specific tty port, **pppattachd** becomes a daemon. When using stdin (standard in) and stdout (standard out) as the tty device for PPP communications **pppattachd** does not become a daemon. (It would be executed from the **$HOME/.profile** upon login on a tty device.)

You can activate PAP or CHAP authentication with the **authenticate** and **peer** options. Use the **smit** command to create entries in either the **/etc/ppp/pap-secrets** or **/etc/ppp/chap-secrets** file. The **pppattachd** daemon uses the passwords in these files to authenticate the connection. It searches only the **/etc/ppp/pap-secrets** file for PAP authentication and the **/etc/ppp/chap-secrets** file for CHAP authentication.

The multilink option is to used to identify the PPP link as having several attachments between the two PPP peers. PPP packets are fragmented at one peer, sent over the multiple attachments, and then reconnected on the remote peer that must also support multilink. The maximum receive reconstruction unit (MMRU) and endpoint descriptor are set through SMIT on the PPP Link Configuration menu. MRRU is the maximum data size before fragmentation. The endpoint discriminator uniquely identifies the local system.

Errors and messages are logged using the **syslog** facility.

Options

Item	Description
authenticate pap | **chap**	Defines the current system as the authenticator of either PAP or CHAP.
client | **server** | **demand**	Defines the type of subsystem connection to be bound to on the system running the daemon.
ip | **ipv6** | **ip ipv6**	Specifies protocol types.
connect "*ConnectorProgram*"	Specifies the program to use to place an outgoing connection. The tty device opened is passed as stdin and stdout to the program. The **/usr/sbin/pppdial** command is a connector program that can be used.
inactive *Seconds*	Specifies the number (unsigned integer) of seconds to wait for inactivity on the link before terminating the connection. The default value is 0 (no timeout).
multilink	Identifies the PPP link as having a group of attachments connecting to two PPP peers.
nodaemon	Specifies to the attachment process that it is not to become a daemon. You must use this option for attachment processes that are invoked with demand connections.
peer pap | **chap**	Defines the current system as the peer of either PAP or CHAP.

Item	Description
remote *HostName*	Defines the remote host name to be used for PAP authentication. An entry for *UserName RemoteHostName Password* must exist in **/etc/ppp/pap-secrets** file for a successful connection. This option only has meaning for PAP authentication on both the authenticator and peer.
user *Name*	Defines the user entry to use for PAP authentication. An entry for *UserName RemoteHostName Password* must exist in **/etc/ppp/pap-secrets** file for a successful connection. This option only has meaning for PAP authentication on the peer.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
!0	An error occurred.

Security

Access Control: Any User

Auditing Events: N/A

Examples

1. You want System A to act as a client to server System B. From System A enter:

   ```
   /usr/sbin/pppattachd /dev/tty0 client ip connect "sysbconnector"
   ```

   where `sysbconnector` is the connector program.

   On System B, the user that logged in would have invoked from **$HOME/.profile**:

   ```
   exec /usr/sbin/pppattachd server ip 2>/dev/null
   ```

2. You want server System B to contact client System A. From System B enter:

   ```
   /usr/sbin/pppattachd /dev/tty0 server ipv6 connect "sysaconnector"
   ```

   where `sysaconnector` is the connector program.

   On System A, the user that logged in would have invoked from **$HOME/.profile**:

   ```
   exec /usr/sbin/pppattachd client ipv6 2>/dev/null
   ```

3. You want System A to act as a client to server System B using PAP authentication. System B acts as the authenticator and System A is the peer to be authenticated. From System A enter:

   ```
   /usr/sbin/pppattachd /dev/tty0 client ip ipv6 peer pap user username \
   connect "sysbconnector"
   ```

   where `sysbconnector` is the connector program.

   On System A, the **/etc/ppp/pap-secrets** file contains: username * ppppassword. On System B, the user that logged in would have invoked from **$HOME/.profile**:

   ```
   exec /usr/sbin/pppattachd server ip ipv6 authenticate pap 2>/dev/null
   ```

On System B, the **/etc/ppp/pap-secrets** file contains: username * ppppassword.

Files

Item	Description
/usr/sbin/pppattachd	Contains the **pppattachd** daemon.
/etc/ppp/att*XXX*.pid	Contains the process id. *XXX* is the pid, the content of the file is the network layer ID to which the attachment was bound. The user must belong to uucp group for the pid file to be created.

pppcontrold Daemon

Purpose

Controls startup and management of the PPP (Point to Point Protocol) subsystem.

Syntax

To Start and Stop by Using the System Resource Controller:

startsrc -s pppcontrold

stopsrc -s pppcontrold

Description

The **pppcontrold** daemon reads in the **/etc/ppp/lcp_config** and **/etc/ppp/if_conf** files to install and configure the PPP subsystem. SMIT should be used to generate both /etc/ppp/lcp_config and /etc/ppp/if_conf. To modify these files the user must have root authority or be a member of the uucp group. The configuration files are read at initialization where the appropriate streams modules are configured and loaded, and the tcpip network interface layers are installed into the system. After configuring the subsystem, the **pppcontrold** daemon monitors the streams associated with the IP and IPv6 interfaces to perform operations such as setting IP addresses, and the flags of the IP and IPv6 interface. The **pppcontrold** daemon terminates upon reciept of SIGTERM or when the **stopsrc** command is invoked. The prefered method of starting and stopping the **pppcontrold** daemon is with SRC (System Resource Controller). You must have root authority to run the src commands.

Errors and messages are logged using the **syslog** facility.

The **pppcontrold** daemon creates the **/etc/ppp/pppcontrold.pid** file, which contains a single line with the command process ID used to terminate the **pppcontrold** daemon.

Flags

None

/etc/ppp/lcp_config File

This file provides the configuration information required for the subsystem. These values are used to ensure proper allocation of storage at the time the subsystem is configured. It is important to configure just what is needed since these values define storage that is allocated within the kernel. Blank lines and lines beginning with a # (pound sign) are ignored in the configuration file. Do not use blank lines or lines beginning with # (pound sign) within the interface definition. Only use these lines between interface definitions.

Required Keywords	
server_name *name*	Name of this system. This name should be unique to the system. Ensure that the first 20 bytes of the name are unique.

Required Keywords

lcp_server #	Number of server connections. Represents the number of server connections that the subsystem will allow. Storage for all specified connections is allocated at the time the subsystem is configured. The minimum value is 0 and the maximum value is gated by the amount of memory in the system.
lcp_demand #	Specify the maximum number of demand links that you want the PPP LCP multiplexor to support. Set this value to the number of demand interfaces that you expect to configure. The default value is 0.
lcp_client #	Number of client connections. The minimum value is 0 and the maximum value is gated by the amount of memory in the system. Client connections are IP and IPv6 interfaces configured without addresses.
num_if #	Number of IP and IPv6 interfaces to configure. Must be less than or equal to lcp_server + lcp_client.
num_if6 #	Maximum number of TCP/IPv6 interfaces to allow. The value is a decimal number. This number, along with "max ip interfaces" and "max ip & ipv6 interfaces", cannot be greater than total maximum number of server, client and demand links (max server links + max client links + max demand links = max ip interfaces + max ipv6 interfaces + max ip & ipv6 interfaces). When a machine is used solely as a client connecting up to one server, this field would be set to 1. On a server, this field would be set to the maximum number of IPv6 clients that can simultaneously connect to the server. In this case, make sure that you have enough IPv6 interfaces defined.
num_if_and_if6 #	Maximum number of TCP/IP and IPv6 interfaces to allow. The value is a decimal number. This number along with "max ip interfaces" and "max ipv6 interfaces" cannot be greater than total maximum number of server, client and demand links (max server links + max client links + max demand links = max ip interfaces + max ipv6 interfaces + max ip & ipv6 interfaces). When a machine is used solely as a client connecting up to one server, this field would be set to 1. On a server, this field would be set to the maximum number of IP and IPv6 clients that can simultaneously connect to the server. In this case, make sure that you have enough IP and IPv6 interfaces defined.
num_hdlc #	Maximum number of concurrent asynchronous PPP sessions (server, client and demand) that can be active. This field is a decimal number. The value can not be greater than the total maximum number of server, client and demand links ([max server connections + max client connections + max demand connections] = max async hdlc attachments = [max ip interfaces + max ipv6 interfaces + max ip & ipv6 interfaces]).

Optional Keywords

These keywords will override the global default LCP options.

Item	Description
txacm *0xXXXXXXXX*	Transmit Asynchronous Character Map.
-negacm	Do not negotiate async character mapping. Rejects the peers configuration information frames that contains this option.

Item	Description
-negmru	Do not negotiate MRU (Maximum Receive Unit). Rejects the peers configuration information frames that contains this option.
mru #	MRU desired. A default is 1500.
-negacf	Do not negotiate ACF (address control field) compression. ACF will not be compressed. Rejects the peers configuration information frames that contain this option.
-negprotocolcompress	Do not negotiate protocol compression. Normally, the PPP protocol field will be compressed by one byte for Network protocols. This disables negotiation of this option for both receiving and sending frames.

/etc/ppp/if_conf File

This file defines all the server TCP/IP interfaces. Blank lines and lines beginning with a # (pound sign) are ignored in the configuration file. Do not use blank lines or lines beginning with # (pound sign) within the interface definition . Only use these lines between interface definitions.

Keywords

interface Indicates that a new interface definition is being started.

`ip` and `ipv6` Specifies the protocol or protocols used for this interface and will coincide with the local_ip, local_ip6, remote_ip, and remote_ip6 keywords. These keywords can be used alone or in combination.

server Indicates that the interface is a server connection.

Requires the following keywords:

local_ip xxx.yyy.zzz.qqq

remote_ip xxx.yyy.zzz.qqq

local_ip6 ::XXXX:XXXX:XXXX:XXXX

remote_ip6 ::XXXX:XXXX:XXXX:XXXX

These addresses MUST be different on the pair basis, however the local IP and IPv6 address can be the same for all PPP interfaces. On a given server, the remote address must be unique. The "interface" "server" entry will contain only local_ip and remote_ip addresses if the smitty PPP IP Interfaces menu is used to configure the interface. remote_ip6 and local_ip6 will be seen in the entry if the smitty PPP IPv6 Interfaces menu is used. Finally, all four will be found if smitty PPP IP and IPv6 Interfaces is used.

client This is an IPv6 option only. A client interface is required for all IPv6 connections. The address will be randomly generated based on the system model and ID. You can elect to zero out the address, (::0:0:0:0 or simply ::) and have the server assign an IPv6 address to the client. An example **if_conf** file entry follows:

```
interface
client
ipv6
local_ip6 ::0000:0000:0000:0000

interface
client
ip
ipv6
local_ip6 ::0007:0000:0000:4445
```

Keywords

demand There is a local_XXX and remote_XXX that are dependant on the protocol type (IP, IPv6 or both). A quoted command string is also required to establish connection with the authenticating host (server). An example **if_conf** file entry follows:

```
interface
demand
ipv6
local_ip6 ::0007:0000:0000:4444
remote_ip6 ::0009:0000:0000:5555
dcmd "exec /usr/sbin/pppattachd /dev/tty3 demand ipv6 >/dev/tty3 nodaemon"

interface
demand
ip
ipv6
local_ip 44.44.44.46
remote_ip 66.66.66.66
netmask 255.255.255.0
local_ip6 ::0007:0000:0000:4446
remote_ip6 ::0009:0000:0000:6666
dcmd "exec /usr/sbin/pppattachd /dev/tty4 demand ip ipv6 >/dev/tty4 nodaemon"
```

Optional Keywords

netmask xxx.xxx.xxx.xxx Specifies a netmask for an IPv4 interface.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
!0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Examples

Example **/ect/ppp/lcp_config** File:

```
# Comment line
server_name pppclient
lcp_server 0
lcp_client 3
lcp_demand 2
num_if 1
num_if6 2
num_if_and_if6 2
num_hdlc 5
```

Example **/ect/ppp/if_conf** File:

```
# Sample ip server configuration information.
# Note that the complete stanza does not contain
# comments or blank lines
interface
server
ip
local_ip 129.35.130.45
remote_ip 129.35.131.191
netmask 255.255.240.0

# Sample ipv6 server configuration information.
# Note that the complete stanza does not contain
```

```
# comments or blank lines
interface
server
ipv6
local_ip6 ::0009:2313:4C00:3193
remote_ip6 ::0009:2313:4C00:3194

#However between stanzas one can have blank or
# comment lines.

interface
server
ip
local_ip 129.35.130.45
remote_ip 129.35.131.196
netmask 255.255.240.0

interface
server
ip
local_ip 129.35.130.45
remote_ip 129.35.131.197
netmask 255.255.240.0

interface
server
ip
local_ip 129.35.130.45
remote_ip 129.35.131.201
netmask 255.255.240.0

interface
server
ip
local_ip 129.35.130.45
remote_ip 129.35.131.212
netmask 255.255.240.0
```

The above configuration files would result in a subsystem that installs the IP and IPv6 interfaces as follows:

```
pp0: flags=71<UP,POINTOPOINT,NOTRAILERS>
     inet 129.35.130.45 --> 129.35.131.191 netmask 0xfffff000
pp1: flags=31<UP,POINTOPOINT,NOTRAILERS>
     inet 129.35.130.45 --> 129.35.131.196 netmask 0xfffff000
pp2: flags=31<UP,POINTOPOINT,NOTRAILERS>
     inet 129.35.130.45 --> 129.35.131.197 netmask 0xfffff000
pp3: flags=31<UP,POINTOPOINT,NOTRAILERS>
     inet 129.35.130.45 --> 129.35.131.201 netmask 0xfffff000
pp4: flags=31<UP,POINTOPOINT,NOTRAILERS>
     inet 129.35.130.45 --> 129.35.131.212 netmask 0xfffff000
pp5: flags=30<POINTOPOINT,NOTRAILERS>
     inet netmask
```

Note: pp5 is the result of the lcp_client keyword in the /etc/ppp/lcp_config file (lcp_client 1). Both IP and IPv6 client interfaces will have no address associated with them until a connection" is established with the server and the IPs are negotiated through IPCP/IPV6CP. The only exception is with demand client interfaces. These interfaces will specify their own address and demand it during negotiation. As such, they will have their IP and IPv6 address associated with their interface as soon as the PPP subsystem is started.

Files

Item	Description
/usr/sbin/pppcontrold	Contains the **pppcontrold** daemon.
/etc/ppp/lcp_config	Configures the subsystem (**lcp_config** should be generated by SMIT).
/etc/ppp/if_conf	Configures the TCP/IP interfaces (**if_conf** should be generated by SMIT).
/etc/ppp/pppcontrold.pid	Contains the **pppcontrold** process id.
/etc/ppp/ppp.conf	Contains input to the **strload** command.

pppdial Command

Purpose

Establish an asynchronous connection with a remote system for use by the PPP (Point to Point Protocol) subsystem.

Syntax

pppdial [**-t** *TimeOut*] [**-v**] [**-d** *VerboseFile*] **-f** *ChatFile*

Description

The **pppdial** command provides the capability to establish a connection with a remote system over an asynchronous device. It is used with the **pppattachd** daemon as the means for carrying out the dialog with modems and remote systems to the point where PPP frames should be sent. The **pppdial** command uses standard input (stdin) and standard output (stdout) as the devices over which the dialog occurs.

Errors and messages are logged using the **syslog** facility.

Flags

Item	Description
-d *VerboseFile*	Logs the chat activity to *VerboseFile*. If *VerboseFile* does not exist, the **pppdial** command creates it. If *VerboseFile* does exists, the **pppdial** command appends the output to the existing file.
-f *ChatFile*	Specifies the file which contains the dialog that is to occur over the tty device. The content of *ChatFile* conforms to the syntax of the Basic Networking Utility (BNU)/UNIX to UNIX Copy Program (UUCP).
-t *TimeOut*	Specifies the number of seconds to wait before timing out during the **Expect** phase of the chat activity.
-v	Logs the chat activity using the syslog facility.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
!0	An error occurred.

Security

Access Control: Any User

Examples

To establish a connection with a remote system, enter on the command line in one line:

```
/usr/sbin/pppattachd client ip /dev/tty0 connect "/usr/sbin/pppdial
-v -f /home/pppuser/dialer.file"
```

The *ChatFile* named `/home/pppuser/dialer.file` contains:

```
''
atdt4311088
CONNECT
\\d\\n
ogin
pppuser
ssword
pppuserpwd
```

with the following meaning:

```
''            Expect a nul string
atdt4311088   Send the modem the dial command
              4311088 is the phone number to dial
CONNECT       Expect connect from the modem
\\d\\n        Delay for 1 second then send a new line
ogin          Expect the string ogin
pppuser       Send the string pppuser
              pppuser is the user id on the remote system
ssword        Expect the string ssword
pppuserpwd    Send the string pppuserpwd
              pppuserpwd is the password of the user pppuser on the
              remote system
```

The remote system must have a user `pppuser` defined with a password `pppuserpwd` and a **$HOME/.profile** containing:

```
exec pppattachd server ip ipv6 2>/dev/null
```

This is a very simplistic example. The example requires that the PPP subsystem is running on both the client and server (or remote) system. The example requires that the client system have a modem defined on `/dev/tty0`. The *ChatFile* contains the number 4311088 to dial. The remote system must also have a user defined with a password and a **.profile** which starts a PPP attachment on the remote system. The device (`/dev/tty0`), phone number, user, user password and mechanism starting the PPP attachment are variable and should reflect the current values on the server system.

Files

Item	Description
/usr/sbin/pppdial	Contains the **pppdial** command.

pppstat Command

Purpose

Extracts and displays the RAS (Reliability, Availability, and Serviceability) information of the PPP (Point to Point Protocol) subsystem.

Syntax

pppstat

Description

The **pppstat** command provides the capability to monitor particular characteristics of active links. The following information is displayed for all active links:

LCP Multiplexing Layer

Item	Description
Local MRU	Specifies the Maximum Receive Unit setting for this host. This is maximum length of a packet that the remote host can send to the local host.
Remote MRU	Specifies the Maximum Receive Unit setting for the remote host. This is the maximum length of a packet that we can send to the remote host.
Local To Peer ACCM	Specifies the ASYNC Character Map used in the transmission of packets to the remote host.
Peer To Local ACCM	Specifies the ASYNC Character Map used by the remote host in the transmission of packets to the local host.
Local To Remote Protocol Field Compression	Specifies whether Protocol Compression is used in the transmission of packets to the remote host.
Remote To Local Protocol Field Compression	Specifies whether Protocol Compression is used in the transmission of packets from the remote host to the local host.
Local To Remote Address/Control Field Compression	Specifies whether Address/Control field compression is being used in the transmission of packets to the remote host.
Remote To Local Address/Control Field Compression	Specifies whether Address/Control field compression is being used in the transmission of packets from the remote host to the local host.

LCP Multiplexing Layer prior to PPP negotiating

Item	Description
MRU	Specifies the Maximum Receive Unit for receiving packets. This is the value that this host attempted to negotiate with the remote host.
Receive ACCM	Specifies the initial remote-to-local ASYNC Character Map that was used in the negotiation.
Transmit ACCM	Specifies the initial local-to-remote ASYNC Character Map that was used in the negotiation.
Magic Number	Specifies the magic number attempted in negotiation.
Frame Check Size	Specifies the length of the Frame Check Sequence that this host attempted to negotiate. This is fixed at 16 bits.

HDLC Framing Layer

Item	Description
Bad Address Fields	Specifies the number of times a packet has been received with an incorrect address field.
Bad Controls Fields	Specifies the number of times a packet has been received with an incorrect control field.
Oversized Packets	Specifies the number of times a packet has been received that has a length that exceeds the Maximum Receive Unit length.
Bad Frame Check Sequence	Specifies the number of times a packet has been received with a bad Frame Check Sequence.

Item	Description
Incoming Good Octets	Specifies the number of octets received in valid packets.
Outgoing Good Octets	Specifies the number of octets sent successfully in packets.
Incoming Good Packets	Specifies the number of packets received successfully.
Outgoing Good Packets	Specifies the number of packets sent successfully.

The output is sent to **stdout**. Messages are sent to **stderr**.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: Any User

Auditing Events: N/A

Files

Item	Description
/usr/sbin/pppstat	Contains the **pppstat** command.

pprof Command

Purpose

Reports CPU usage of all kernel threads over a period of time.

Syntax

pprof { *time* | **-I** *pprof.flow* | **-i** *tracefile* | **-d** } [**-T** *bytes*] [**-v**] [**-s**] [**-n**] [**-f**] [**-p**] [**-w**] [**-r PURR**] [**-@** [*WparList* | **ALL**]

Description

The **pprof** command reports on all kernel threads running within an interval using the **trace** utility. The raw process information is saved to **pprof.flow**, and five reports are generated. The **pprof** command can also take previously generated **Pprof.flow** to regenerate reports. If no flags are specified, all reports are generated.

Types of Reports

Item	Description
pprof.cpu	Lists all kernel level threads sorted by actual cpu time. Contains: Process Name, Process ID, Parent Process ID, Process State at Beginning and End, Thread ID, Parent Thread ID, Actual CPU Time, Start Time, Stop Time, Stop - Start The WPAR name is also provided when the **-@** flag with no argument has been selected.
pprof.start	Lists all kernel threads sorted by start time. Contains: Process Name, Process ID, Parent Process ID, Process State Beginning and End, Thread ID, Parent Thread ID, Actual CPU Time, Start Time, Stop Time, Stop - Start The WPAR name is also provided when the **-@** flag with no argument has been selected.
pprof.namecpu	Lists information about each type of kernel thread (all executable with the same name). Contains: Process Name, Number of Threads, CPU Time, % of Total CPU Time The WPAR name is also provided when the **-@** flag with no argument has been selected.
pprof.famind	Lists all processes grouped by families (processes with a common ancestor). Child process names are indented with respect to the parent. Contains: Start Time, Stop Time, Actual CPU Time, Process ID, Parent Process ID, Thread ID, Parent Thread ID, Process State at Beginning and End, Level, Process Name. The WPAR name is also provided when the **-@** flag with no argument has been selected.
pprof.famcpu	Lists the information for all families (processes with a common ancestor). The Process Name and Process ID for the family is not necessarily the ancestor. Contains: Start Time, Process Name, Process ID, Number of Threads, Total CPU Time. The WPAR name is also provided when the **-@** flag with no argument has been selected.

Flags

Item	Description
-d	Waits for the user to execute **trcon** and **trcstop** from the command line.
-f	Specifies to only generate the **pprof.famcpu** and **pprof.famind** reports.
-i *tracefile*	Indicates to generate reports from a **tracefile**. The trace must contain the following hooks: 135,106,10C,134,139,465,467,00A
-I *pprof.flow*	Indicates to generate reports from a previously generated **pprof.flow**. Specifies to only generate the **pprof.namecpu** report.
-n	Specifies to only generate the **pprof.namecpu** report.
-p	Specifies to only generate the **pprof.cpu** report.
-r PURR	Uses PURR time instead of TimeBase in percent and CPU time calculation. Elapsed time calculations are unaffected.
-s	Specifies to only generate the **pprof.start** report.

Item	Description
-T	Sets the trace kernel buffer size in bytes. The default is 32000.
-v	Sets verbose mode (print extra details).
-w	Specifies to only generate **pprof.flow**.
-@ [*WparList* | **ALL**]	Displays WPAR information. **ALL** Lists all WPARs. ***WparList*** Specifies a comma-separated list of WPARs of interest.
time	Specifies the number of seconds to trace the system.

Note: Review the **/usr/lpp/perfagent/README.perfagent.tools** file for the latest on changes to the performance analysis tools.

pr Command

Purpose

Writes a file to standard output.

Syntax

pr [**+***Page*] [**-***Column* [**-a**] | **-m**] [**-d**] [**-F**] [**-r**] [**-t**] [**-e** [*Character*] [*Gap*]] [**-h** *Header*] [**-i** [*Character*] [*Gap*]] [**-l** *Lines*] [**-n** [*Character*] [*Width*]] [**-o** *Offset*] [**-s** [*Character*]] [**-w** *Width*] [**-x** [*Character*] [*Width*]] [**-f**] [**-p**] [*File* ... | **-**]

Description

The **pr** command writes the specified file or files to standard output. If you specify the **-** (minus sign) parameter instead of the *File* parameter, or if you specify neither, the **pr** command reads standard input. A heading that contains the page number, date, time, and name of the file separates the output into pages.

Unless specified, columns are of equal width and separated by at least one space. Lines that are too long for the page width are cut off. If standard output is a workstation, the **pr** command does not display error messages until it has ended.

Flags

Item	Description
-*Column*	Sets the number of columns to the value specified by the *Column* variable. The default value is 1. This option should not be used with the **-m** flag. The **-e** and **-i** flags are assumed for multicolumn output. A text column should never exceed the length of the page (see the **-l** flag). When the **-***Column* flag is used with the **-t** flag, use the minimum number of lines to write the output.
+*Page*	Begins the display with the page number specified by the *Page* variable. The default value is 1.
-a	Modifies the effect of the **-***Column* flag so that multiple columns are filled horizontally, from left to right. For example, if there are two columns, the first input line goes in column 1, the second goes in column 2, the third becomes line 2 of column 1, and so forth. If the **-a** flag is not specified, columns are created vertically.

Item	Description
-d	Produces double-spaced output.
-e[*Character*][*Gap*]	Expands tabs to character positions as follows: *Gap*+1, 2**Gap*+1, 3**Gap*+1, and so on. The default value of *Gap* is 8. Tab characters in the input expand to the appropriate number of spaces in order to line up with the next tab setting. If you specify a value for the *Character* variable (any character other than a digit), that character becomes the input tab character. The default value of the *Character* variable is the ASCII TAB character.
-F	Uses a form-feed character to advance to a new page. (Otherwise the **pr** command issues a sequence of line-feed characters.) Pauses before beginning the first page if the standard output is a workstation. This flag is equivalent to the **-f** flag.
-f	Uses a form-feed character to advance to a new page. (Otherwise the **pr** command issues a sequence of line-feed characters.) Pauses before beginning the first page if the standard output is a workstation. This flag is equivalent to the **-F** flag.
-h *Header*	Uses the specified header string as the page header. If the **-h** flag is not used, the page header defaults to the file name specified by the *File* parameter.
-i[*Character*][*Gap*]	Replaces white space wherever possible by inserting tabs to character positions, as follows: *Gap*+1, 2**Gap*+1, and 3**Gap*+1, and so forth. The default value of *Gap* is 8. If you specify a value for the *Character* variable (any character other than a digit), that character is used as the output tab character.
-l *Lines*	Overrides the 66-line default and resets the page length to the number of lines specified by the *Lines* variable. If the *Lines* value is smaller than the sum of both the header and trailer depths (in lines), the header and trailer are suppressed (as if the **-t** flag were in effect).
-m	Merges files. Standard output is formatted so the **pr** command writes one line from each file specified by the *File* parameter, side by side into text columns of equal fixed widths, based on the number of column positions. This flag should not be used with the **-** *Column* flag.
-n[*Character*][*Width*]	Provides line numbering based on the number of digits specified by the *Width* variable. The default is 5 digits. The line number occupies the first *Width*+1 column positions of each text column of default output, or of each line of output when the **-m** flag is set. If the *Character* variable is specified (any non-digit character), it is appended to the line number to separate it from what follows on the line. The default character separator is the tab character.
-o *Offset*	Indents each line by the number of character positions specified by the *Offset* variable. The total number of character positions per line is the sum of the width and offset. The default *Offset* value is 0.
-p	Pauses before beginning each page if the output is directed to a workstation. The **pr** command sounds the alarm at the workstation and waits for you to press the Enter key.
-r	Does not display diagnostic messages if the system cannot open files.
-s[*Character*]	Separates columns by the single character specified by the *Character* variable instead of by the appropriate number of spaces. The default value for the *Character* variable is an ASCII TAB character.

Item	Description
-t	Does not display the five-line identifying header and the five-line footer. Stops after the last line of each file without spacing to the end of the page.
-w *Width*	Sets the width of line to width column positions for multiple text-column output only. If the **-w** option is not specified and the **-s** option is not specified, the default width is 72. If the **-w** is not specified and the **-s** option is specified, the default width is 512. For single column output, input lines will not be truncated.
-x[*Character*][*Width*]	Provides the same line numbering functions as the **-n** flag.

Exit Status

This command returns the following exit values:

Item	Description
0	All files were successfully written.
>0	An error occurred.

Examples

1. To print a file with headings and page numbers on the printer, type:

   ```
   pr prog.c | qprt
   ```

   This adds page headings to the **prog.c** file and sends it to the **qprt** command. The heading consists of the date the file was last modified, the file name, and the page number.
2. To specify a title, type:

   ```
   pr  -h "MAIN PROGRAM" prog.c | qprt
   ```

   This prints the **prog.c** file with the title Main Program in place of the file name. The modification date and page number are still printed.
3. To print a file in multiple columns, type:

   ```
   pr -3 word.lst | qprt
   ```

   This prints the **word.lst** file in three vertical columns.
4. To print several files side by side on the paper:

   ```
   pr -m -h "Members and Visitors" member.lst visitor.lst | qprt
   ```

   This prints the **member.lst** and **visitor.lst** files side by side with the title Members and Visitors.
5. To modify a file for later use, type:

   ```
   pr -t -e prog.c > prog.notab.c
   ```

   This replaces tab characters in the **prog.c** file with spaces and puts the result in **prog.notab.c** file. Tab positions are at every eighth column (that is 9, 17, 25, 33, . . .). The **-e** flag tells the **pr** command to replace the tab characters; the **-t** flag suppresses the page headings.

Files

Item	Description
/usr/bin/pr	Contains the **pr** command.

Item	Description
/dev/tty*	Suspends messages.

praliases Command

Purpose

Displays mail aliases of the system.

Syntax

praliases [**-C** *file*] [**-f** *file*] [key]

Description

The **praliases** command displays current aliases of the system for each line, in no particular order. The special internal `@:@` alias is displayed, if present.

Flags

Item	Description
-C *file*	Reads the specified sendmail configuration file instead of the default sendmail configuration file.
-f *file*	Reads the specified file instead of the configured aliases files of the sendmail file.
key	Displays entries that match the keys, if one or more keys are specified on the command line.

Note: The **praliases** command exits with `0` on success and with `>0` if an error occurs.

Files

Item	Description
/etc/mail/ sendmail.cf	Contains the default sendmail configuration file.

prctmp Command

Purpose

Displays the session record files.

Syntax

/usr/sbin/acct/prctmp *File...*

Description

A user with administrative authority can enter the **prctmp** command to display the session record file created by the **acctcon1** command, normally the **/var/adm/acct/nite/ctmp** file. The session record file is converted into the connect-time total accounting record by the **acctcon2** command and then incorporated into the daily accounting report.

Security

Access Control: This command should grant execute (x) access only to members of the **adm** group.

Example

To display the session record file, enter:

```
prctmp /var/adm/acct/nite/ctmp
```

This command displays the session record file created by the **acctcon1** command.

Files

Item	Description
/usr/sbin/acct	The path to the accounting commands.
/var/adm/acct/nite	Contains accounting data files.

prdaily Command

Purpose

Creates an ASCII report of the previous day's accounting data.

Syntax

/usr/sbin/acct/prdaily [**-X**] [**-l**] [*mmdd*] [**-c**]

Description

The **prdaily** command is called by the **runacct** command to format an ASCII report of the previous day's accounting data. The report resides in the **/var/adm/acct/sum/rprt***mmdd* file, where *mmdd* specifies the month and day of the report.

Flags

Item	Description
-c	Reports exceptional resource usage by command. This flag may be used only on the current day's accounting data.
-l [*mmdd*]	Reports exceptional usage by login ID for the specified date. Use the *mmdd* variable to specify a date other than the current day.
-X	Processes all available characters for each user name instead of truncating to the first 8 characters. This flag will also cause the **prdaily** command to use the **/var/adm/acct/sumx** directory instead of the **/var/adm/acct/sum** directory.

Security

Access Control: This command should grant execute (x) access only to members of the **adm** group.

Files

Item	Description
/usr/sbin/acct	The path to the accounting commands.

Item	Description
/usr/sbin/acct/ptelus.awk	Calculates the limits for exceptional usage by login ID. This is a shell procedure.
/usr/sbin/acct/ptecms.awk	Calculates the limits of exceptional usage by command name. This is a shell procedure.
/var/adm/acct/sum	Cumulative directory for daily accounting records.
/var/adm/acct/sumx	Cumulative directory for daily accounting records when long user name processing is requested.

preparevsd Command

Purpose

Makes a virtual shared disk available.

Syntax

`preparevsd` {`-a` | *vsd_name*...}

Description

The **preparevsd** command brings the specified virtual shared disks from the stopped state to the suspended state. The virtual shared disks are made available. Open and close requests are honored, while read and write requests are held until the virtual shared disks are brought to the active state. If they are in the suspended state, this command leaves them in the suspended state.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Prepare a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Flags

-a
Specifies that all the virtual shared disks in the stopped state are to be prepared.

Parameters

vsd_name
Specifies a virtual shared disk. If the virtual shared disk is not in the stopped state, you will get an error message.

Security

You must have `root` authority to run this command.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use

the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to *RSCT Administration Guide* .

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Examples

To bring the virtual shared disk **vsd1vg1n1** from the stopped state to the suspended state, enter:

```
preparevsd vsd1vg1n1
```

Location

/opt/rsct/vsd/bin/preparevsd

preprpnode Command

Purpose

Prepares a node to be defined to a peer domain.

Syntax

`preprpnode` [`-k`] [`-h`] [`-TV`] *node_name1* [*node_name2* ...]

`preprpnode -f` | `-F` { *file_name* | `"-"` } [`-k`] [`-h`] [`-TV`]

Description

The `preprpnode` command prepares security on the node on which the command is run so it can be defined in a peer domain. It allows for peer domain operations to be performed on this node and must be run before the node can join a peer domain using the `mkrpdomain` or `addrpnode` command.

Before the `mkrpdomain` command is issued on a node, the `preprpnode` command must be run on each node to be defined to the new peer domain, using the name of the node that is to run the `mkrpdomain` command as the parameter. This gives the `mkrpdomain` node the necessary authority to create the peer domain configuration on each new node and set up additional security.

Before the `addrpnode` command is issued on a node, the `preprpnode` command must be run on each node that is to be added, using the names of all online nodes as the parameters. This gives the online nodes the authority to perform the necessary operations on the new node.

The `preprpnode` command performs the following:

1. Establishes trust with the node names specified on the command by adding their public keys to the trusted host list.
2. Modifies the resource monitoring and control (RMC) access control list (ACL) file to enable access to peer domain resources on this node from the other nodes in the peer domain. This allows peer domain operations to occur on the node. The RMC subsystem is refreshed so that these access changes will take effect.
3. RMC remote connections are enabled.

If the nodes that are to be defined to a peer domain are already in a management domain, you do not need to exchange public keys. You can use the `-k` flag to omit this step.

Flags

-f | -F { *file_name* | "–" }
Reads a list of node names from *file_name*. Each line of the file is scanned for one node name. The pound sign (#) indicates that the remainder of the line (or the entire line if the # is in column 1) is a comment.

Use `-f "-"` or `-F "-"` to specify STDIN as the input file.

-k
Specifies that the command should not exchange public keys.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

***node_name1* [*node_name2 ...*]**
Specifies the node (or nodes) from which peer domain commands can be accepted. Typically, this is the name of the node that will be running the `mkrpdomain` command when forming the peer domain. When adding to the peer domain, it is a list of the nodes that are currently online in the peer domain. The node name is the IP address or the long or short version of the DNS host name. The node name must resolve to an IP address.

Security

The user of the `preprpnode` command needs write permission to the access control list (ACL) file. Permissions are specified in the ACL file. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Restrictions

This command must run on a node that will be defined to the peer domain.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the `-f "-"` or `-F "-"` flag is specified, this command reads one or more node names from standard input.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. Suppose `mkrpdomain` will be issued from `nodeA`. To prepare `nodeB`, `nodeC`, and `nodeD` to be defined to a new peer domain, `ApplDomain`, run this command on `nodeB`, on `nodeC`, and then on `nodeD`:

   ```
   preprpnode nodeA
   ```

2. Suppose `nodeA` and `nodeB` are online in `ApplDomain`. To prepare `nodeC` to be added to the existing domain, run this command on `nodeC`:

   ```
   preprpnode nodeA nodeB
   ```

   Alternatively, create a file called `onlineNodes` with these contents:

   ```
   nodeA
   nodeB
   ```

   Then, run this command on `nodeC`:

   ```
   preprpnode -f onlineNodes
   ```

Location

/opt/rsct/bin/preprpnode

Files

The access control list (ACL) file — `/var/ct/cfg/ctrmc.acls` — is modified. If this file does not exist, it is created.

prev Command

Purpose

Shows the previous message.

Syntax

prev [**+***Folder*] [**-header** | **-noheader**] [**-showproc** *CommandString* | **-noshowproc**]

Description

The **prev** command displays the previous message in a folder. The **prev** command is similar to the **show** command with the **prev** value specified.

The **prev** command passes any flags that it does not recognize to the **showproc** program.

Flags

Item	Description
+*Folder*	Specifies the folder that contains the message you want to show.
-header	Displays a one-line description of the message being shown. The description includes the folder name and the message number. This flag is the default.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.
-noheader	Prevents display of a one-line description of each message.
-noshowproc	Uses the **/usr/bin/cat** command to list the previous command.
-showproc *CommandString*	Uses the specified command string to perform the listing.

Profile Entries

The following entries are part of the *UserMhDirectory*/**.mh_profile** file:

Item	Description
`Current-Folder:`	Sets the default current folder.
`Path:`	Specifies the *UserMhDirectory*.
`showproc:`	Specifies the program used to show messages.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the previous message in the current folder, enter:

   ```
   prev
   ```

   The system responds with a message similar to the following:

   ```
   (Message schedule: 10)
   ```

   The text of the message is also displayed. In this example, message `10` in the current folder `schedule` is the previous message.
2. To show the previous message in the `meetings` folder, enter:

   ```
   prev  +meetings
   ```

   The system responds with a message similar to the following:

   ```
   (Message inbox: 5)
   ```

   In this example, message 5 in the `meetings` folder is the previous message.

Files

Item	Description
$HOME/.mh_profile	Contains the MH user profile.
/usr/bin/prev	Contains the **prev** command.

printenv Command

Purpose

Displays the values of environment variables.

Syntax

printenv [*Name*]

Description

The **printenv** command displays the values of environment variables. If you specify the *Name* parameter, the system only prints the value associated with the *Name* parameter. If you do not specify the *Name* parameter, the **printenv** command displays the current environment, showing one *Name* **=***Value* sequence per line.

If you specify a *Name* parameter that you have not defined in the environment, the **printenv** command returns an exit status of 1; otherwise it returns a status of 0 (zero).

Examples

1. To find the current setting of the **MAILMSG** environment variable, enter:

   ```
   printenv MAILMSG
   ```

2. The command returns the value of the **MAILMSG** environment variable. For example:

   ```
   YOU HAVE NEW MAIL
   ```

printf Command

Purpose

Writes formatted output.

Syntax

printf *Format* [*Argument* ...]

Description

The **printf** command converts, formats, and writes its *Argument* parameters to standard output. The *Argument* parameters are formatted under control of the *Format* parameter. The formatted output line cannot exceed **LINE_MAX** bytes in length.

The following environment variables affect the execution of the **printf** command:

Item	Description
LANG	Determines the locale to use for the locale categories when both **LC_ALL** and the corresponding environment variable (beginning with **LC_**) do not specify a locale.
LC_ALL	Determines the locale to be used to override any values for locale categories specified by the setting of **LANG** or any other **LC_** environment variable.
LC_CTYPE	Determines the locale for the interpretation of sequences of bytes of text data as characters; for example, single versus multibyte characters in parameters.
LC_MESSAGES	Determines the language in which messages should be written.
LC_NUMERIC	Determines the locale for numeric formatting. This environment variable affects the format of numbers written using the **e, E, f, g,** and **G** conversion characters.

The *Format* parameter is a character string that contains three types of objects:

- Plain characters copied to the output stream.
- Conversion specifications, each of which cause 0 or more items to be retrieved from the value parameter list.
- The following escape sequences. When copied to the output stream, these sequences cause their associated action to be displayed on devices capable of the action:

Item	Description
****	Backslash
\a	Alert
\b	Backspace
\f	Form feed
\n	New line
\r	Carriage return
\t	Tab
\v	Vertical tab
*****ddd*	Where *ddd* is a one-, two-, or three-digit octal number. These escape sequences are displayed as a byte with the numeric value specified by the octal number.

The *Argument* parameter is a list of one or more strings to be written to standard output under the control of the *Format* parameter.

The *Format* parameter is reused as often as necessary to satisfy the *Argument* parameters. Any extra **c** or **s** conversion specifications are evaluated as if a null string *Argument* were supplied; other extra conversion specifications are evaluated as if a 0 *Argument* were supplied. Where the *Format* parameter contains no conversion specifications and *Argument* parameters are present, the results are unspecified.

Each conversion specification in the *Format* parameter has the following syntax in this order:

1. A **%** (percent sign).
2. Zero or more options, which modify the meaning of the conversion specification. The option characters and their meanings are:

Item	Description
-	The result of the conversion is left-aligned within the field.
+	The result of a signed conversion always begins with a sign (+ or -).

Item	Description
blank	If the first character of a signed conversion is not a sign, a blank is prefixed to the result. If both the blank and + option characters are displayed, then the blank option character is ignored.
#	This option specifies that the value is to be converted to an alternate form. For **c**, **d**, **i**, **u**, and **s** conversions, the option has no effect. For **o** conversion, it increases the precision to force the first digit of the result to be a, 0 (zero). For **x** and **X** conversions, a nonzero result has 0x, or 0X prefixed to it, respectively. For **e**, **E**, **f**, **g**, and **G** conversions, the result always contains a radix character, even if no digits follow the radix character. For **g** and **G** conversions, trailing zeros are not removed from the result as they usually are.
0	For **d**, **i**, **o**, **u**, **x**, **e**, **E**, **f**, **g**, and **G** conversions, leading zeroes (following any indication of sign or base) are used to pad to the field width, no space padding is performed. If the **0** (zero) and the **-** (minus sign) options are displayed, the **0** (zero) option is ignored. For **d**, **i**, **o**, **u**, **x**, and **X** conversions, if a precision is specified, the **0** (zero) option is ignored.

Note: For other conversions, the behavior is undefined.

3. An optional decimal digit string that specifies the minimum field width. If the converted value has fewer characters than the field width, the field is padded on the left to the length specified by the field width. If the left-adjustment option is specified, the field is padded on the right. If the result of a conversion is wider than the field width, the field is expanded to contain the converted result. No truncation occurs. However, a small precision may cause truncation on the right.
4. An optional precision. The precision is a . (dot) followed by a decimal digit string. If no precision is given, it is treated as 0 (zero). The precision specifies:
   - The minimum number of digits to be displayed for the **d**, **o**, **i**, **u**, **x**, or **X** conversions.
   - The number of digits to be displayed after the radix character for the **e** and **f** conversions.
   - The maximum number of significant digits for the **g** conversion.
   - The maximum number of bytes to be printed from a string in the **s** conversion.
5. A character that indicates the type of conversion to be applied, such as:

Item	Description
%	Performs no conversion. Prints a % (percent sign).
a, **A**	Accepts a floating-point value and converts the value to a decimal notation that is in the format `[-]0x`*h*`.`*hhhh*`p±`*d*. The decimal notation contains one hexadecimal digit before the decimal point. This hexadecimal digit must be a non-zero value if the specified floating-point value is a normalized floating-point value or is unspecified. The number of hexadecimal digits after the decimal point indicates the precision value. If the precision value is not specified by using the *Format* parameter and the exponent value of the FLT_RADIX argument is 2, the precision value represents the floating-point value. If the precision value is not specified and the exponent value of the FLT_RADIX argument is not 2, the precision value can distinguish between the different floating-point values in the internal representation format that is used by the **a**, **A** conversion specifier. The trailing zeros in the decimal notation can be removed. If the precision value is zero and the **#** flag is not specified, the decimal point is not displayed. The letters `abcdef` are used for **a** conversion specifier and the letters ABCDEF are used for **A** conversion specifier. The **A** conversion specifier provides a number with the characters X and P instead of the characters x and p. To represent the decimal exponent of 2, the exponent of the FLT_RADIX argument must contain one digit to as many digits as needed. If the floating-point value is zero, the exponent value is also zero. The floating-point value that represents an infinity or NaN data type is converted to the format of the **f**, **F** conversion specifier.

Item	Description
d, **i**	Accepts an integer value and converts it to signed decimal notation. The precision specifies the minimum number of digits to be displayed. If the value being converted can be represented in fewer digits, it is expanded with leading zeros. The default precision is 1. The result of converting a zero value with a precision of zero is a null string. Specifying a field width with a zero as a leading character causes the field width value to be padded with leading zeros.
o	Accepts an integer value and converts it to signed octal notation. The precision specifies the minimum number of digits to be displayed. If the value being converted can be represented in fewer digits, it is expanded with leading zeros. The default precision is 1. The result of converting a zero value with a precision of zero is a null string. Specifying a field width with a zero as a leading character causes the field width value to be padded with leading zeros. An octal value for field width is not implied.
u	Accepts an integer value and converts it to unsigned decimal notation. The precision specifies the minimum number of digits to be displayed. If the value being converted can be represented in fewer digits, it is expanded with leading zeros. The default precision is 1. The result of converting a zero value with a precision of zero is a null string. Specifying a field width with a zero as a leading character causes the field width value to be padded with leading zeros.
x, **X**	Accepts an integer value and converts it to hexadecimal notation. The letters abcdef are used for the **x** conversion and the letters ABCDEF are used for the **X** conversion. The precision specifies the minimum number of digits to be displayed. If the value being converted can be represented in fewer digits, it is expanded with leading zeros. The default precision is 1. The result of converting a zero value with a precision of zero is a null string. Specifying a field width with a zero as a leading character causes the field width value to be padded with leading zeros.
f, **F**	Accepts a floating-point value and converts it to a decimal notation in the format **[-]***ddd . ddd*. The number of digits after the radix character or the decimal point is equal to the specified precision value. The **LC_NUMERIC** locale category determines the radix character that must be used in this format. If a precision value is removed from the floating-point value, six digits are added in the output after the radix character. If the precision value is 0 (zero), the radix character is not displayed. A floating-point value that represents an infinity data type is converted to the format `[-]inf` or `[-]infinity`. The format `[-]inf` or `[-]infinity` is implementation-dependent. A floating-point value that represents a NaN data type is converted to the format `[-]nan` (*n-char-sequence*) or `[-]nan`. The meaning of any *n-char-sequence* is implementation-dependent. The **F** conversion specifier provides the formats `INF`, `INFINITY`, or `NAN` instead of `inf`, `infinity`, or `nan`.
e, **E**	Accepts a float or double value and converts it to the exponential form [**-**] *d***.***dd***e**{**+**|**-**}*dd*. There is one digit before the radix character (shown here as the decimal point) and the number of digits after the radix character is equal to the precision specification. The **LC_NUMERIC** locale category determines the radix character to use tin this format. If no precision is specified, then six digits are output. If the precision is 0 (zero), then no radix character will be displayed. The **E** conversion character produces a number with E instead of e before the exponent. The exponent always contains at least two digits. However, if the value to be printed requires an exponent greater than two digits, additional exponent digits are printed as necessary.
g, **G**	Accepts a float or double value and converts it in the style of the **f** or **e** conversion characters (or **E** in the case of the **G** conversion), with the precision specifying the number of significant digits. Trailing zeros are removed from the result. A radix character is displayed only if it is followed by a digit. The style used depends on the value converted. Style **g** results only if the exponent resulting from the conversion is less than -4, or if it is greater than or equal to the precision.
c	Accepts a value as a string and prints the first character in the string.

Item	Description
s	Accepts a value as a string and prints characters from the string until the end of the string is encountered or the number of characters indicated by the precision is reached. If no precision is specified, all characters up to the first null character are printed.
b	Accepts a value as a string, that may contain backslash-escape sequences. Bytes from the converted string are printed until the end of the string or number of bytes indicated by the precision specification is reached. If the precision is omitted, all bytes until the first null character are printed.

The following backslash-escape sequences are supported:

- The escape sequences previously listed above under the description of the *Format* parameter. These are converted to the individual characters they represented.
- The \c (backslash c) sequence, which is not displayed and causes the **printf** command to ignore any remaining characters in the string parameter containing it, any remaining string parameters, and any additional characters in the *Format* parameter.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. Enter the following command:

   ```
   printf "%5d%4d\n" 1 21 321 4321 54321
   ```

   This produces the following output:

   ```
       1  21
     3214321
   54321   0
   ```

   The *Format* parameter is used three times to print all of the given strings. The 0 (zero) is supplied by the **printf** command to satisfy the last %4d conversion specification.
2. Enter the following command:

   ```
   printf "%c %c\n" 78 79
   ```

   This produces the following output:

   ```
   7 7
   ```

3. The following example demonstrates how the **%$** format specifier can be used to print the date in an order different from the order of the arguments:

   ```
   printf (""%1$s, %3$d. %2$s, %4$d:%5$.2d", weekday, month, day, hour, min);
   Sunday, 3. July, 10:02
   (weekday, day. month, hour:min)
   ```

Files

Item	Description
/usr/bin/printf	Contains the **printf** command.

probevctrl Command

Purpose

Changes and displays the ProbeVue dynamic tracing parameters and the ProbeVue sessions.

Syntax

probevctrl [**-c** *attribute = value*] [**-C**] [**-f** { *milli*|*micro* }] [**-d** *sessionID*] [**-l**] [**-n** *attribute = value*] [**-p**] [**-s** { *probevue_session_id* }] [**-t**] [**-u** *user-list*] [**-T** { *show*|*start*|*stop*|*reset* }]

Description

The **probevctrl** command changes and displays the ProbeVue dynamic tracing parameters, the per-processor trace buffer size, the consumed pinned memory, the user owning the session, the identifier of the process that started the session, and the information on whether the session has kernel probes for the ProbeVue sessions.

The following ProbeVue parameters are configurable:

- ProbeVue status (enabled/disabled).
- Maximum pinned memory (MB) allocated for all ProbeVue sessions.
- Maximum pinned memory (KB) allocated for a non-privileged user's ProbeVue session including the memory for the trace buffers.
- Number of concurrent ProbeVue sessions allowed for a regular user.
- Default size of the per-processor trace buffers (KB).
- The minimum period in milliseconds that a regular user can request the trace consumer to read from its trace buffers.
- The default period in milliseconds that the ProbeVue buffers will be read by the trace consumer.
- The size of the per-processor computation stack used by a ProbeVue session (KB).
- The minimum time interval allowed for global root user in interval probes.
- The percentage of memory that is allocated for the dynamic data structure.
- The size of the per-processor local table in KB.
- The number of page fault contexts for handling page faults.
- The maximum number of threads a ProbeVue session should support when it has thread local variables.
- The maximum size of per-CPU buffer, in bytes, used by a **net** probe action.
- The maximum time, in milliseconds, a **systrace** probe action can take when the action is started in interrupt context.
- The maximum time, in milliseconds, a **sysproc** probe action can take when the action is started in interrupt context.
- The maximum time, in milliseconds, an **io** probe action can take when the action is started in interrupt context.
- The maximum time, in milliseconds, a **net** probe action can take when the action is started in interrupt context.
- The maximum time, in milliseconds, a **CPU-bound interval** probe action can take when the action is started in the interrupt context.

Only the root user or the users having the **aix.ras.probevue.manage** authorization can update the ProbeVue parameters and view all the ProbeVue sessions. Otherwise, users can view only the sessions owned by themselves. Each session is displayed in the following format:

Sid	Pid	Uid	Buffer size in bytes	Consumed memory in bytes	Kernel probes	Profiling
sid	*pid*	*uid*	*bufsize*	*memory*	yes or no	yes or no

By default, the ProbeVue is enabled. Attempt to disable the ProbeVue when the ProbeVue sessions are active will fail.

Flags

-c

Specifies non-user ProbeVue parameters. Arguments to this flag must be separated by commas or enclosed in double quotation marks and separated by commas or spaces. If either the **-p** or the **-t** flag is not specified with this flag, new values will be made effective both in the present boot and next boot sessions. The valid *attribute-value* pairs are as follows:

trace= {on | off}
Specifies whether the ProbeVue must be enabled or disabled.

default_buffer_size=*default_buffer_size*
Specifies the default size of the per-processor trace buffers in KB. This is rounded to the next 4KB page.

max_total_mem_size=*max_total_mem_size*
Specifies the maximum pinned memory in MB consumable by the entire ProbeVue framework.

default_read_rate=*default_read_rate*
Specifies the default period in milliseconds that the ProbeVue buffers will be read by the trace consumer.

stack_size=*stack_size_in_4Kpages*
Specifies the size of the per-processor computation stack in KB. This will be rounded to the next 4KB page.

local_table_size=*number*
Specifies the size of the per-processor local table in KB. Half of the space allocated for the local table is used by temporary strings. The default value is set to 4 KB.

min_interval=*interval in ms*
Specifies the minimum time interval allowed for global root user in interval probes.

num_pagefaults=*number*
Specifies the number of page fault contexts for handling page faults. The specified number of page fault contexts are preallocated during ProbeVue framework initialization.

num_threads_traced=*number*
Specifies the maximum number of threads a ProbeVue session can support when it has thread local variables. The ProbeVue framework preallocates all the thread-local variables at the start of a session for the maximum number of threads that are specified with this attribute.

max_net_buf_size=*number*
Specifies the maximum size of per-CPU buffer, in bytes, used by a **net** probe action.

max_intr_systrcprb_time=*number*
Specifies the maximum amount of time, in milliseconds, that a probe action can take to run when the **systrace** probe action is started in interrupt context.

max_intr_sysprocprb_time=*number*
Specifies the maximum amount of time, in milliseconds, that a probe action can take to run when the **sysproc** probe action is started in interrupt context.

max_intr_ioprb_time=*number*
Specifies the maximum amount of time, in milliseconds, that a probe action can take to run when the **io** probe action is started in interrupt context.

max_intr_netprb_time=*number*
Specifies the maximum amount of time, in milliseconds, that a probe action can take to run when the **net** probe action is started in interrupt context.

async_stats_fetch_interval=*number*
Specifies the asynchronous fetch interval in milliseconds to fetch the system statistics. This attribute is a global value that is applicable for all ProbeVue sessions. ProbeVue sets the asynchronous fetch interval based on this value. The default value for the tunable is 1000 milliseconds. Changes in the tunable value does not affect the running sessions and the specified value is used only for the new sessions.

fetch_stats_async_only={yes | no}
Specifies that all system statistics must be fetched in the asynchronous mode even if the synchronous fetch is possible. The default value for this parameter is no. Changes in the tunable value does not affect the running sessions and the specified value will be used only for the new sessions.

max_intr_cpuboundprb_time=*number*
Specifies the maximum amount of time, in milliseconds, that a probe action can take to run when the **CPU-bound interval** probe action is started in interrupt context.

-C
Sets the ProbeVue session tunable to initial values. The **-C** option uses the current configuration of system to determine configuration initial values (High Config or Low Config) to be set and updates both current and next boot parameters. For more information about high and low configuration values of tunables, see the **ProbeVue dynamic tracing facility** topic.

Note: Before using the **-C** flag, ensure there is no active ProbeVue session.

-d *sessionId*
Displays the list of probes enabled for the specified session. When you specify all as the session ID, then the probes for all the ProbeVue sessions that can be viewed by the user is displayed. A list of ProbeVue sessions and the associated session ID can be obtained using the **probevctrl** command.

-f
Specifies the format in which time consumed data for probe actions needs to be displayed. The **-f** option can be used along with the **-T** option and it shows action on a specific session. The possible formats follow:

milli
Displays the time in milliseconds.

micro
Displays the time in microseconds.

The default format is **milli**.

-l
Lists the present value of the ProbeVue configuration parameters. If the **-p** or the **-t** flag is not specified, parameter values for the present boot session are displayed.

-n
Specifies the configurable parameters for regular users. Arguments to this option must be separated by commas or enclosed in double quotation marks and separated by commas or spaces. If either the **-p** or the **-t** flag is not specified with this flag, new values will be made effective both in the present boot and next boot sessions. The valid *attribute-value* pairs are as follows:

max_mem_size=*max_mem_size*
Specifies the maximum pinned memory in MB consumable by a ProbeVue session.

max_sessions=*max_sessn*
Specifies the maximum concurrent sessions allowed.

min_read_rate=*min_read_rate*
Specifies the minimum period in milliseconds that a regular user can request the trace consumer to read from its trace buffers.

pin_mem_dvar_pc=*pin_mem_dvar_pc*
Specifies the percentage of memory that can be allocated to the dynamic data structure for dynamic type variables. This memory can be used for stack trace and associative array type dynamic variables. The value of this parameter is set in the range 10-100. The default value is 50.

-p
Specifies that the default values for the next boot must be updated and displayed.

-s *probevue_session_id*
Specifies the session on which the action is to be performed. This flag must be used with the **-T** flag.

-u
Specifies comma-separated user list whose ProbeVue sessions must be listed. If the **-u** flag is not specified, all of the ProbeVue sessions that the user can view are displayed. A user with the **aix.ras.probevue.manage** authorization can view all of the ProbeVue sessions in the system. Users without this authorization can view only the ProbeVue sessions they own.

-t
Specifies that the default values for the present boot session must be updated and displayed.

-T show | start | stop | reset
Specifies whether to show, start, stop, or reset the probe action profiling for the session that is specified by the **-s** flag. The **-s** flag must be used along with this flag.

Examples

1. To modify the next boot default buffer size and to turn on the dynamic tracing, enter:

   ```
   probevctrl -c trace=on,default_buffer_size=8 -p
   ```

   or

   ```
   probevctrl -c "trace=on default_buffer_size=8" -p
   ```

2. To list the next boot ProbeVue configuration, enter:

   ```
   probevctrl -l -p
   ```

3. To list the present ProbeVue configuration, enter:

   ```
   probevctrl -l -t
   ```

4. To list all of the ProbeVue sessions, enter:

   ```
   probevctrl
   ```

5. To list all of the ProbeVue sessions owned by the user guest, enter:

   ```
   probevctrl -u guest
   ```

6. To increase the percentage of pinned memory that is allocated for the dynamic data structures (stack trace and associative array) for the next boot from a default 50 -75, enter:

   ```
   probevctrl -n  pin_mem_dvar_pc = 75
   ```

probevue Command

Purpose

Starts a dynamic trace session. The command can preprocess the header file and exit without starting the dynamic trace session.

Syntax

probevue [**-c** "{ **timestamp** = { **0** | **1**| **2**} **thread** = { **on** | **off** } **tid** = { *t1*, ... } **pid** = { *p1*, ... } abs_mem_for_dvars=*memory_in_KB* num_threads_traced=*number_of_thread_to_trace*}"] [**-d**] [**-i** *Async_Fetch_Interval*] [**-e** *Pinned_memory_dvar_percent*] [**-f** *to_print_time_profile_data_milli_or_micro*] [**-I** *Include_file1, ...*] [**-K**] [**-o** *Output_file*] [**-q** info={none|normal|detail}] [**-s** *Buffer_size*][**-t** *Interval*][**-T**] [**-u**] [**-X** *Program_name* [**-A** "*Arguments_to_program*"] [**-L** *"Lib path"*] [**-g**]] [*Script_name* [*Arguments_to_script*]]

probevue [**-P** *C ++_header_file*]

probevue [**-l** "{ **syscall** | **syscallx** | **syscallx32** | **syscallx64** | **interval** | **systrace** | **sysproc** | **io** | **net**}"]

Description

The **probevue** command analyzes the operating system and user programs by dynamically enabling the user-specified probes, starting the actions that are associated with the probes when they are triggered, and presenting the captured trace data.

When you specify the **probevue** command with a vue script, the command enables the tracing that was specified in the script, and produces the tracing output.

When the **-P** option is specified with the C++ header file, the command produces the preprocessed encrypted C header file. The encrypted C header file can be further used to probe C++ application by using the **-I** option of the **probevue** command.

The arguments to the **probevue** command and the vue script can be specified in a script instead of the command line. The script can be run repeatedly with same arguments by using this script. A vue language construct #VUE_CMD_ARGS can be used to specify the arguments to the **probevue** command in the file and the #VUE_SCRIPT_ARGS language construct can be used to specify the arguments to the vue script.

Notes:

- When the arguments are specified in the script, all the arguments for a vue construct must be on the same line.
- If arguments to the **probevue** command are specified by command line and in the script, only command line arguments are considered and all command arguments in the script are ignored. This process applies to the vue script arguments also.
- The #VUE_CMD_ARGS and #VUE_SCRIPT_ARGS constructs do not support standard input.
- The dynamic memory requirement of the `probevue` command is proportional to the product of number of CPUs and per-CPU trace buffer size (the value of the **`default_buffer_size`** tunable parameter of the `probevctrl` command). Hence, in a system that has large number of CPUs and higher value of per-CPU buffer can cause the `probevue` command to exceed the memory limit set by the `ulimit` parameter. In such scenarios, run the `probevue` command with the **`-u`** flag to cross that limit.

Flags

Item	Description
-A "*Arguments_to_program*"	Specifies the arguments to the program that you specified to using the **-X** flag. If there are multiple arguments to the application, enclose each argument in quotation marks.

Item	Description
-g	Turn on grouping of output from every clause of the probevue session i.e. all the output from a clause will appear together without getting interleaved by output from any other clause that is running simultaneously in another CPU. The **-g** flag overrides the `group_output_start()` or the `group_output_end()` statements present in the VUE script. Output of every clause is grouped from clause start to clause end irrespective of the `group_output_start()` or the `group_output_end()` statements present in the clauses.
-c	Specifies how the trace data needs to be formatted. You must enclose arguments to this option in quotation marks and separate each argument by spaces. The options are as follows: **timestamp={0|1|2}** Controls the reporting of the time stamp that is associated with an event in the trace report. Specify one of the following values: **0** Displays the timestamp, in seconds and microseconds, for each message relative to the beginning of the trace. The first line of the trace output shows the base time from which the individual time stamps are measured. **1** With each message, displays the actual time taken to create the message. **2** With each message, displays the actual time taken to create the message as per the *printf's %A* format. **Note:** If both options are desired then 0,1 must be entered. That is, there must be no spaces between 0,1. **thread={on|off}** Displays the thread ID which generated the message, with each message. The default value is **off**. **pid={*p1*,..}** Displays only the messages that were generated by the processes specified. **Note:** If the thread has died before the trace consumer tries to know the process to which the thread belongs, or if the process that you specified no longer exists, the consumer cannot display the messages that were generated by the threads in this process, when you filter the messages by the process ID. **tid={*t1*,..}** Displays only the messages that were generated by the threads that you specified. **abs_mem_for_dvars=*memory_in_KB*** Specifies the pinned memory, in kilo bytes (KB), that is allocated for dynamic type variables. This option is mutually exclusive to the **-e** option. **num_threads_traced=*number_of_thread_to_trace*** Specifies the maximum number of threads that the current ProbeVue session can support when the session has thread-local variables. This value overrides the corresponding global ProbeVue tunable value.
-d	Displays the list of probes enabled for the session.

Item	Description
-e *Pinned_memory_dvar_percent*	Specifies the percentage of the dynamic data structure memory allocated for dynamic type variables. A minimum of 10 and a maximum of 100 value can be specified as the percentage.
-f	Specifies the format in which the time taken by probe action must be displayed. The supported formats follow: **milli** Displays time in milliseconds. **micro** Display time in microseconds. The default format is **milli**.
-i *Async_Fetch_Interval*	Specifies how often the asynchronous statistics is fetched for the **probevue** command. This option overrides the global interval time for the **probevue** command. The minimum interval is 100 milliseconds.
-I *Include_file1*	Uses the file specified as a post-processed header file, that is one with no C-preprocessor operators. It can be passed through the command line to be included when compiling the vue script.
-K	Enables RAS events related functionality in a probeVue session.
-l	Lists all the probe points supported by the probe manager. When you specify the **-l** flag with the **probevue** command, no other flags must be used. You can specify more than one probe manager with the **-l** flag, such as `-l syscall -l syscallx -l interval`. The probe manager supports **interval**, **syscall**, **syscallx**, **systrace**, **io**, **sysproc**, and **net** probes for the **-l** flag. If you specify wrong arguments or an incorrect probe manager with the **-l** option, a usage error is displayed. • `probevue -l syscall`: Lists all the possible system call that can be traced on the system. • `probevue -l syscallx`: Displays all base system calls that can be traced on the system. This option lists the system call separately for the 32 and 64-bit systems. • `probevue -l syscallx32`: Displays the 32-bit base system calls that can be traced on the system. • `probevue -l syscallx64`: Displays the 64-bit base system calls that can be traced on the system. • `probevue -l interval`: Specifies the minimum and maximum interval duration supported for regular and root users with the **interval** probe. • `probevue -l systrace`: Displays a description about the systrace probe. **Note:** For syscallx probe manager, when the **-l syscallx** probe is used, it displays both 32-bit and 64-bit base-system calls. To view only 32-bit calls, use **syscallx32** probes, and to view only 64-bit calls, use **syscallx64** probes.

Item	Description
-L "*Lib path*"	Prioritizes library search in the specified path. If the libraries are not found in the specified path, searches libraries in the default library path that is saved in the header of the `loader` section in the eXtended COFF (XCOFF) file. The **-L** flag is applicable to the user function tracing (uft) and C++ probe managers that use the name of the executable file. The **-L** flag is ignored for other probe managers.
-o *Output_file*	Writes the report to a file rather than to the standard output.
-P C++ *header file*	Preprocesses the **C++** header file and creates an output preprocessed file for each input **C++** header file. The preprocessed output file has the same name as the input **C++** header file, with a .Vue suffix. **Note:** You cannot use other flags with the **-P** option. The **-P** flag accepts any file name, except the file name with a .Vue suffix.
-q *info=level*	Specifies reporting level of informational messages while parsing the vue script. The possible values follow: **none** No informational messages are displayed. **normal** Only important informational messages are displayed. **detail** All informational messages that can result in incorrect execution are reported.
-s *Buffer_size*	Specifies the size of the per-CPU trace buffers in KB. This is rounded to the next 4K page. You can use following levels:
-t *Interval*	Specifies how often the trace buffers are read. The minimum interval that you can specify is 10 milliseconds. The time interval specified by the regular user (that is a user without the **aix.ras.probevue.trace** privilege) is rounded to the next highest multiple of 10 milliseconds. The read rate is retrieved from the **probeVue** configuration. **Note:** A regular user can specify the minimum read rate and the **probevctrl** command can change the default read rate.
-T	Starts probe action profiling at the start of session. The **-T** flag ensures that probe actions are profiled when the session is started.
-u	Starts probeVue session and sets the data segment to unlimited value. The **-u** flag is used to print stack traces from multiple processes of large binaries.
-X *Program_name*	Starts a program and enables probes before the program starts. You can use the special environment variables **$__CPID** and **$__CTID** within a vue script to identify the process ID and the thread ID of the application that is launched.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To start a **probeVue** session with script **syscall.e**, enter:

```
probevue syscall.e
```

2. To send the trace report to the **/tmp/trace_report** file, enter:

```
probevue -o /tmp/trace_report syscall.e
```

3. To display the trace report of the thread IDs 12345,4567 and the timestamp relative to the beginning of trace, enter:

```
probevue -c "timestamp=0 tid=12345,4567" syscall.e
```

4. To include the header file **stat.i** and allocate 4K of per-CPU buffer, enter:

```
probevue –I stat.i –s 4 syscall.e
```

5. To preprocess the C++ header file **myheader.h** , enter:

```
  probevue –P myheader.h
```

   The **probevue** command generates the **myheader.Vue** file, which is an encrypted **C++** header file and is included in the trace session by using the **-I** option.

6. To increase the percentage of pinned memory for the current session of the dynamic data structures (stack trace and associative array), from a default of 50 -75 for the **ASO.e** script, enter:

```
probevue -e 75  ASO.e
```

7. The following script is an example of providing the arguments in the script:

```
!#/usr/bin/probevue
    #VUE_CMD_ARGS=-o /tmp/trace_out
    #VUE_SCRIPT_ARGS=read

    @@syscall:*:$1:entry
    {
        printf("%t\n", get_stktrace(4));
    }
```

   The script runs as: `./script.e`

Files

Item	Description
/usr/bin/ probevue	Contains the **probevue** command.

proccred Command

Purpose

Prints the credentials (effective, real, saved user IDs and group IDs) of processes.

Syntax

proccred *ProcessID* ...

Description

The **/proc** filesystem provides a mechanism to control processes. It also gives access to information about the current state of processes and threads, but in binary form. The proctools commands provide ascii reports based on some of the available information.

Most of the commands take a list of process IDs or **/proc/***ProcessID* strings as input. The shell expansion **/proc/*** can therefore be used to specify all processes in the system.

Each of the proctools commands gathers information from **/proc** for the specified processes and displays it to the user. The proctools commands like procrun and procstop start and stop a process using the **/proc** interface.

The information gathered by the commands from **/proc** is a snapshot of the current state of processes, and therefore can vary at any instant except for stopped processes.

The **proccred** command prints the credentials (effective, real, saved user IDs and group IDs) of processes.

Flags

Item	Description
ProcessID	Specifies the process id.

Examples

1. To display the credentials of process 5046, enter:

```
proccred  5046
```

Files

Item	Description
/proc	Contains the **/proc** filesystem.

procfiles Command

Purpose

Reports information about all file descriptors opened by processes.

Syntax

procfiles [**-F**] [**-n**] [**-c**] *ProcessID* ...

Description

The **/proc** filesystem provides a mechanism to control processes. It also gives access to information about the current state of processes and threads, but in binary form. The **proctools** commands provide ASCIIi reports based on some of the available information.

Most of the commands take a list of process IDs or **/proc/***ProcessID* strings as input. The shell expansion **/proc/*** can therefore be used to specify all processes in the system.

Each of the proctools commands gathers information from **/proc** for the specified processes and displays it to the user. The **proctools** commands like procrun and procstop start and stop a process using the **/proc** interface.

The information gathered by the commands from **/proc** is a snapshot of the current state of processes, and therefore can vary at any instant except for stopped processes.

Regular files have permission based on mode it was opened with. Any non-regular files have 0 access mode.

The **procfiles** command reports information on all file descriptors opened by processes. With the **-n** option it also displays the names of the corresponding files.

Flags

Item	Description
-c	Prints the output in column format.
-F	Forces procfiles to take control of the target process even if another process has control.
-n	Prints the names of the files referred to by file descriptors.
ProcessID	Specifies the process id.

Examples

1. To display status and control information on the file descriptors opened by process 11928, enter the following command:

   ```
   procfiles 11928
   ```

   The output of this command might look like this:

   ```
   11928 : -sh
     Current rlimit: 2000 file descriptors
      0: S_IFCHR mode:0622 dev:10,4  ino:2584 uid:100 gid:100 rdev:28,1
         O_RDONLY
      1: S_IFCHR mode:0622 dev:10,4  ino:2584 uid:100 gid:100 rdev:28,1
         O_RDONLY
      2: S_IFCHR mode:0622 dev:10,4  ino:2584 uid:100 gid:100 rdev:28,1
         O_RDONLY
      63: S_IFREG mode:0600 dev:10,8  ino:311 uid:100 gid:100 rdev:40960,10317
         O_RDONLY size:2574
   ```

2. To display name, status and control information on the file descriptors opened by process 15502, enter the following command:

   ```
   procfiles -n 15502
   ```

   The output of this command might look like this:

   ```
   15502 : /home/guest/test
     Current rlimit: 2000 file descriptors
      0: S_IFCHR mode:0622 dev:10,4  ino:2584 uid:100 gid:100 rdev:28,1
         O_RDONLY
      1: S_IFCHR mode:0622 dev:10,4  ino:2584 uid:100 gid:100 rdev:28,1
         O_RDONLY
      2: S_IFCHR mode:0622 dev:10,4  ino:2584 uid:100 gid:100 rdev:28,1
         O_RDONLY
      3: S_IFREG mode:0644 dev:10,7  ino:26 uid:100 gid:100 rdev:0,0
         O_RDONLY size:0  name:/tmp/foo
   ```

3. To display status and control information on the file descriptors opened by the 278684 process, enter the following command:

   ```
   procfiles -c 278684
   ```

   The output of this command might look like this:

```
278684 : -ksh
  Current rlimit: 2000 file descriptors
------------------------------------------------------------------------------
 FD   TYPE   MODE      DEV/RDEV     UID       GID          OPMOD           INODE
------------------------------------------------------------------------------
 0     c   ---------  10, 4(19, 0)  root      system        R-W            16385
 1     c   ---------  10, 4(19, 0)  root      system        R-W            16385
 2     c   ---------  10, 4(19, 0)  root      system        R-W            16385
 61    -   rw-r--r--  10, 7         root      system        R-W            32
 63    -   rw-------  10, 4         root      system        R-W | A        1051
```

Files

Item	Description
/proc	Contains the **/proc** filesystem.

procflags Command

Purpose

Prints the **/proc** tracing flags, the **pending** and **held** signals, and other **/proc** status information for each thread in the specified processes.

Syntax

procflags [**-r**] *ProcessID* ...

Description

The **/proc** filesystem provides a mechanism to control processes. It also gives access to information about the current state of processes and threads, but in binary form. The proctools commands provide ascii reports based on some of the available information.

Most of the commands take a list of process IDs or **/proc/***ProcessID* strings as input. The shell expansion **/proc/*** can therefore be used to specify all processes in the system.

Each of the proctools commands gathers information from **/proc** for the specified processes and displays it to the user. The proctools commands like procrun and procstop start and stop a process using the **/proc** interface.

The information gathered by the commands from **/proc** is a snapshot of the current state of processes, and therefore can vary at any instant except for stopped processes.

The **procflags** command prints the **/proc** tracing flags, the pending and held signals, and other **/proc** status information for each thread in the specified processes. The machine register contents are printed when option **-r** is used and the process is stopped on an event of interest. The events of interest are **PR_REQUESTED**, **PR_FAULTED**, **PR_SYSENTRY**, and **PR_SYSEXIT** as defined in **<sys/procfs.h>**.

Flags

Item	Description
-r	Displays the current machine registers state if a process is stopped in an event of interest.
ProcessID	Specifies the process id.

Examples

1. To display the tracing flags of process 5046, enter:

```
procflags  5046
```

The output of this command might look like this:

```
5046 : -sh
data model = _ILP32 flags = PR_FORK
/4289: flags = PR_ASLEEP | PR_NOREGS
```

2. To display the tracing flags and registers values of process 5040 which was stopped on an event of interest, enter:

```
procflags -r 5040
```

The output of this command might look like this:

```
5040 : ls
data model = _ILP32 flags = PR_FORK
/6999: flags = PR_STOPPED | PR_ISTOP
why = PR_FAULTED  what = FLTBPT what = kfork
gpr0  = 0x0               gpr1  = 0x2ff227b0        gpr2  = 0xf0083bec
gpr3  = 0x2ff22cb3        gpr4  = 0x11              gpr5  = 0x65
gpr6  = 0x50              gpr7  = 0x0               gpr8  = 0x41707a7c
gpr9  = 0x4c4f47          gpr10 = 0x80000000        gpr11 = 0x34e0
gpr12 = 0x0               gpr13 = 0xdeadbeef        gpr14 = 0x1
gpr15 = 0x2ff22c0c        gpr16 = 0x2ff22c14        gpr17 = 0x0
gpr18 = 0xdeadbeef        gpr19 = 0xdeadbeef        gpr20 = 0xdeadbeef
gpr21 = 0xdeadbeef        gpr22 = 0x10              gpr23 = 0xfd
gpr24 = 0x2f              gpr25 = 0x2ff227f0        gpr26 = 0x0
gpr27 = 0x2ff22d87        gpr28 = 0x2ff22cb3        gpr29 = 0x0
gpr30 = 0x0               gpr31 = 0xf0048260        iar = 0xd01be900
msr = 0x2d032             cr = 0x28222442           lr = 0xd01d9de0
ctr = 0xec                xer = 0x0                 fpscr = 0x0
fpscrx = 0x0
```

Files

Item	Description
/proc	Contains the **/proc** filesystem.

procldd Command

Purpose

Lists the objects loaded by processes, including shared objects explicitly attached using **dlopen()**.

Syntax

procldd [**-F**] *ProcessID* ...

Description

The **/proc** filesystem provides a mechanism to control processes. It also gives access to information about the current state of processes and threads, but in binary form. The **proctools** commands provide ASCII reports based on some of the available information.

Most of the commands take a list of process IDs or **/proc/***ProcessID* strings as input. The shell expansion **/proc/*** can therefore be used to specify all processes in the system.

Each of the proctools commands gathers information from **/proc** for the specified processes and displays it to the user. The **proctools** commands like procrun and procstop start and stop a process using the **/proc** interface.

The information gathered by the commands from **/proc** is a snapshot of the current state of processes, and therefore can vary at any instant except for stopped processes.

The **procldd** command lists the objects loaded by processes, including shared objects explicitly attached using **dlopen()**. All the information needed is gathered from the **/proc/***ProcessID***/map** files.

Flags

Item	Description
-F	Forces procldd to take control of the target process even if another process has control.
ProcessID	Specifies the process id.

Examples

1. To display the list of objects loaded by process 12644, enter:

```
procldd 12644
```

The output of this command might look like this:

```
12644 : -ksh
ksh
/usr/lib/libiconv.a[shr4.o]
/usr/lib/libi18n.a[shr.o]
/usr/lib/nls/loc/en_US
/usr/lib/libcrypt.a[shr.o]
/usr/lib/libc.a[shr.o]
```

Files

Item	Description
/proc	Contains the **/proc** filesystem.

procmap Command

Purpose

Prints the address space map of processes.

Syntax

procmap [**-F**] [**-S**] { **-X** [**-f**] [**-n**] [**-u**] [**-q**]}] *ProcessID* ...

Description

The **/proc** file system provides a mechanism to control processes. It also gives access to information about the current state of processes and threads, but in binary form. The proctools commands provide *ascii* reports based on some of the available information.

Most of the commands take a list of process IDs or **/proc/***ProcessID* strings as input. The shell expansion **/proc/*** can therefore be used to specify all processes in the system.

Each of the proctools commands gathers information from **/proc** for the specified processes and displays it to the user. The proctools commands like the **procrun** and **procstop**start and stop a process using the **/proc** interface.

The information gathered by the commands from **/proc** is a snapshot of the current state of processes, and therefore can vary at any instant except for stopped processes.

The **procmap** command prints the address space map of processes. It displays the starting address and size of each of the mapped segments in the process. It gets all the information necessary from the **/proc/***ProcessID***/map** files.

Flags

Item	Description
-F	Forces the **procmap** command to take control of the target process even if another process has control.
-S	Displays shared memory information of the target process. **Note:** The **-S** option is obsolete. You can use the **-X** option for similar functionality.
-X	Displays extended information about each address range and includes all mapped objects of the target process.
-n	Displays unsorted address ranges for the mapped segments of the target process.
-u	Displays the values in the most appropriate unit. The unit used in the report is specified for each metric.
-q	Suppresses the header information.
-f	Displays the filename instead of device number and inode number for the mapped file of the target process. The *MAPPED OBJECT* column for the mapped files displays one of the following values: • `dev:remote` for the remote files. • `dev:remote` from within a WPAR using global host storage. • `<full pathname>` from within a WPAR using direct storage. • `dev:<major#>, <minor#>, ino:<inode#>` outside a WPAR using direct storage. • For the non-WPAR local files, the file names is resolved, or the string `dev<major#>, <minor#>, ino:<inode#>` is displayed with the correct major, minor, and inode numbers for the file.
ProcessID	Specifies the process id.

The following are brief descriptions of the contents of the columns with -X flag:

Item	Description
`Start-ADD`	Start address of the memory region.
`End-ADD`	End address of the memory region.
`SIZE`	Size of the memory region.
`MODE`	Read, write and execute permission of the memory region.
`PSIZ`	Page size of the memory region. The following values are displayed:
`s`	4K page size
`m`	64K page size
`L`	16M page size
`H`	16GB page size
`TYPE`	Type of the memory region. The following values are displayed:

Item	Description
HEAP	Heap Region
KER/LDR	Other Kernel/Loader Segments
KERTXT	Kernel Text
MAINDATA	Main Data/BSS for the process
MAINTEXT	Main Executable for the process
MFILE	Map File
PLIBDATA	Private Library Data
PLIBTEXT	Private Library Text
PMMAP	Private Memory Map
PMMFILE	Private Memory Map file
PMMPSXRT	Private POSIX Real Time Shared Memory Map
REALMEM	Real Memory Mapped
SHM	Shared Memory
SHMEXT	Extended Shared Memory
SHMFILE	Shared Memory File
SLIBDATA	Shared Library Data
SLIBTEXT	Shared Library Text
SMMAP	Shared Memory Map
SMMFILE	Shared Memory Map file
SMMPSXRT	Shared POSIX Real Time Shared Memory
STACK	Stack Region
VSID	Virtual Segment ID of the memory region.
MAPPED OBJECT	Mapped object name.

Examples

1. To display the address space of process 12644, enter:

```
procmap 12644
```

   The output of this command might look like this:

```
12644 : -ksh
10000000         232K  read/exec         ksh
20000ef8          54K  read/write        ksh
d008b100          80K  read/exec         /usr/lib/libiconv.a[shr4.0]
f03e4c70          41K  read/write        /usr/lib/libiconv.a[shr4.o]
d0080100          40K  read/exec         /usr/lib/libi18n.a[shr.o]
f03f0b78           4K  read/write        /usr/lib/libi18n.a[shr.o]
d007a000          11K  read/exec         /usr/lib/nls/loc/en_US
d007d130           8K  read/write        /usr/lib/nls/loc/en_US
d00790f8           2K  read/exec         /usr/lib/libcrypt.a[shr.o]
f03e3508           0K  read/write        /usr/lib/libcrypt.a[shr.o]
d02156c0        2282K  read/exec         /usr/lib/libc.a[shr.o]
f03474e0         621K  read/write        /usr/lib/libc.a[shr.o]
   Total        3380K
```

2. To display the address ranges of process with PID 1573580, enter:

```
# procmap -X 1573580
```

```
1573580 : ./self_mod_32

Start-ADD       End-ADD            SIZE MODE  PSIZ  TYPE       VSID          MAPPED OBJECT
0               10000000         262144K r--   m     KERTXT     20002
10000000        10001d33              7K rwx   sm    MAINTEXT   1730DF3       self_mod_32
200001d0        200007fc              1K rw-   sm    MAINDATA   4F0CCF        self_mod_32
200007fc        20011000             66K rw-   sm    HEAP       4F0CCF
20011000        20011685              1K rwx   sm    PLIBTEXT   4F0CCF        ./libself_priv.a[dl_prv32.o]
20011685        2001234c              3K rw-   sm    HEAP       4F0CCF
2001234c        200125c4              0K rw-   sm    PLIBDATA   4F0CCF        ./libself_priv.a[dl_prv32.o]
200125c4        20013000              2K rw-   sm    HEAP       4F0CCF
20013000        2ff23000         261184K rw-   sm    STACK      4F0CCF
30000000        30001000              4K rw-   sm    SMMFILE    8C0C0C        dev:10,7 ino:35
30001000        30002000              4K r--   sm    SMMFILE    8C0C0C        dev:10,7 ino:36
30002000        30003000              4K rw-   sm    PMMFILE    8C0C0C        dev:10,7 ino:35
30003000        30004000              4K r--   sm    PMMFILE    8C0C0C        dev:10,7 ino:36
30004000        30005000              4K rw-   sm    SMMPSXRT   8C0C0C        POSIX RT SHM 1
30005000        30006000              4K r--   sm    SMMPSXRT   8C0C0C        POSIX RT SHM 1
30006000        30007000              4K rw-   sm    PMMPSXRT   8C0C0C        POSIX RT SHM 2
30007000        30008000              4K r--   sm    PMMPSXRT   8C0C0C        POSIX RT SHM 2
30008000        30009000              4K rw-   sm    SMMAP      8C0C0C
30009000        3000a000              4K r--   sm    SMMAP      8C0C0C
3000a000        3000b000              4K rw-   sm    PMMAP      8C0C0C
3000b000        3000c000              4K r--   sm    PMMAP      8C0C0C
40000000        40000000              0K rw-   s     SHMFILE    250CA5        dev:10,7 ino:35
50000000        50000000              0K r--   s     SHMFILE    1960D16       dev:10,7 ino:36
60000000        60001000              4K rw-   sm    SHM        D0C8D         shmid:16
70000000        70001000              4K r--   sm    SHM        11F0D9F       shmid:17
d0100100        d052343c           4236K rwx   m     SLIBTEXT   3010B81       /usr/lib/libc.a[shr.o]
d0564100        d0564abe              2K rwx   m     SLIBTEXT   3010B81       /usr/lib/libcrypt.a[shr.o]
d0652100        d0653654              5K rwx   m     SLIBTEXT   3010B81       ./libself.a[support32.o]
d0654380        d0654a02              1K rwx   m     SLIBTEXT   3010B81       ./libself.a[shr32.o]
d0655a80        d0656105              1K rwx   m     SLIBTEXT   3010B81       ./libself.a[dl_shr32.o]
f05935cc        f0593844              0K rw-   sm    PLIBDATA   17F0DFF       ./libself.a[shr32.o]
f06a5b6f        f06a60c0              1K rw-   sm    PLIBDATA   17F0DFF       ./libself.a[support32.o]
f07b4ccc        f07b4f44              0K rw-   sm    PLIBDATA   17F0DFF       ./libself.a[dl_shr32.o]
f07dfbb0        f08b7388            861K rw-   sm    PLIBDATA   17F0DFF       /usr/lib/libc.a[shr.o]
f08b86a8        f08b87c8              0K rw-   sm    PLIBDATA   17F0DFF       /usr/lib/libcrypt.a[shr.o]
                          Total   528579K
```

3. To display the address ranges with file name association for the mapped file of process with PID 2031848, enter:

```
# procmap -Xf 2031848
2031848 : ./self_mod_64

Start-ADD         End-ADD              SIZE MODE  PSIZ  TYPE       VSID      MAPPED OBJECT
0                 10000000           262144K r--   m     KERTXT     20002
100000000         100002058               8K rwx   sm    MAINTEXT   3C0CBC    self_mod_64
1100002d0         1100009e0               1K rw-   sm    MAINDATA   240CA4    self_mod_64
1100009e0         110010a00              64K rw-   sm    HEAP       240CA4
800000000000e80   8000000000012eb         1K rwx   sm    PLIBTEXT   5A0CDA    ./libself_priv.a[dl_prv64.o]
8001000a0000180   8001000a00001c0         0K rw-   sm    PLIBDATA   16A0DEA   ./libself_priv.a[dl_prv64.o]
800200140000000   80020014003d000       244K r--   sm    KER/LDR    2A20E22
8fffffff0000000   900000000000000    262144K r--   s     KER/LDR    530CD3
900000000000e00   900000000440541      4349K rwx   m     SLIBTEXT   28C0E0C   /usr/lib/libc.a[shr_64.o]
```

```
900000000466400   900000000466f43         2K rwx   m     SLIBTEXT   28C0E0C   /usr/lib/libcrypt.a[shr_64.o]
900000000467980   900000000468e98         5K rwx   m     SLIBTEXT   28C0E0C   ./libself.a[support64.o]
900000000469100   900000000469568         1K rwx   m     SLIBTEXT   28C0E0C   ./libself.a[shr64.o]
90000000046a800   90000000046ac6b         1K rwx   m     SLIBTEXT   28C0E0C   ./libself.a[dl_shr64.o]
9001000a0000580   9001000a010cb88      1073K rw-   sm    PLIBDATA   12A0DAA   /usr/lib/libc.a[shr_64.o]
9001000a010da28   9001000a010dbb8         0K rw-   sm    PLIBDATA   12A0DAA   /usr/lib/libcrypt.a[shr_64.o]
9001000a0137380   9001000a01378b8         1K rw-   sm    PLIBDATA   12A0DAA   ./libself.a[support64.o]
9001000a0248400   9001000a0248440         0K rw-   sm    PLIBDATA   12A0DAA   ./libself.a[shr64.o]
9001000a045ab00   9001000a045ab40         0K rw-   sm    PLIBDATA   12A0DAA   ./libself.a[dl_shr64.o]
900200140000000   900200150000000    262144K r--   s     KER/LDR    B70037
9fffffffd0000000  9fffffffe0000000   262144K r--   sm    KER/LDR    50005
9fffffffe0000000  9ffffffff0000000   262144K r--   sm    KER/LDR    E000E
9ffffffff0000000  9ffffffff000fa8e       62K rwx   s     SLIBTEXT   1180198   /usr/ccs/bin/usla64
9ffffffff000fa8e  9ffffffff000fa8e        0K rw-   s     PLIBDATA   1180198   /usr/ccs/bin/usla64
a00000000000000   a00000000001000         4K rw-   sm    SMMFILE    420CC2    /tmp/mmfile1
a00000000001000   a00000000002000         4K r--   sm    SMMFILE    420CC2    /tmp/mmfile2
a00000000002000   a00000000003000         4K rw-   sm    PMMFILE    420CC2    /tmp/mmfile1
a00000000003000   a00000000004000         4K r--   sm    PMMFILE    420CC2    /tmp/mmfile2
a00000000004000   a00000000005000         4K rw-   sm    SMMPSXRT   420CC2    POSIX RT SHM 1
a00000000005000   a00000000006000         4K r--   sm    SMMPSXRT   420CC2    POSIX RT SHM 1
a00000000006000   a00000000007000         4K rw-   sm    PMMPSXRT   420CC2    POSIX RT SHM 2
a00000000007000   a00000000008000         4K r--   sm    PMMPSXRT   420CC2    POSIX RT SHM 2
a00000000008000   a00000000009000         4K rw-   sm    SMMAP      420CC2
a00000000009000   a0000000000a000         4K r--   sm    SMMAP      420CC2
a0000000000a000   a0000000000b000         4K rw-   sm    PMMAP      420CC2
a0000000000b000   a0000000000c000         4K r--   sm    PMMAP      420CC2
a00010000000000   a00010000000000         0K rw-   s     SHMFILE    250CA5   /tmp/mmfile1
a00010010000000   a00010010000000         0K r--   s     SHMFILE    1960D16  /tmp/mmfile2
a00020000000000   a00020000001000         4K rw-   sm    SHM        5B0CDB   shmid:18
a00030000000000   a00030000001000         4K r--   sm    SHM        1980D18  shmid:19
fffffff00000000   1000000000000000  4194304K rw-   sm    STACK      10D0D8D
                            Total    5510897K
```

Files

Item	Description
/proc	Contains the **/proc** filesystem.

procrun Command

Purpose

Starts a process that has stopped on the **PR_REQUESTED** event.

Syntax

procrun *ProcessID* ...

Description

The **/proc** filesystem provides a mechanism to control processes. It also gives access to information about the current state of processes and threads, but in binary form. The proctools commands provide ascii reports based on some of the available information.

Most of the commands take a list of process IDs or **/proc/***ProcessID* strings as input. The shell expansion **/proc/*** can therefore be used to specify all processes in the system.

Each of the proctools commands gathers information from **/proc** for the specified processes and displays it to the user. The proctools commands like procrun and procstop start and stop a process using the **/proc** interface.

The information gathered by the commands from **/proc** is a snapshot of the current state of processes, and therefore can vary at any instant except for stopped processes.

The **procrun** command starts the process that has stopped on the **PR_REQUESTED** event.

Flags

Item	Description
ProcessID	Specifies the process id.

Examples

1. To restart process 30192 which was stopped on the **PR_REQUESTED** event, enter:

```
procrun 30192
```

Files

Item	Description
/proc	Contains the **/proc** filesystem.

procsig Command

Purpose

Lists the signal actions defined by processes.

Syntax

procsig *ProcessID* ...

Description

The **/proc** filesystem provides a mechanism to control processes. It also gives access to information about the current state of processes and threads, but in binary form. The proctools commands provide ASCII reports based on some of the available information.

Most of the commands take a list of process IDs or **/proc/***ProcessID* strings as input. The shell expansion **/proc/*** can therefore be used to specify all processes in the system.

Each of the proctools commands gathers information from **/proc** for the specified processes and displays it to the user. The proctools commands like procrun and procstop start and stop a process using the **/proc** interface.

The information gathered by the commands from **/proc** is a snapshot of the current state of processes, and therefore can vary at any instant except for stopped processes.

The **procsig** command lists the signal actions defined by processes.

Flags

Item	Description
ProcessID	Specifies the process id.

Examples

1. To list all the signal actions defined for process 11928, enter:

```
procsig 11928
```

The output of this command might look like this:

```
HUP         caught
INT         caught
QUIT        caught
ILL         caught
TRAP        caught
ABRT        caught
EMT         caught
FPE         caught
KILL        default  RESTART
BUS         caught
SEGV        default
SYS         caught
PIPE        caught
ALRM        caught
TERM        ignored
URG         default
STOP        default
TSTP        ignored
CONT        default
CHLD        default
TTIN        ignored
TTOU        ignored
IO          default
XCPU        default
XFSZ        ignored
MSG         default
WINCH       default
PWR         default
USR1        caught
USR2        caught
PROF        default
DANGER      default
VTALRM      default
MIGRATE     default
PRE         default
VIRT        default
ALRM1       default
WAITING     default
CPUFAIL     default
KAP         default
RETRACT     default
SOUND       default
SAK         default
```

Files

Item	Description
/proc	Contains the **/proc** filesystem.

procstack Command

Purpose

Prints the hexadecimal addresses and symbolic names for all the threads in the process.

Syntax

procstack [**-F**] [**-g**] *ProcessID* ...

Description

The /proc filesystem provides a mechanism to control processes. It also gives access to information about the current state of processes and threads, but in binary form. The **proctools** commands provide ASCII reports based on some of the available information.

Most of the commands take a list of process IDs or **/proc/***ProcessID* strings as input. The shell expansion **/proc/*** can therefore be used to specify all processes in the system.

Each of the **proctools** commands gathers information from /proc for the specified processes and displays it to the user. The **proctools** commands like **procrun** and **procstop** start and stop a process using the **/proc** interface.

The information gathered by the commands from **/proc** is a snapshot of the current state of processes, and therefore can vary at any instant except for stopped processes.

The **procstack** command prints the hexadecimal addresses and symbolic names for all the threads in the process.

Flags

Item	Description
-F	Forces the **procstack** command to take control of the target process even if another process has control.
-g	Prevents the conversion of symbol names to human-readable names.
ProcessID	Specifies the process ID.

Examples

1. To display the current stack of process 11928, enter:

```
procstack 11928
```

   The output of this command might look like this:

```
11928 : -sh
d01d15c4  waitpid   (?, ?, ?) + e0
10007a1c  job_wait   (?) + 144
10020298  xec_switch   (?, ?, ?, ?, ?) + 9c0
10021db4  sh_exec   (?, ?, ?) + 304
10001370  exfile   () + 628
10000300  main   (?, ?) + a1c
10000100  __start   () + 8c
```

2. To display the current stack of all the threads of the multi-threaded process 28243 for application *appl*, enter:

```
procstack 28243
```

   The output of this command would look like this:

```
28243 : appl
---------- tid# 54321 -----------
d0059eb4  _p_nsleep   (?, ?) + 10
d01f1fc8  nsleep   (?, ?) + b4
d026a6c0  sleep   (?) + 34
100003a8  main   () + 98
10000128  __start   () + 8c
```

```
---------- tid# 43523 ----------
d0059eb4  _p_nsleep   (?, ?) + 10
d01f1fc8  nsleep   (?, ?) + b4
d026a6c0  sleep   (?) + 34
10000480  PrintHello   (d) + 30
d004b314  _pthread_body   (?) + ec
---------- tid# 36352 ----------
d0059eb4  _p_nsleep   (?, ?) + 10
d01f1fc8  nsleep   (?, ?) + b4
d026a6c0  sleep   (?) + 34
10000480  PrintHello   (c) + 30
d004b314  _pthread_body   (?) + ec
```

Files

Item	Description
/proc	Contains the **/proc** filesystem.

procstop Command

Purpose

Stops processes on the **PR_REQUESTED** event.

Syntax

procstop *ProcessID* ...

Description

The **/proc** filesystem provides a mechanism to control processes. It also gives access to information about the current state of processes and threads, but in binary form. The **proctools** commands provide ASCII reports based on some of the available information.

Most of the commands take a list of process IDs or **/proc/***ProcessID* strings as input. The shell expansion **/proc/*** can therefore be used to specify all processes in the system.

Each of the proctools commands gathers information from **/proc** for the specified processes and displays it to the user. The **proctools** commands like **procrun** and **procstop** start and stop a process using the **/proc** interface.

The information gathered by the commands from **/proc** is a snapshot of the current state of processes, and therefore can vary at any instant except for stopped processes.

The **procstop** command stops processes on the **PR_REQUESTED** event.

Flags

Item	Description
ProcessID	Specifies the process id.

Examples

1. To stop process 7500 on the **PR_REQUESTED** event, enter:

```
procstop 7500
```

Files

Item	Description
/proc	Contains the **/proc** filesystem.

proctree Command

Purpose

Prints the process tree containing the specified process IDs or users.

Syntax

proctree [**-a**] [{ *ProcessID* | *User* }]

proctree [**-a**] [**-T**] [**-t**] [{ *-p ProcessID* | *-u User* }] [**-@** [*WparName*]]

Description

The **/proc** filesystem provides a mechanism to control processes. It also gives access to information about the current state of processes and threads, but in binary form. The **proctools** commands provide ASCII reports based on some of the available information.

Most of the commands take a list of process IDs or **/proc/***ProcessID* strings as input. The shell expansion **/proc/*** can therefore be used to specify all processes in the system.

Each of the proctools commands gathers information from **/proc** for the specified processes and displays it to the user. The **proctools** commands like **procrun** and **procstop** start and stop a process using the **/proc** interface.

The information gathered by the commands from **/proc** is a snapshot of the current state of processes, and therefore can vary at any instant except for stopped processes.

The **proctree** command prints the process tree containing the specified process IDs or users. The child processes are indented from their respective parent processes. An argument of all digits is taken to be a process ID, otherwise it is assumed to be a user login name. The default action is to report on all processes, except children of process 0.

When you specify the -@ flag with no parameters, all WPAR names are displayed. If you specify the *WparName* parameter, only those WPAR names are displayed.

For displaying thread IDs and associated pthread IDs, specify the **-t** option. For a kernel process, the **proctree** command displays only the thread ID.

Note: If the information about the process cannot be retrieved, the **proctree** command displays only the process ID. The other information about the process is shown as blank. For example, the **proctree** command shows only the process ID for the zombie process.

Flags

Item	Description
-a	Includes children of process 0 in the display. The default is to exclude them.
ProcessID	Specifies the process ID.
-p ProcessID	Specifies the process ID.
-T	Displays the formatted output of the process tree.
-t	Displays thread IDs and associated pthread IDs for the process.

Item	Description
User	Specifies the user name.
-u User	Specifies the user name.
-@	Displays all WPAR names. **Note:** The **-@** flag is not supported when executed within a workload partition.
-@ WparName	Displays only the processes of the WPAR you specify using the *WparName* parameter. **Note:** The **-@** flag is not supported when executed within a workload partition.

Examples

1. To display the ancestors and all the children of the 12312 process, enter the following command:

   ```
   proctree 12312
   ```

   The output of this command might look like this:

   ```
   4954    /usr/sbin/srcmstr
      7224    /usr/sbin/inetd
         5958    telnetd -a
            13212    -sh
               14718    ./proctree 13212
   ```

2. To display the ancestors and children of the 12312 process, including children of process 0, enter the following command:

   ```
   proctree -a 12312
   ```

   The output of this command might look like this:

   ```
   1    /etc/init
      4954    /usr/sbin/srcmstr
         7224    /usr/sbin/inetd
            5958    telnetd -a
               13212    -sh
                  14724    ./proctree -a 13212
   ```

3. To display the process tree of WPAR corral2, enter the following command:

   ```
   proctree -@ corral2
   ```

   The output of this command might look like this:

   ```
   corral2       401496   /etc/init
   corral2       319680   /usr/sbin/srcmstr
   corral2       102636   /usr/sbin/inetd
   corral2       249954   /opt/rsct/bin/rmcd -a IBM.LPCommands -r
   corral2       254132   /opt/rsct/bin/IBM.AuditRMd
   corral2       295098   /opt/rsct/bin/IBM.ServiceRMd
   corral2       303218   /usr/dt/bin/dtlogin
   corral2       307370   /usr/sbin/writesrv
   corral2       323836   /usr/sbin/qdaemon
   corral2       331970   /usr/sbin/muxatmd
   corral2       348210   /usr/sbin/syslogd
   corral2       352472   sendmail: accepting connections H nnections
   corral2       364564   /opt/rsct/bin/IBM.ERrmd
   corral2       405522   /usr/sbin/portmap
   corral2       282800   /usr/bin/xmwlm -L
   corral2       311454   /usr/sbin/cron
   corral2       376920   /usr/lib/errdemon
   ```

4. To display the WPAR name of the processes, enter the following command:

```
proctree -@
```

The output of this command might look like this:

```
Global     114788   /usr/dt/bin/dtlogin -daemon
Global     86108   dtlogin <:0>        -daemon
Global     123022   dtgreet 8  :0
Global     77944   /usr/lib/errdemon
Global     94314   /usr/sbin/syncd 60
Global     168084   /usr/sbin/srcmstr
Global     110688   /opt/rsct/bin/IBM.ServiceRMd
corral2    401496   /etc/init
corral2    319680   /usr/sbin/srcmstr
corral2    102636   /usr/sbin/inetd
corral2    249954   /opt/rsct/bin/rmcd -a IBM.LPCommands -r
corral2    254132   /opt/rsct/bin/IBM.AuditRMd
corral2    331970   /usr/sbin/muxatmd
corral2    348210   /usr/sbin/syslogd
corral2    364564   /opt/rsct/bin/IBM.ERrmd
corral2    405522   /usr/sbin/portmap
corral2    282800   /usr/bin/xmwlm -L
corral2    311454   /usr/sbin/cron
corral2    376920   /usr/lib/errdemon
Global     151626   /usr/ccs/bin/shlap64
Global     274578   /usr/sbin/getty /dev/console
...
```

5. To display the ancestors, all of the children, and the WPAR name of the 102636 process, enter the following command:

```
proctree  -p 102636 -@
```

The output of this command might look like this:

```
Global     168084    /usr/sbin/srcmstr
corral2    401496    /etc/init
corral2    319680    /usr/sbin/srcmstr
corral2    102636    /usr/sbin/inetd
```

6. To display the formatted process-tree output of the 213246 process, enter the following command:

```
proctree -T -p 213246
```

The output of this command might look like this:

```
192652        \--/usr/sbin/srcmstr
200830              \--/usr/sbin/inetd
213246                    \--telnetd -a
229592                          \---ksh
```

7. To display thread IDs and associated pthread IDs for the 344172 process, enter the following command:

```
proctree -t -p 344172
```

The output of this command might look like this:

```
192652    /usr/sbin/srcmstr
     TID : 225535 (pTID :      1)
   200830    /usr/sbin/inetd
        TID : 360677 (pTID :      1)
      323642    telnetd -a
           TID : 770057 (pTID :      1)
         307428    -ksh
              TID : 1056861 (pTID :      1)
            344172    appthd
                 TID : 1065119 (pTID :      1)
                 TID : 1028171 (pTID :    258)
                 TID : 1011789 (pTID :   2057)
                 TID : 1024105 (pTID :   1800)
```

8. To display the formatted process-tree output for the 344172 process along with thread IDs and associated pthread IDs, enter the following command:

```
proctree -tT -p 344172
```

The output of this command might look like this:

```
192652   \--/usr/sbin/srcmstr
             ~~TID : 225535 (pTID :      1)
200830       \--/usr/sbin/inetd
                 ~~TID : 360677 (pTID :      1)
323642           \--telnetd -a
                      ~~TID : 770057 (pTID :      1)
307428                \---ksh
                         ~~TID : 1056861 (pTID :      1)
344172                        \--appthd
                                 |~~TID : 1065119 (pTID :      1)
                                 |~~TID : 1028171 (pTID :    258)
                                 |~~TID : 1011789 (pTID :   2057)
                                  ~~TID : 1024105 (pTID :   1800)
```

Files

Item	Description
/proc	Contains the **/proc** filesystem.

procwait Command

Purpose

Waits for all of the specified processes to terminate.

Syntax

procwait [**-v**] *ProcessID* ...

Description

The **/proc** filesystem provides a mechanism to control processes. It also gives access to information about the current state of processes and threads, but in binary form. The **proctools** commands provide ASCII reports based on some of the available information.

Most of the commands take a list of process IDs or **/proc/***ProcessID* strings as input. The shell expansion **/proc/*** can therefore be used to specify all processes in the system.

Each of the proctools commands gathers information from **/proc** for the specified processes and displays it to the user. The **proctools** commands like **procrun** and **procstop** start and stop a process using the **/proc** interface.

The information gathered by the commands from **/proc** is a snapshot of the current state of processes, and therefore can vary at any instant except for stopped processes.

The **procwait** command waits for all of the specified processes to terminate.

Flags

Item	Description
-v	Specifies verbose output. Reports terminations to standard output.
ProcessID	Specifies the process id.

Examples

1. To wait for process 12942 to exit and display the status, enter:

```
procwait -v 12942
```

   The output of this command might look like this:

```
12942 : terminated, exit status 0
```

Files

Item	Description
/proc	Contains the **/proc** filesystem.

procwdx Command

Purpose

Prints the current working directory of processes.

Syntax

procwdx [**-F**] *ProcessID* ...

Description

The **/proc** filesystem provides a mechanism to control processes. It also gives access to information about the current state of processes and threads, but in binary form. The **proctools** commands provide ASCII reports based on some of the available information.

Most of the commands take a list of process IDs or **/proc/***ProcessID* strings as input. The shell expansion **/proc/*** can therefore be used to specify all processes in the system.

Each of the proctools commands gathers information from **/proc** for the specified processes and displays it to the user. The **proctools** commands like **procrun** and **procstop** start and stop a process using the **/proc** interface.

The information gathered by the commands from **/proc** is a snapshot of the current state of processes, and therefore can vary at any instant except for stopped processes.

The **procwdx** command prints the current working directory of processes.

Flags

Item	Description
-F	Forces procfiles to take control of the target process even if another process has control.
ProcessID	Specifies the process id.

Examples

1. To display the current working directory of process 11928, enter:

```
procwdx 11928
```

   The output of this command might look like this:

```
11928 :  /home/guest
```

Files

Item	Description
/proc	Contains the **/proc** filesystem.

prof Command

Purpose

Displays object file profile data.

Syntax

prof [**-t** | **-c** | **-a** | **-n**] [**-o** | **-x**] [**-g**] [**-z**] [**-h**] [**-s**] [**-S**] [**-v**] [**-L** *PathName*] [*Program*] [**-m** *MonitorData* ...]

Description

The **prof** command interprets profile data collected by the **monitor** subroutine for the object file *Program* (**a.out** by default). It reads the symbol table in the object file *Program* and correlates it with the profile file (**mon.out** by default). The **prof** command displays, for each external text symbol, the percentage of execution time spent between the address of that symbol and the address of the next, the number of times that function was called, and the average number of milliseconds per call.

Note: Symbols from C++ object files have their names demangled before they are used.

To tally the number of calls to a function, you must have compiled the file using the **cc** command with the **-p** flag. The **-p** flag causes the compiler to insert a call to the **mcount** subroutine into the object code generated for each recompiled function of your program. While the program runs, each time a parent calls a child function the child calls the **mcount** subroutine to increment a distinct counter for that parent-child pair. Programs not recompiled with the **-p** flag do not have the **mcount** subroutine inserted and therefore keep no record of which function called them.

The **-p** flag also arranges for the object file to include a special profiling startup function that calls the **monitor** subroutine when the program begins and ends. The call to the **monitor** subroutine when the program ends actually writes the **mon.out** file. Therefore, only programs that explicitly exit or return from the main program cause the **mon.out** file to be produced.

Note: To change the name of the generated output file, use the PROF environment variable and set it as follows:

```
PROF=filename:<filename>
```

For example, if you set PROF=`myprof`, then the generated file will be named as `myprof.out`.

The location and names of the objects loaded are stored in the **mon.out** file. If you do not select any flags, **prof** will use these names. You must specify a program or use the **-L** option to access other objects.

Note: Imported external routine calls, such as a call to a shared library routine, have an intermediate call to local **glink** code that sets up the call to the actual routine. If the timer clock goes off while running this code, time is charged to a routine called *routine***.gl**, where *routine* is the routine being called. For example, if the timer goes off while in the **glink** code to call the **printf** subroutine, time is charged to the **printf.gl** routine.

Flags

The mutually exclusive flags **a**, **c**, **n**, and **t** determine how the **prof** command sorts the output lines:

Item	Description
-a	Sorts by increasing symbol address.
-c	Sorts by decreasing number of calls.
-n	Sorts lexically by symbol name.
-t	Sorts by decreasing percentage of total time (default).

Note: The **prof** command can still run successfully if you use more than one of flags **a**, **c**, **n**, and **t** in the same command. The **prof** command accepts the first of these flags it encounters on the command line and ignores the others.

The mutually exclusive flags **o** and **x** specify how to display the address of each symbol monitored.

Item	Description
-o	Displays each address in octal, along with the symbol name.
-x	Displays each address in hexadecimal, along with the symbol name.

Note: The **prof** command can still run successfully if you use both the **-o** and **-x** flags in the same command. The **prof** command accepts the first of these two flags it encounters on the command line and ignores the other flag.

Use the following flags in any combination:

Item	Description
-g	Includes non-global symbols (static functions).
-h	Suppresses the heading normally displayed on the report. This is useful if the report is to be processed further.
-L *PathName*	Uses alternate path name for locating shared objects.
-m *MonitorData*	Takes profiling data from *MonitorData* instead of **mon.out**.
-s	Produces a summary file in **mon.sum**. This is useful when more than one profile file is specified.
-S	Displays a summary of monitoring parameters and statistics on standard error.
-v	Suppresses all printing and sends a graphic version of the profile to standard output for display by the plot filters. When plotting, low and high numbers, by default 0 and 100, can be given to cause a selected percentage of the profile to be plotted with accordingly higher resolution.
-z	Includes all symbols in the profile range, even if associated with 0 (zero) calls and 0 (zero) time.

Examples

1. To display, without a header, the amount of time spent at each symbol address, sorted by time, enter:

```
prof -t -h
```

2. The following example obtains a local version of any shared libraries used to create the **runfile** file in the **/home/score/lib** directory. The data file used will be **runfile.mon** rather than **mon.out**.

```
prof -x -L/home/score/lib runfile -m runfile.mon
```

Files

Item	Description
mon.out	Default profile.
a.out	Default object file.
mon.sum	Summary profile.

proff Command

Purpose

Formats text for printers with personal printer data streams.

Syntax

proff [**-L***List*] [**-P***Printer*] [**-t**] [*nroffFlags*] [*File* ...]

Description

The **proff** command formats text by using the **nroff** command on the specified files for printers that support ppds (personal printer data streams), such as the Quietwriter III printer, the Quickwriter printer, and the Proprinter printer.

If no file is specified, standard input is read. A parameter value of **-** (minus) specifies standard input.

Parameters

Item	Description
nroffFlags	Specifies the **nroff** command flags used by the **proff** command to format the text file for a ppds-supported printer output.
File	Specifies the text file that the **proff** command formats for printers that support ppds.

Flags

Item	Description
-L*List*	Passes the specified list as flags for the **qprt** command. To pass a single flag to the **qprt** command, use the **-L** flag followed immediately by the **nroff** command flag being passed. For example: `-L-h.` To pass multiple flags or a string to the **lpr** command, use the **-L** flag followed immediately by the flags or string enclosed by " " (double quotes): `-L"-h -r -m".`
-P*Printer*	Sends output to a specified printer corresponding to an entry in the **/etc/qconfig** file. The default is taken from the **PRINTER** environment variable, if it exists; otherwise the system default queue name is used.
-t	Sends output to standard output.

Item	Description
-	Specifies that standard input is used as the source for the formatting process. All other flags are passed to the **nroff** command.

Example

The following is a typical command sequence to process output for the IBM Proprinter printer:

```
proff -t testfile
```

Environment Variable

Item	Description
PRINTER	Specifies the desired printer queue.

Files

Item	Description
/usr/share/lib/nterm/tab.ppds	Contains driving tables for printers with personal printer data streams.
/etc/qconfig	Describes the queues and devices.

projctl Command

Purpose

Supports project-based advanced accounting activities.

Syntax

projctl **add** *projname projnumber* [*comment*] [{ **-d** *projpath* | **-p** [*DN*] }]

projctl **merge** *sourceprojpath* [**-d** *targetprojfile*]

projctl **rm** *projname* [{ **-d** *projpath* | **-p** [*DN*] }]

projctl **chg** *projname* [**-p** *pid* [, *pid*]] [**-f**]

projctl **exec** *projname* <*cmd line*> [**-f**]

projctl **chattr agg** *projname* {**-s**|**-u**} [{ **-d** *projpath* | **-p** [*DN*] }]

projctl *qpolicy* [**-g** [*DN*]]

projctl **qprojs** [**-n**]

projctl **qproj** [*projectname*]

projctl **qapp** *appname*

projctl {**chkusr** | **chkgrp** | **chkprojs** | {{**chkadm** | **chkall**} [**-d** *admpath*]}}

projctl **ldusr** [**-r**] [**-a**]

projctl **unldusr** [**-a**]

projctl **ldgrp** [**-r**] [**-a**]

projctl **unldgrp** [**-a**]

projctl ldprojs -g [**-r**] [**-a**]

projctl ldprojs -g [*DN*] **-d** *projpath*

projctl ldprojs -p [*DN*] **-d** *projpath*

projctl unldprojs -g [*DN*] [**-f**] [**-a**]

projctl unldprojs -p [*DN*]

projctl ldadm -g [*name*] [**-r**] [**-a**]

projctl ldadm -g [*name:*]*DN* | *name*] **-d** *admpath*

projctl ldadm -p [[*name:*]*DN* | *name*] **-d** *admpath*

projctl unldadm -g [**-a**]

projctl unldadm -p [[*name:*]*DN* | *name*]

projctl ld [**-r**]

projctl ldall [**-d** *admpath*] [**-r**] [**-a**]

projctl unldall [**-f**] [**-a**]

Description

The various subcommands of **projctl** command perform project-based advanced accounting activities such as adding a new project, removing a new project, and loading a specific accounting policy. These various options of **projctl** command are as explained below.

Flags

Item	Description
-a	Automatically loads the policies during system reboot.
-d	Generally specifies the path from where the project definition file or the admin policy file should be referred. When used in the `merge` subcommand, it specifies the target project definition file where the merged project definitions are to be stored.
-f	Overrides the policy rules when specified with `chg` and `exec` subcommands. Clears the project assigned to the processes when called with `unldall` subcommand. Force unload all the project definitions when called with `unldprojs` subcommand.
-g	Specifies that the projects and policies are to be downloaded from the LDAP repository.
-n	Sorts the list of project definitions based on the name.
-p	When used in the `chg` subcommand, passes the list of process IDs that require a change in project assignment. When used in the `add`, `rm`, and `chattr` subcommands, specifies the LDAP DN where the project definition is to be updated. When used in the `ld` and `unld` subcommands, specifies that the projects and policies are to be uploaded to the LDAP repository. Its argument indicates the DN where the projects and policies are to be uploaded.
-r	Reloads the policies.
-s	Used in **projctl chattr agg** subcommand to enable the project aggregation property.
-u	Used in **projctl chattr agg** subcommand to disable the project aggregation property.

Parameters

Item	Description
admpath	Path from where to select the admin policy file.

Item	Description
appname	Absolute path of the application whose project assignment list is requested.
cmd line	Absolute path of the command to be executed through **projctl exec** command.
comment	Project comments.
DN	Distinguished Name that indicates the absolute path to the project and policy objects on the LDAP server.
name	Name of the alternate admin policy definitions on the LDAP server.
pid	Process IDs.
projname	Name of the project.
projnumber	Numeric value for the project.
projpath	Path from where to select the project definition file.
sourceprojpath	Path from where the project definition file to be merged is to be picked up.
targetprojfile	Target project definition file where the project definitions should be merged.

Subcommands

add Subcommand

The **add** subcommand adds the definition of the project to the project definition file. If the **–d** flag is specified then the project definition is added into the project definition file, under the named path. The default is to add to the **/etc/project/projdef** system project definition file. The project definition file under any other path should be named as **.projdef:**. If the new project is to be added to the system project definition file and the projects are already loaded in kernel, then the specified new project will be added into kernel project registry. Otherwise, the entry will be made only in the file. The **add** subcommand takes the project name, project number, and an option argument for project comments as parameters. By default, the aggregation property of the project will be set to no for all the projects created using this command.

If `-p` is specified, the new project definition is added to default project *DN* or the specified *DN* on the LDAP server. If `-p` is not specified, `.config` will provide source information. Running the `-p` option requires root authority.

Each entry created by **projctl add** in the Project Definition File has the following format:

```
ProjectName:ProjectNumber:AggregationStatus::Comment
```

Examples for Project Definitions that illustrate the file format are as follows:

```
:: Project Definition File
:: Dated: 23-JUN-2003
AIX:3542:yes::To Classify AIX Legacy Applications
Test_Project:0x10000:yes::To Classify Testing work
```

chattr agg Subcommand

The **chattr agg** subcommand enables and disables aggregation property for the given project. If **-s** flag is used the aggregation is enabled. If **-u** flag is used aggregation is disabled. If **–d** flag is specified then the project definition is updated in the project definition file under the specified path. The default is to update the system project definition file (**/etc/project/projdef**). If the update is to the system project definition file and it is already loaded in kernel, then the specified new project is updated in kernel project registry as well. Otherwise, the changes will be made only to the project definition file.

If `-p` is specified, the project definition is modified on default project *DN* or the specified DN on the LDAP server. If `-p` is not specified, `.config` will provide source information. Executing the `-p` option requires root authority.

chg Subcommand

The **chg** subcommand enables the user to change the list of projects that the user is permitted to use for his processes. The intended project name is given as input to this subcommand. If the process IDs are provided as input, those processes will be classified under the specified project. If there are no process IDs provided as input, the project change will happen to the process which started the **projctl** command.

By default, the **chg** subcommand changes the project assignment within the scope of available rules. To override the rules and assign a project directly to a process, the **-f** force option must be specified.

chk Subcommand

The **chk** subcommands check the validity of various project policies. The subcommands validate the projects and policies so that they can be loaded safely into the kernel. There are several **chk** subcommands to support various project policies. The subcommands include:

Item	Description
chkadm	Validates the admin policies. Each rule in the admin policy file usually has four attributes: user-id, group-id, application path name, and the project names. The **chkadm** subcommand checks whether these attributes are valid and reports any errors found in the policies. When the **-d** option is used, the **chkadm** subcommand uses the admin policy file from the specified path for checking the rules. It also uses the alias and the temporary project definition file (**.projdef**), if required. The projects used in the rule will be first searched in the system project definition file. If it is not found there, then the **.projdef** file under the specified path will be used.
chkall	Performs all the above validation activities, that is, it validates projects, user, group and admin policies. When the **–d** option is used, the **chkadll** subroutine uses the admin, alias, and project definition files from the specified path to validate the admin policies.
chkgrp	Validates the group policies. The validation involves checking whether the project list of the group contains valid projects.
chkprojs	Validates the system project definition file. Project Definitions are validated for uniqueness, project name and number validity, and attributes validity. The project name should be a POSIX alphanumeric string and the project number should be within the numeric range 0x00000001 - 0x00ffffff. The project numbers can be either decimal or hexadecimal numbers. All hexadecimal numbers should be shown with a prefix of `0x`. The aggregation property can be either a y or a n to indicate the status of aggregation. The **chkprojs** subcommand performs all these validity checks on the project definitions and reports any errors found with the project definitions.
chkusr	Validates the user policies. The validation involves checking whether the project list of the user contains valid projects.

Note: If wildcard characters are used in the admin policy rules then **chkadm** and **chkall** subcommands expand the wildcard characters and validate the output obtained.

exec Subcommand

The **exec** subcommand allows a user to launch arbitrary commands with any of the project names from the list of projects on which the command can work. Similar to **chg** option, used to override the rules and use any project to run the command line, the **-f** force option should be used. To get the list of projects that the command can be assigned to, use the **projctl qapp** subcommand.

ld Subcommand

The **ld** subcommands are used to load and reload projects and policies. There are specific load commands to perform the load operation on a specific policy. These various subcommands are as follows:

Item	Description
ld	Loads the policies, which should be loaded during the system startup. It refers the **/etc/project/.config** file to determine which policies to load. If the kernel is loaded already with any one policy or project definition, then this command simply returns.
ldadm	Loads the admin policies. Similar to **ldusr** and **ldgrp** subcommands, **ldadm** also checks and loads the projects first, if they are yet to be loaded. Then it loads the admin policy rules, after validating them. When the **-d** option is used, the admin policy file will be picked from the specified path. The alias and the temporary project definition file under the specified path will be used to check the existence of alias and project entries. After the policies are loaded, this subcommand also copies the admin policy file to **/etc/project/.admin**. Loading the admin policies related to LDAP is handled by the following `-p` and `-g` arguments: **projctl ldadm -g [*name*]** Specifies that an admin policy will be loaded into the kernel using the LDAP repository. If `-g` is not specified, the local admin policy (`/etc/project/admin`) will be downloaded into the kernel. **projctl ldadm -g [[*name*:]DN | *name*] -d *admpath*** Specifies that an LDAP admin policy will be downloaded to a local file without downloading the policy into the kernel. The source admin policy is located at the specified DN or is found using the accounting DNs in the `ldap.cfg` file. The `-d` parameter is used to specify where the policy files (projects, admin, and alias) are written. If the target location is below `/etc/project/`, the files are written according to the conventions used by the system. Files are written to: • `/etc/project/admin`, `/etc/project/alias`, `/etc/project/projdef` • `/etc/project/ldap/admin`, `/etc/project/ldap/alias`, `/etc/project/ldap/projdef` • `/etc/project/projdef`, `/etc/project/alter/policyname/admin`, `.../alias` • `/etc/project/ldap/projdef`, `/etc/project/ldap/alter/policyname/admin`, `.../alias` Otherwise, the three files are written to the specified directory. When an explicit DN is specified with the `-g` option, the projects are not downloaded because they could also be located on a different DN. In this case, the user has to download them separately. **projctl ldadm -p [[*name*:]DN | *name*] -d *admpath*** Specifies that an admin policy located at the directory *localpath* will be uploaded to the LDAP server. This command also uploads projects that it finds in the `localpath/.projdef` temporary project definition file. When an explicit DN is specified with the `-p` option, only the admin policy is uploaded to the LDAP server because the projects could be located on a different DN. In this case, the user must explicitly upload the respective `.projdef` file to the appropriate DN. The system does not know the identity of this DN. The `-d` argument must be specified when the `-g` or `-p` arguments are used. The `-r` and `-a` arguments cannot be specified with the `-p` argument. If the `-a` argument is specified and the `-g` argument is not specified, the admin policies in the `.config` file are loaded. If the `-r` option is used, the `.active` file is used to determine the identity of the policies to load. The `-r` and `-a` options cannot be used together.
ldall	Downloads user, group, and admin policies into the kernel. Similar to the `ldusr` and `ldgrp` commands, this option attempts to download LDAP projects if an accounting DN has been specified for projects, because the User and Group Policies are not associated with Local or LDAP Users individually. This command attempts to download the default Admin policy using the configured admin DN in addition to downloading the Local Admin Policy.

Item	Description
ldgrp	Loads the group project policies. If they are not yet loaded, the **ldgrp** subcommand checks and loads the projects first. It then verifies the validity of the project list for all the groups and loads the rules.
ldprojs	Loads the project definitions from the system project definition **/etc/project/projdef** file. Before loading the projects, it checks the validity of the rules. If the rules are found to valid, then it loads them. **projctl ldprojs -g** Specifies that the project definitions will be loaded into the kernel using the LDAP repository. **projctl ldprojs -p** Specifies that project definitions are to be uploaded to the LDAP server. If `-g` and the `-p` are not specified, the locally defined projects (`/etc/project/projdef`) are loaded into the kernel. **projctl ldprojs -g [*DN*] -d *localpdfpath*** Specifies that the project definition file from the LDAP repository will be downloaded to a local file without downloading the projects into the kernel. If the `-d` argument is not specified, the projects are downloaded to `/etc/project/ldap/projdef` and they are downloaded into the kernel. The `-d` argument directs you to create the file at the designated location, but not to download it into the kernel. In this case, the `projdef` file is created at the designated location rather than in the `.projdef` file. The source project definitions are located at the specified DN. Alternately, you can find them using the configured accounting DN in the `ldap.cfg` file. **projctl ldprojs -d *localpdfpath*** Loads the local project definition file into the kernel. **projctl ldprojs -p [*DN*] -d *localpdfpath*** Specifies that the project definitions located at the specified path will be uploaded to the LDAP server. The project definitions should be available in the `projdef` file at the specified directory. The `-d` argument must be specified when the `-g` or `-p` directs you to create the file at the designated location, but not to download it into the kernel. In this case, the `projdef` arguments are used. This way, the upload and download operations can be symmetric with respect to the specification of parameters. The `-r` and `-a` arguments cannot be specified with the `-p` argument. If the `-a` argument is specified and the `-g` argument is not specified, the project repositories in the `.config` file are loaded. If the `-r` option is used, the `.active` file is used to determine the project repositories to load. The `-r` and `-a` options cannot be used together.
ldusr	Loads the user project policies. If they are not yet loaded, the **lduser** subcommand checks and loads the projects first. It then verifies the validity of the project list for all the users and loads the rules.

Note:

- When **–r** option is used, all the above subcommands reload the respective policies. The **ld –r** subcommand queries the kernel to get the details of loaded policies and reloads them. The policy files to be reloaded will be referred from the **/etc/project/.active** file.
- When **ldadm** and **ldall** subcommands are issued with both the options **–d** and **–r**, **-r** will be ignored.
- All the **ld** subcommands update the **/etc/project/.active** file with the details of the policy that is loaded. When the **-a** option is passed, these subcommands also update the **/etc/project/.config** file in addition to updating the **.active** file. The **/etc/project/.config** file provides the details on the policies to be loaded automatically on system reboot.

merge Subcommand

The **merge** subcommand merges the projects defined in the project definition file under the specified path with the system project definition **/etc/project/projdef** file, by default. If a target project file name

is passed using the **-d** option, the project definitions under the specified path are merged with the target project definition file. The merge operation will fail if there are conflicting entries between the target project definition file and the project definition file under the specified path. The **merge** command skips any duplicate entries to maintain unique entries in the target project definition file.

qapp Subcommand

The **qapp** subcommand displays the list of projects that an application can switch to in the current environment. It displays the list of all projects with which the specified application can be started.

qpolicy Subcommand

The **qpolicy** subcommand displays the currently loaded policies. This command queries the kernel to get the information about the types of loaded policies and displays them. If `-g` is specified, this command lists the policies from the LDAP default admin DN or from the specified DN.

qproj Subcommand

The **qproj** subcommand displays the details of the project name passed as its argument. If no argument is passed, then this subcommand lists all the project definitions in the system to which the calling process can be assigned. The display format will be the same as that of **qprojs** subcommand.

qprojs Subcommand

The **qprojs** subcommand displays the list of all the project definitions that is currently loaded in the kernel registry. The **-n** option provides the list sorted based on the project name. The display contains the project name, project number, and its aggregation status.

rm Subcommand

The **rm** subcommand removes the definition of locally defined projects from the project definition file. If the **–d** flag is specified, then the project definition is removed from the project definition file under the specified path. The default is to remove it from the system project definition file (**/etc/project/projdef**). If the update is to the system project definition file and it is already loaded in kernel, then the specified project is removed from kernel project registry. Otherwise, the entry will be removed only from the file.

If `-p` is specified, the source will be the LDAP from where the project definitions are to be removed. If an explicit DN is specified, the project definition will be removed from that specific DN. If no DN is passed, the default DN configured in the `ldap.cfg` file will be used. If the LDAP projects are currently loaded, the project definition is removed from the kernel project registry and the local LDAP project file also. Otherwise, only the LDAP repository is updated.

Note: The `-p` and `-d` options cannot be used together. If neither of these options are specified, the `.config` file will be used to provide the source information. This command requires root authority to execute.

unld Subcommand

The **unld** subcommands are used to unload project policies. Similar to the **ld** subcommands, the **unld** subcommands are used to unload specific policies. These various subcommands are as follows:

Item	Description
unldadm	Unloads the admin policies.
unldall	Unloads all the loaded policies.
unldgrp	Unloads the group policies.
unldprojs	Unloads only the project definitions.
unldusr	Unloads the user policies.

Note:

- All these subcommands update the **.active** file after the respective policy is unloaded.

- When the **-a** option is used, the **/etc/project/.config** file is also updated with the unloaded status of the respective policy.
- The `-g` parameter specifies that the respective LDAP repository should be unloaded from the kernel. If `-g` is not specified, then the loaded repositories that are named in the `.active` file are unloaded.
- The `-p` option must be specified to remove the specified LDAP repository from the LDAP server.
- In the `unldadm` and `unsubcommand`, the *name* parameter indicates the admin policy name on the admin DN.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred. **1** Default error return code for read, write, and malloc failures. **2** EINVAL and ENOENT **3** EPERM and EACCES **4** EEXIST

Examples

1. To add a project `newproj` to the system project definition file, type:

   ```
   projctl add newproj 34 "Test Project"
   ```

2. To remove the project `test1` from the project definition file under the path `/tmp/myproj`, type:

   ```
   projctl rm test1 –d /tmp/myproj
   ```

3. To enable the aggregation status of the project `newproj`, type:

   ```
   projctl chattr agg newproj –s
   ```

4. To execute the **ps** command under the project `newproj`, overriding the existing rules, type:

   ```
   projctl exec newproj "/usr/bin/ps" –f
   ```

5. To retrieve the currently loaded policies, type:

   ```
   projctl qpolicy
   ```

   Output:

   ```
   Project definitions are loaded.
   Project definition file name:  /etc/project/projdef
   User policies are loaded.
   ```

6. To load the admin policies from the path `/tmp/myproj`, type:

   ```
   projctl ldadm –d /tmp/myproj
   ```

7. To unload all the project policies now and during system reboot, type:

   ```
   projctl unldall -a
   ```

8. To add a new project to the LDAP repository on a different DN, where DN is `ou=projects,ou=aacct,ou=cluster1,cn=aixdata`, type:

```
projctl add newproj 34 -p ou=projects,ou=aacct,ou=cluster1,cn=aixdata
```

9. To download the LDAP projects from the default DN to a local file under the `/etc/project/ldap` path, type:

```
projctl ldprojs -g -d /etc/project/ldap
```

10. To load the LDAP admin policies stored under the label `newdef` in the default DN to the kernel, type:

```
projctl ldadm -g newdef
```

Location

/usr/bin/projctl

Files

Item	Description
/usr/bin/projctl	Contains the **projctl** command.
/etc/project/projdef	Contains the system project definition file.
/etc/project/ldap/projdef	Contains the default LDAP project definition file.
/etc/project/.active	Contains the status of currently loaded policies.
/etc/project/.config	Contains the status of the policies to be loaded during system reboot.
/etc/security/ldap/ldap.cfg	Contains the LDAP client configuration details for handling advanced accounting data.

prompter Command

Purpose

Starts a prompting editor.

Syntax

prompter [**-erase** *Character*] [**-kill** *Character*] [**-prepend** | **-noprepend**] [**-rapid** | **-norapid**] *File*

Description

Part of the Message Handler (MH) package, the **prompter** command starts the prompting editor for message entry. The **prompter** command is not started by the user. The **prompter** command is called by other programs only.

The **prompter** command opens the file specified by the *File* parameter, scans it for empty components such as the `To:` component, and prompts you to fill in the blank fields. If you press the Enter key without filling in a required field, the **prompter** command deletes the component.

The **prompter** command accepts text for the body of the message after the first blank line or line of dashes in the file. If the body already contains text and the **-noprepend** flag is specified, the **prompter** command displays the text followed by the message:

```
--------Enter additional text
```

The **prompter** command appends any new text entered after the existing message. If you specify the **-prepend** flag, the **prompter** command displays the following message:

```
--------Enter initial text
```

Any new text precedes the body of the original message. When you press the Ctrl-D key sequence for End of File, the **prompter** command ends text entry and returns control to the calling program.

Flags

Item	Description
-erase *Character*	Sets the character to be used as the erase character. The value of the *Character* variable can be the octal representation of the character in the form \NNN where \NNN is a number or the character itself. For example, the character \e is \145 in octal representation.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-kill *Character*	Sets the character to be used as the kill, or stop, character. The value of the *Character* variable can be the octal representation of the character in the form \NNN where \NNN is a number or the character itself. For example, the character \e is \145 in octal representation.
-noprepend	Appends additional text after text already in the message body.
-norapid	Displays text already in the message body. This is the default.
-prepend	Appends additional text before text already in the message body. This is the default.
-rapid	Does not display text already in the message body.

Profile Entries

Item	Description
`Msg-Protect:`	Sets the protection level for your new message files.
`prompter-next:`	Specifies the editor used after exiting the **prompter** command.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
$HOME/ html	

proto Command

Purpose

Constructs a prototype file for a file system.

Syntax

proto *Directory* [*Prefix*]

Description

The **proto** command creates a prototype file for a file system or part of a file system. The **mkfs** command.

Specify the root directory from which the prototype file is made with the *Directory* parameter. The prototype file includes the complete subtree below the *Directory* parameter, and is contained on the same file system as the base directory specified by the *Directory* parameter.

The *Prefix* parameter is added to the names of all the initialization files, forcing the initialization files to be taken from a place other than the prototype. Before the output from the **proto** command can be used with the **LC_COLLATE** environment variables.

Example

To make a prototype file for an existing file system `/works`, enter:

```
proto /works
```

If the `/works` file system contains two directories called `dir1` and `dir2`, and the `dir1` directory contains the `file1` file, then the **proto** command displays:

```
#Prototype file for /works
d---  755  0 0
  dir1  d---  755  0 0
    file1         ----   644  0 0   /works/dir1/file1
          $
  dir2  d---  755  0 0
          $
    $
$
```

Files

Item	Description
/usr/sbin/proto	Contains the **proto** command.

proxymngr Command

Purpose

Proxy manager service.

Syntax

proxymngr [**-config** filename] [**-timeout** seconds] [**-retries** #] [**-verbose**]

Description

The **proxymngr** (proxy manager), is responsible for resolving requests from **xfindproxy** (and other similar clients), starting new proxies when appropriate, and keeping track of all of the available proxy services. The proxy manager strives to reuse existing proxies whenever possible.

There are two types of proxies that the proxy manager deals with, managed and unmanaged proxies.

A managed proxy is a proxy that is started on demand by the proxy manager.

An unmanaged proxy is started either at system boot time, or manually by a system administrator. The proxy manager is made aware of its existence, but no attempt is made by the proxy manager to start unmanaged proxies.

Flags

Item	Description
-config	Overrides the default **proxymngr** config file. See below for more details about the **proxymngr** config file.
-timeout	Sets the number of seconds between attempts made by the proxy manager to find an unmanaged proxy. The default is 10.
-retries	Sets the maximum number of retries made by the proxy manager to find an an unmanaged proxy. The default is 3.
-verbose	Causes various debugging and tracing records to be displayed as requests are received and proxies are started.

Proxy Manager Config File

The proxy manager maintains a local configuration file describing the proxy services available. This configuration file is installed in **/usr/X11R6.3/lib/X11/proxymngr/pmconfig** during the installation of **proxymngr**. The location of the configuration file can be overwritten using the **-config** command line flag.

Aside from lines starting with an exclamation point for comments, each line of the configuration file describes either an unmanaged or managed proxy service.

For unmanaged proxies, the format is:

<*service-name*> unmanaged <*proxy-address*>

service-name is the name of the unmanaged proxy service, and must not contain any spaces, for example XFWP. *service-name* is case insenstive.

proxy-address is the network address of the unmanaged proxy. The format of the address is specific to the *service-name*. For example, for the XFWP service, the *proxy-address* might be `firewall.x.org:100`.

If there is more than one entry in the config file with the same unmanaged *service-name*, the proxy manager will try to use the proxies in the order presented in the config file.

For managed proxies, the format is:

<*service-name*> managed <*command-to-start-proxy*>

service-name is the name of the managed proxy service, and must not contain any spaces, for example LBX. *service- name* is case insensitive.

command-to-start-proxy is the command executed by the proxy manager to start a new instance of the proxy. If *command- to-start-proxy* contains spaces, the complete command should be surrounded by single quotes. If desired, *command-to- start-proxy* can be used to start a proxy on a remote machine. The specifics of the remote execution method used to do this is not specified here.

Example: sample configuration file

```
! proxy manager config file
!
! Each line has the format:
!    <serviceName> managed <startCommand>
!        or
!    <serviceName> unmanaged <proxyAddress>
!
lbx managed /usr/X11R6.3/bin/lbxproxy
!
! substitute site-specific info
xfwp unmanaged firewall:4444
```

Proxy Manager Details

When the proxy manager gets a request from **xfindproxy** (or another similar client), its course of action will depend on the *service-name* in question.

For a managed proxy service, the proxy manager will find out if any of the already running proxies for this service can handle a new request. If not, the proxy manager will attempt to start up a new instance of the proxy (using the *command-to-start-proxy* found in the config file). If that fails, an error will be returned to the caller.

For an unmanaged proxy service, the proxy manager will look in the config file to find all unmanaged proxies for this service. If there is more than one entry in the config file with the same unmanaged *service-name*, the proxy manager will try to use the proxies in the order presented in the config file. If none of the unmanged proxies can satisfy the request, the proxy manager will timeout for a configurable amount of time (specified by **-timeout** or default of 10) and reattempt to find an unmanaged proxy willing to satisfy the request. The number of retries can be specified by the **-retries** argument, or a default of 3 will be used. If the retries fail, the proxy manager has no choice but to return an error to the caller (since the proxy manager can not start unmanaged proxy services).

prs Command (SCCS)

Purpose

Displays a Source Code Control System (SCCS) file.

Syntax

prs [**-a**] [**-d** *String*] [**-r** [*SID*] | [**-c** *Cutoff*]] [**-e** | **-l**] *File* ...

Description

The **prs** command first reads the specified files and then writes to standard output a part or all of a Source Code Control System (SCCS) file. If you specify a directory for the *File* parameter, the **prs** command performs the requested actions on all SCCS files (those with the **s.** prefix). If you specify a **-** (minus) for the *File* parameter, the **prs** command reads standard input and interprets each line as the name of an SCCS file. The **prs** command continues to read input until it reaches an end-of-file character.

Data Keywords

Data keywords specify the parts of an SCCS file to be retrieved and written to standard output. All parts of an SCCS file have an associated data keyword. There is no limit to the number of times a data keyword can be in a specified file.

The information that the **prs** command displays consists of user-supplied text and appropriate values (extracted from the SCCS file) substituted for the recognized data keywords in the order they are displayed in the specified file. The format of a data keyword value is either simple, in which the keyword substitution is direct, or multiline, in which the substitution is followed by a carriage return. Text consists of any characters other than recognized data keywords. Specify a tab character with **\t** (backslash, letter t) and a carriage return or new-line character with a **\n** (backslash, letter n). Remember to use the **\t** and

\n with an extra **** (backslash) to prevent the shell from interpreting the **** and passing only the letter **t** or **n** to the **prs** command as text.

The following table lists the keywords associated with information in the delta table of the SCCS file. All the keywords have a simple format unless otherwise indicated.

Delta Table Keywords

Keyword	Data Represented	Value
:R:	Release number	num
:L:	Level number	num
:B:	Branch number	num
:S:	Sequence number	num
:I:	SCCS ID string (SID)	:R::L::B::S:
:Dy:	Year delta created	YY
:Dm:	Month delta created	MM
:Dd:	Day delta created	DD
:D:	Date delta created	YY/MM/DD
:Th:	Hour delta created	HH
:Tm:	Minute delta created	MM
:Ts:	Second delta created	SS
:T:	Time delta created	HH/MM/SS
:DT:	Delta type	D or R

Item	Description	Value
:P:	User who created the delta	login name
:DS:	Delta sequence number	num
:DP:	Previous delta sequence number	num
:Dt:	Delta information	:DT::I::D::T::P::DS::DP:
:Dn:	Sequence numbers of deltas included	:DS: . . .
:Dx:	Sequence numbers of deltas excluded	:DS: . . .
:Dg:	Sequence numbers of deltas ignored	:DS: . . .
:DI:	Sequence numbers of deltas included,excluded, and ignored	:Dn:/:Dx:/:Dg:
:Li:	Lines inserted by delta	num

Item	Description	Value
:Ld:	Lines deleted by delta	num
:Lu:	Lines unchanged by delta	num
:DL:	Delta line statistics	:Li:/:Ld:/:Lu:
:MR: (multiline format)	MR numbers for delta	text
:C: (multiline format	Comments for delta	text

The following table lists the keywords associated with header flags in the SCCS file. All the keywords have a simple format unless otherwise indicated.

Header Flag Keywords

Keyword	Data Represented	Value
:Y:	Module type	text
:MF:	MR validation flag set	yes or no
:MP:	MR validation program name	text
:KF:	Keyword/error warning flag set	yes or no
:BF:	Branch flag set	yes or no
:J:	Joint edit flag set	yes or no
:LK:	Locked releases	:R: . . .
:Q:	User-defined keyword	text
:M:	Module name	text
:FB:	Floor boundary	:R:
:CB:	Ceiling boundary	:R:
:Ds:	Default SID	:I:
:ND:	Null Delta flag set	yes or no
:FL: (multiline format)	Header flag list	text

The following table lists the keywords associated with other parts of the SCCS file. All the keywords have a simple format unless otherwise indicated.

Other Keywords

Keyword	Data Represented	Value
:UN: (multiline format)	User names	text

Other Keywords *(continued)*

Keyword	Data Represented	Value
:FD: (multiline format)	Descriptive text	text
:BD: (multiline format)	Body of text	text
:GB: (multiline format)	Text in a g-file	text
:W:	A what string	:Z::M: \tab :I:
:A:	A what string	:Z::Y::M::I::Z:
:Z:	A what string delimiter	@(#)
:F:	SCCS file name	text
:PN:	SCCS file path name	text

Flags

Each flag or group of flags applies independently to each named file.

Item	Description
-a	Writes information for the specified deltas, whether or not they have been removed (see the **rmdel** command). If you do not specify the **-a** flag, the **prs** command supplies information only for the specified deltas that have not been removed.
-c *Cutoff*	Specifies a cutoff date and time for the **-e** and **-l** flags. Specify the *Cutoff* value in the following form: `YY[MM[DD[HH[MM[SS]]]]]` All omitted items default to their maximum values, so specifying -c8402 is the same as specifying -c840229235959. You can separate the fields with any non-numeric character. For example, you can specify `-c84/2/20,9:22:25` or `-c"84/2/20 9:22:25"` or `"-c84/2/20 9:22:25"`. The **-c** flag cannot be specified with the **-r** flag.
-d *String*	Specifies the data items to be displayed. The string consists of optional text and SCCS file-data keywords. The string may include MBCS (multibyte character set) characters. If the string contains spaces, you must enclose the string in quotation marks.
-e	Requests information for all deltas created earlier than and including the delta specified by the **-r** flag.
-l	Requests information for all deltas created later than and including the delta specified by the **-r** flag.
-r [*SID*]	Specifies the SCCS ID string (SID) of the delta for which the **prs** command will retrieve information. Do not enter a space between the **-r** flag and the optional SID parameter. If no SID is specified, the command retrieves the information for the SID of the highest numbered delta. The **-r** flag cannot be specified with the **-c** flag.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To display information on all deltas generated for SCCS file name **s.test.c** (including all deltas removed using the **rmdel** command), type:

```
prs -a s.test.c
```

2. To display user login name, the number of lines inserted by delta, and the number of lines deleted by delta for SID `1.2` of `s.test.c`, type:

```
prs -r1.2 -d":P:\n:Li:\n:Ld:" s.test.c
```

Files

Item	Description
/usr/bin/prs	Contains the **prs** command**.**

prtacct Command

Purpose

Formats and displays files in **tacct** format.

Syntax

/usr/sbin/acct/prtacct [**-X**] [**-W**] [**-f** *Fields*] [**-v**] *File* [**"***Heading***"**]

Description

The **prtacct** command formats and displays any total-accounting file; these files are in **tacct** format. You can enter this command to view any **tacct** file, such as the daily reports on connect time, process time, disk usage, and printer usage. To specify a title for the report with the *Heading* parameter, enclose the heading text in " " (quotation marks).

Flags

Item	Description
-f *Fields*	Selects fields to be displayed, using the field-selection mechanism of the **acctmerg** command.
-v	Produces verbose output in which more precise notation is used for floating-point numbers.
-W	Processes all available characters for each user name instead of truncating to the first 8 characters. This flag causes the **prtacct** command to expect to read in **tacctx** structures. It will then print out in the same column order, but it will allow long user names to misalign the columns. If the **-W** flag and the **-X** flag are used together, the **-X** takes precedence.

Item	Description
-X	Processes all available characters for each user name instead of truncating to the first 8 characters. This flag causes the **prtacct** command to expect to read in **tacctx** structures and print out the user name in the last column. If the **-W** flag and the **-X** flag are used together, the **-X** will take precedence.

Security

Access Control: This command should grant execute (x) access only to members of the **adm** group.

Examples

To format and display selected records from the total accounting file for connect-time accounting, you first must create a file upon which to execute the **prtacct** command. In this example, you create the **tacct** file using the **acctcon1** and **acctcon2** commands. Enter:

```
tail /var/adm/wtmp > wtmp.sav

acctcon1 -t < wtmp.sav | sort +1n +2 | acctcon2 > tacct
```

If you created this file previously to process connect-time accounting data, you do not need to create it again.

The next step uses the **prtacct** command with the **-f** flag to display the fields of data in the total-accounting file that you want to see. The text for a heading can be included in quotation marks. To view the login name, prime connect-time, and nonprime connect-time records, and include the heading, Connect-time Accounting, enter:

```
prtacct -f 2,11,12 tacct "Connect-time Accounting"
```

You can also use this command to format and display other total-accounting files, such as the daily reports on process time, disk usage, and printer usage.

Files

Item	Description
/usr/sbin/acct	The path to the accounting commands.
/var/adm/pacct	Current file for process accounting.
/var/adm/pacct*	Used if the **pacct** file gets too large.

prtconf Command

Purpose

Displays system configuration information.

Syntax

prtconf [**-c**] [**-k**] [**-L**] [**-m**] [**-s**] [**-v**]

Description

If you run the **prtconf** command without any flags, it displays the system model, machine serial, processor type, number of processors, processor clock speed, cpu type, total memory size, network information, filesystem information, paging space information, and devices information.

Flags

Item	Description
-c	Displays cpu type, for example, 32-bit or 64-bit.
-k	Display the kernel in use, for example, 32-bit or 64-bit.
-L	Displays LPAR partition number and partition name if this is an LPAR partition, otherwise returns "-1 NULL".
-m	Displays system memory.
-s	Displays processor clock speed in MegaHertz.
-v	Displays the VPD found in the Customized VPD object class for devices.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Examples

1. To display the system configuration information, enter:

```
prtconf
```

The system displays a message similar to the following:

```
System Model: IBM,7025-F50
Machine Serial Number: 1025778
Processor Type: PowerPC_604
Number Of Processors: 2
Processor Clock Speed: 332 MHz
CPU Type: 32-bit
Kernel Type: 32-bit
LPAR Info: -1 NULL
Memory Size: 512 MB
Good Memory Size: 512 MB
Firmware Version: IBM,L02113
Console Login: enable
Auto Restart: false
Full Core: false

Network Information
Host Name: vd01.austin.ibm.com
IP Address: 9.3.207.112
Sub Netmask: 255.255.255.128
Gateway: 9.3.207.1
Name Server: 9.3.199.2
Domain Name: austin.ibm.com
Paging Space Information


Total Paging Space: 512MB
Percent Used: 1%

Volume Groups Information
==============================================================================
rootvg:
PV_NAME           PV STATE          TOTAL PPs   FREE PPs    FREE DISTRIBUTION
hdisk0            active            537         394         107..43..29..107..108
==============================================================================

INSTALLED RESOURCE LIST
```

```
The following resources are installed on the machine.
+/- = Added or deleted from Resource List.
* = Diagnostic support not available.

Model Architecture: chrp
Model Implementation: Multiple Processor, PCI bus

+ sys0 00-00 System Object
+ sysplanar0 00-00 System Planar
+ mem0 00-00 Memory
etc.
```

2. To display the processor clock speed, enter:

```
prtconf -s
```

   The system displays a message similar to the following:

```
Processor Clock Speed: 332 MHz
```

3. To display the VPD for all physical devices in the Customized database, enter:

```
prtconf -v
```

   The system displays a message similar to the following:

```
INSTALLED RESOURCE LIST WITH VPD

The following resources are installed on your machine.

  Model Architecture: chrp
  Model Implementation: Uni-Processor, PCI bus

  sys0             P1-C1       System Object
  sysplanar0                   System Planar
  mem0                         Memory
  L2cache0                     L2 Cache
  proc0            P1-C1       Processor

        Device Specific.(YL).........P1-C1

  pci0             P1          PCI Bus

        Device Specific.(YL).........P1

  isa0             P1          ISA Bus

        Device Specific.(YL).........P1

  fda0             P1/D1       Standard I/O Diskette Adapter

        Device Specific.(YL).........P1/D1

  fd0              P1-D1       Diskette Drive
  siokma0          P1/K1       Keyboard/Mouse Adapter

        Device Specific.(YL).........P1/K1

  sioka0           P1-K1       Keyboard Adapter
  kbd0             P1-K1-Lkbd  PS/2 keyboard
  sioma0           P1-O1       Mouse Adapter
  mouse0           P1-O1-Lmouse3 button mouse
  siota0           P1/Q1       Tablet Adapter

        Device Specific.(YL).........P1/Q1

  paud0            P1/Q2       Ultimedia Integrated Audio

        Device Specific.(YL).........P1/Q2

  ppa0             P1/R1       CHRP IEEE1284 (ECP) Parallel Port Adapter

        Device Specific.(YL).........P1/R1

  sa0              P1/S1       Standard I/O Serial Port

        Device Specific.(YL).........P1/S1
```

```
  tty0             P1/S1-L0    Asynchronous Terminal
  sa1              P1/S2       Standard I/O Serial Port

        Device Specific.(YL).........P1/S2

  ent0             P1/E1       IBM 10/100 Mbps Ethernet PCI Adapter (23100020)

        Network Address.............0004AC2A0419
        Displayable Message.........PCI Ethernet Adapter (23100020)
        Device Specific.(YL).........P1/E1

  scsi0            P1/Z1       Wide/Fast-20 SCSI I/O Controller

        Device Specific.(YL).........P1/Z1

  cd0              P1/Z1-A3    SCSI Multimedia CD-ROM Drive (650 MB)

        Manufacturer................IBM
        Machine Type and Model......CDRM00203
        ROS Level and ID............1_00
        Device Specific.(Z0).........058002028F000018
        Part Number.................97H7608
        EC Level....................F15213
        FRU Number..................97H7610

  hdisk0           P1/Z1-A5    16 Bit SCSI Disk Drive (4500 MB)

        Manufacturer................IBM
        Machine Type and Model......DDRS-34560W
        FRU Number..................83H7105
        ROS Level and ID............53393847
        Serial Number...............RDHW5008
        EC Level....................F21433
        Part Number.................03L5256
        Device Specific.(Z0).........000002029F00003A
        Device Specific.(Z1).........00K0159S98G
        Device Specific.(Z2).........0933
        Device Specific.(Z3).........0299
        Device Specific.(Z4).........0001
        Device Specific.(Z5).........22
        Device Specific.(Z6).........F21390

  bl0              P1.1-I2/G1  GXT255P Graphics Adapter

      GXT255P 2D Graphics Adapter:
        EC Level....................E76756
        FRU Number..................93H6267
        Manufacture ID..............IBM053
        Part Number.................93H6266
        Serial Number...............88074164
        Version.....................RS6K
        Displayable Message.........GXT255P
        ROM Level.(alterable).......02
        Product Specific.(DD).......00
        Product Specific.(DG).......00
        Device Specific.(YL).........P1.1-I2/G1

  pci1             P1.1        PCI Bus

        Device Specific.(YL).........P1.1
```

4. To display the kernel type in use, type:

```
prtconf -k
```

   The system displays information for the kernel type as follows:

```
Kernel Type: 32-bit
```

5. To display memory, type:

```
prtconf -m
```

   The system displays memory, as follows:

```
Memory Size: 512 MB
```

Files

Item	Description
/usr/sbin/prtconf	Contains the **prtconf** command.

prtgblconfig Command

Purpose

The **prtgblconfig** command configures the global settings for the AIX printing subsystem.

Syntax

prtgblconfig [**-s** *name* = *value*] [**-r** *name*]

Description

The **prtgblconfig** command either sets a printing subsystem setting or resets it to a default value. Currently, this command is used to set the ERRMSGCONTROL setting. This setting affects the global printer message. This setting can be used to select one of the following options:

- ALLON (All messages turned on).
- LOGALL (All messages turned on, but logged to a log file).
- CRITON (Only the most critical error messages turned on).
- ALLOFF (All messages turned off).

Currently, the LOGALL option and the CRITON option is same as the ALLON option.

Flags

Item	Description
-s *name* = *value*	Specifies that the printing subsystem setting specified in the *name* parameter is set with the value specified in the *value* parameter.
-r *name*	Resets the printing subsystem setting specified in the *name* parameter to a default value.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To set the printing subsystem to ignore all of the messages generated by the printing subsystem, enter:

   ```
   prtgblconfig -s ERRMSGCONTROL=ALLOFF
   ```

2. To reset the error message control of the printing subsystem to a default value, enter:

   ```
   prtgblconfig -r ERRMSGCONTROL
   ```

   Note: Some messages generated by the printing subsystem cannot be ignored and are logged in the console log file. For more information about starting a print job, see the **qprt** command.

Files

Item	Description
/etc/prtglobalconfig	Contains the global configuration file.
/usr/sbin/qdaemon	Contains the qdaemon daemon.
/etc/qconfig	Contains the configuration file.
/etc/qconfig.bin	Contains the digested, binary version of the /etc/qconfig file.

ps Command

Purpose

Shows status of processes. This document describes the standard AIX **ps** command as well as the System V version of the **ps** command.

Syntax

X/Open Standards

ps [**-A**] [-M] [**-N**] [**-Z**] [**-a**] [**-d**] [**-e**] [**-f**] [**-k**] [**-l**] [**-F** *format*] [**-o** *Format*] [**-c** *Clist*] [**-G** *Glist*] [**-g** *Glist*] [**-m**] [**-n** *NameList*] [**-p** *Plist*] [**-P**] [**-t** *Tlist*] [**-U** *Ulist*] [**-u** *Ulist*] [**-T** *pid*] [**-L** *pidlist*] [-X] [**-@** [*WparName*]]

Berkeley Standards

ps [**a**] [**c**] [**e**] [**ew**] [**eww**] [**ewww**] [**g**] [**n**] [**w**] [**x**] [**l** | **s** | **u** | **v**] [**t** *tty*] [X] [*ProcessNumber*]

Description

The **ps** command writes the status of active processes and if the **-m** flag is given, displays the associated kernel threads to standard output. While the **-m** flag displays threads associated with processes using extra lines, you must use the **-o** flag with the **THREAD** field specifier to display extra thread-related columns.

Without flags, the **ps** command displays information about the current terminal. The **-f**, **-o**, **l**, **-l**, **s**, **u**, and **v** flags only determine how much information is provided about a process; they do not determine which processes are listed. The **l**, **s**, **u**, and **v** flags are mutually exclusive.

With the **-o** flag, the **ps** command examines memory or the paging area and determines what the command name and parameters were when the process was created. If the **ps** command cannot find this information, the command name stored in the kernel is displayed in square brackets.

The **COLUMNS** environment variable overrides the system-selected, horizontal screen size.

The command-line flags that accept a list of parameters (the **-o**, **-G**, **-g**, **-p**, **-t**, **-U**, and **-u** flags) are limited to 128 items. For example, the **-u** *Ulist* flag can specify no more than 128 users.

For cases in which the output of the **ps** command does not include workload partition (WPAR) names but does include Project IDs (**PROJECT**), User IDs (**UID** or **USER**), or Group IDs (**GID**) associated with a process running within a workload partition under the current operating environment, the IDs are preceded by a plus sign (+) to indicate the association with a workload partition. Each workload partition contains its own definition of users, groups, and project IDs that may be different from the IDs defined for the global environment. The **-@** option may be specified to include workload partition names in the output.

Note: The **ps** command does not show the decrease in the memory usage count when the application releases the memory. When the memory is released from the application, the memory is assigned to the per process memory freelist. The **ps** command accounts the memory that is released as the allocated memory for the application.

Depending on the flags used with the **ps** command, column headings are displayed above the information displayed to standard output. The headings are defined in the following list and the flags that cause these headings to be displayed are shown in parentheses :

ADDR
(**-l** and **l** flags) Contains the segment number of the process stack, if normal; if a kernel process, the address of the preprocess data area.

BND
(**-o THREAD** flag) The logical processor number of the processor to which the kernel thread is bound if any. For a process, this field is displayed if all its threads are bound to the same processor.

C
(**-f**, **l**, and **-l** flags) CPU utilization of process or thread, incremented each time the system clock ticks and the process or thread is found to be running. The value is decayed by the scheduler by dividing it by 2 once per second. For the `sched_other` policy, CPU utilization is used in determining process scheduling priority. Large values indicate a CPU intensive process and result in lower process priority, whereas small values indicate an I/O intensive process and result in a more favorable priority.

CMD
(**-f**, **-l**, and **l** flags) Contains the command name. Under the **-f** flag, the **ps** command tries to determine the current command name and arguments, both of which may be changed asynchronously by the process. These are then displayed. If this fails, the command name is written as it would appear without the **-f** option in square brackets.

COMMAND
(**s**, **u**, and **v**) Contains the command name. The full command name and its parameters are displayed with the **-f** flag.

F Field Table

Flags	Hexadecimal Value	Definition
SLOAD	0x00000001	Indicates that the process is operating in core memory.
SNOSWAP	0x00000002	Indicates that the process cannot be swapped out.
STRC	0x00000008	Indicates that the process is being traced.
SWTED	0x00000010	Indicates that the process stopped while being traced.
SFWTED	0x00000020	Indicates that the process stopped after a call to the **fork** subroutine, while being traced.
SEWTED	0x00000040	Indicates that the process stopped after a call to the **exec** subroutine, while being traced.
SLWTED	0x00000080	Indicates that the process stopped after a call to the **load** or **unload** subroutine, while being traced.
SFIXPRI	0x00000100	Indicates that the process has a fixed priority, ignoring the **pcpu** field descriptor.
SKPROC	0x00000200	Indicates a Kernel process.
SOMASK	0x00000400	Indicates restoration of the old mask after a signal is received.

F Field Table *(continued)*		
Flags	**Hexadecimal Value**	**Definition**
SWAKEONSIG	0x00000800	Indicates that the signal will abort the **sleep** subroutine. The contents must *not* be equal to those of the **PCATCH** flag. The contents of both **PCATCH** and **SWAKEONSIG** must be greater than those of **PMASK**.
SUSER	0x00001000	Indicates that the process is in user mode.
SLKDONE	0x00002000	Indicates that the process has done locks.
STRACING	0x00004000	Indicates that the process is a debugging process.
SMPTRACE	0x00008000	Indicates multi-process debugging.
SEXIT	0x00010000	Indicates that the process is exiting.
SSEL	0x00020000	Indicates that the processor is selecting: wakeup/waiting danger.
SORPHANPGRP	0x00040000	Indicates an orphaned process group.
SNOCNTLPROC	0x00080000	Indicates that the session leader relinquished the controlling terminal.
SPPNOCLDSTOP	0x00100000	Indicates that the **SIGHLD** signal is *not* sent to the parent process when a child stops.
SEXECED	0x00200000	Indicates that process has been run.
SJOBSESS	0x00400000	Indicates that job control was used in the current session.
SJOBOFF	0x00800000	Indicates that the process is free from job control.
PSIGDELIVERY	0x01000000	Indicates that the process is used by the program-check handler.
SRMSHM	0x02000000	Indicates that the process removed shared memory during a call to the **exit** subroutine.
SSLOTFREE	0x04000000	Indicates that the process slot is free.
SNOMSG	0x08000000	Indicates that there are no more **uprintf** subroutine messages.

WPAR

(**-@** flag) Contains the workload partition name. Under the **-@** flag, the **ps** command displays the name of the workload partition in which the process is running. Specify the **-@** flag with the *wparname* parameter to display the process information.

DPGSZ

(**Z** flag) The data page size of the process.

F

(**-l** and **l** flags) Some of the more important F field flags (hexadecimal and additive) associated with processes and threads are listed in the following table:

F Field Table		
Flags	**Hexadecimal Value**	**Definition**
SLOAD	0x00000001	Indicates that the process is operating in core memory.
SNOSWAP	0x00000002	Indicates that the process cannot be swapped out.
STRC	0x00000008	Indicates that the process is being traced.
SKPROC	0x00000200	Indicates a kernel process.
SEXIT	0x00010000	Indicates that the process is exiting.
SLPDATA	0x00020000	Indicates that the process uses large pages.
SEXECED	0x00200000	Indicates that the process has been run.
SEXECING	0x01000000	Indicates that the process is execing (performing an exec).
SPSEARLYALLOC	0x04000000	Indicates that paging space for this process is allocated early.
TKTHREAD	0x00001000	Indicates that the thread is a kernel-only thread.

Note: You can see the definitions of all process and thread flags by consulting the **p_flags** and **t_flags** fields in the **/usr/include/sys/proc.h** and **/usr/include/sys/thread.h** files respectively.

LIM
(**v** flag) The soft limit on memory used, specified through a call to the **setrlimit** subroutine. If the limit has not been specified, then `xx` is displayed. If the limit is set to the system limit (unlimited), a value of `UNLIM` is displayed.

NI
(**-l** and **l** flags) The nice value; used in calculating priority for the `sched other` policy.

PID
(all flags) The process ID of the process.

PGIN
(**v** flag) The number of disk I/Os resulting from references by the process to pages not loaded in core.

PPID
(**-f**, **l**, and **-l** flags) The process ID of the parent process.

PRI
(**-l** and **l** flags) The priority of the process or kernel thread; higher numbers mean lower priority.

PROJECT
(**-P** flag) Project name assigned to the process. Under the current operating environment, the **PROJECT** and **USER** fields are not translated to names for processes running within a workload partition. The **-U** and **-u** flags only apply to the current operating environment, unless the **-@** flag is included with a specific workload partition name. If the **-@** flag is used to specify a workload partition other than the current operating environment, and the **-U** and **-u** flags are specified, the list of user IDs must be numeric.

RSS
(**v** flag) The real-memory (resident set) size of the process (in 1 KB units).

S
(**-l** and **l** flags) The state of the process or kernel thread :

For processes:

O
Nonexistent

A
Active

W
Swapped

I
Idle (waiting for startup)

Z
Canceled

T
Stopped

For kernel threads:

O
Nonexistent

R
Running

S
Sleeping

W
Swapped

Z
Canceled

T
Stopped

SC
(**-o THREAD** flag) The suspend count of the process or kernel thread. For a process, the suspend count is defined as the sum of the kernel threads suspend counts.

SCH
(**-o THREAD, sched** flag) The scheduling policy for a kernel thread. The policies `sched_other`, `sched_fifo`, and `sched_rr` are respectively displayed using: `0`, `1`, `2`. The scheduling policies is displayed only when a **sched** flag is specified.

SIZE
(**v** flag) The virtual size of the data section of the process (in 1 KB units).

SHMPGSZ
(**Z** flag) The shared memory page size of the process.

SPGSZ
(**Z** flag) The stack page size of the process.

SSIZ
(**s** flag) The size of the kernel stack. This value is always 0 (zero) for a multi-threaded process.

STAT
(**s**, **u**, and **v** flags) Contains the state of the process:

0
Nonexistent

A
Active

I
Intermediate

Z
Canceled

T
Stopped

K
Available kernel process

STIME
(**-f** and **u** flags) The starting time of the process. The **LANG** environment variables control the appearance of this field.

SUBPROJ
(**-P** flag) Subproject Identifier assigned to the process.

SZ
(**-l** and **l** flags) The size in 1 KB units of the core image of the process.

THCNT
(**-o thcount** flag) The number of kernel threads owned by the process.

TID
(**-o THREAD** flag) The thread ID of the kernel thread.

TIME
(all flags) The total runtime for the process. The time is displayed in the format of *mm:ss* or *mmmm:ss* if the runtime reaches 100 minutes, which is different from the displayed format if you use the **-o time** flag.

TPGSZ
(**Z** flag) The text page size of the process.

TRS
(**v** flag) The size of resident-set (real memory) of text.

TSIZ
(**v** flag) The size of text (shared-program) image.

TTY
(all flags) The controlling terminal for the process:

-
The process is not associated with a terminal.

?
Unknown.

Number
The TTY number. For example, the entry 2 indicates TTY2.

UID
(**-f**, **-l**, and **l** flags) The user ID of the process owner. The login name is printed under the **-f** flag.

USER
(**u** flag) The login name of the process owner. Under the current operating environment, the **PROJECT** and **USER** fields are not translated to names for processes running within a workload partition.

WCHAN
(**-l** flag) The event for which the process or kernel thread is waiting or sleeping. For a kernel thread, this field is blank if the kernel thread is running. For a process, the wait channel is defined as the wait channel of the sleeping kernel thread if only one kernel thread is sleeping; otherwise a star is displayed.

WCHAN
(**l** flag) The event on which the process is waiting (an address in the system). A symbol that classifies the address is selected, unless a numerical output is requested.

%CPU
(**u** and **v** flags) The percentage of time the process has used the CPU since the process started. This value is computed by dividing the time the process uses the CPU by the elapsed time of the

process. In a multi-processor environment, the value is further divided by the number of available CPUs because several threads in the same process can run on different CPUs at the same time. (Because the time base over which this data is computed varies, the sum of all **%CPU** fields can exceed 100%.)

%MEM
(**u** and **v** flags) The percentage of real memory used by this process. The %MEM value tends to exaggerate the cost of a process that is sharing program text with other processes. It does not account for times when multiple copies of a program are run and a copy of the program text is shared by all instances. The size of the text section is accounted for in every instance of the program. This means that if several copies of a program are run, the total %MEM value of all processes could exceed 100%.

A process that has exited and has a parent that has not yet waited for the process is marked <defunct>. A process that is blocked trying to exit is marked <exiting>. The **ps** command attempts to determine the file name and arguments given when the process was created by memory or by the swap area.

Notes:

1. The process can change while the **ps** command is running. Some data displayed for defunct processes is irrelevant.
2. The **ps** program examines the memory to retrieve the file name and arguments used when the process was created. However, a process can destroy information, making this method of retrieving file name and arguments unreliable.
3. The **ps** program searches the local resources for users and group information.

Flags

The following flags are preceded by a **-** (minus sign):

Item	Description
-A	Writes to standard output information about all processes.
-a	Writes to standard output information about all processes, except the session leaders and processes not associated with a terminal.
-c *Clist*	Displays only information about processes assigned to the workload management classes listed in the *Clist* variable. The *Clist* variable is either a comma-separated list of class names or a list of class names enclosed in double quotation marks (" "), which is separated from one another by a comma or by one or more spaces, or both.
-d	Writes information to standard output about all processes, except the session leaders.
-e	Writes information to standard output about all processes, except kernel processes.
-F *Format*	Same as the **-o** *Format*
-f	Generates a full listing.
-G *Glist*	Writes information to standard output only about processes that are in the effective groups listed for the *Glist* variable. The *Glist* variable is either a comma-separated list of effective group identifiers, or a list of effective group identifiers enclosed in double quotation marks (" ") and separated from one another by a comma or by one or more spaces.
-g *Glist*	Writes information to standard output only about processes that are in the process groups listed for the *Glist* variable. The *Glist* variable is either a comma-separated list of process group identifiers or a list of process group identifiers enclosed in double quotation marks (" ") and separated from one another by a comma or by one or more spaces.
-k	Lists kernel processes.
-l	Generates a long listing. Also see the **l** flag.
-L *pidlist*	Generates a list of descendants of each and every pid that has been passed to it in the *pidlist* variable. The *pidlist* variable is a list of comma-separated process IDs. The list of descendants from all the given pid is printed in the order in which they appear in the process table.
-M	Lists all 64 bit processes.
-m	Lists kernel threads as well as processes. Output lines for processes are followed by an additional output line for each kernel thread. This flag does not display thread-specific fields (**bnd**, **scount**, **sched**, **thcount**, and **tid**), unless the appropriate **-o** *Format* flag is specified.
-N	Gathers no thread statistics. With this flag, **ps** reports those statistics that can be obtained by not traversing through the threads chain for the process.

Item	Description
-n *NameList*	Specifies an alternative system name-list file in place of the default. The operating system does not use the **-n** flag because information is supplied directly to the kernel.
-o *Format*	Displays information in the format specified by the *Format* variable. Multiple field specifiers can be specified for the *Format* variable. The *Format* variable is either a comma-separated list of field specifiers or a list of field specifiers enclosed within a set of " " (double-quotation marks) and separated from one another by a comma or by one or more spaces, or both. Each field specifier has a default header. The default header can be overridden by appending an **=** (equal sign) followed by the user-defined text for the header. The fields are written in the order specified on the command-line in column format. The field widths are specified by the system to be at least as wide as the default or user-defined header text. If the header text is null (such as if **-o user=** is specified), the field width is at least as wide as the default header text. If all header fields are null, no header line is written. The following field specifiers are recognized by the system: **args** Indicates the full command name being executed. All command-line arguments are included, though truncation may occur. The default header for this field is **COMMAND**. **bnd** Indicates to which (if any) processor a process or kernel thread is bound. The default header for this field is **BND**. **class** Indicates the workload management class assigned to the process or thread. The default header for this field is **CLASS**. **comm** Indicates the short name of the command being executed. Command-line arguments are not included. The default header for this field is **COMMAND**. **cpu** Determines process scheduling priority. CPU utilization of a process or thread, incremented each time the system clock ticks and the process or thread is found to be running. The value is decayed by the scheduler by dividing it by 2 once per second. For the sched_other policy, Large values indicate a CPU intensive process and result in lower process priority whereas small values indicate an I/O intensive process and result in a more favorable priority. **dpgsz** Indicates the data page size of a process. **etime** Indicates the elapsed time since the process started. The elapsed time is displayed in the following format: [[*dd***-**]*hh***:**]*mm***:***ss* where *dd* specifies the number of days, *hh* specifies the number of hours, *mm* specifies the number of minutes, and *ss* specifies the number of seconds. The default header for this field is **ELAPSED**. **group** Indicates the effective group ID of the process. The textual group ID is displayed. If the textual group ID cannot be obtained, a decimal representation is used. The default header for this field is **GROUP**. **nice** Indicates the decimal value of the process nice value. The default header for this field is **NI**.

Item	Description
-o *Continued*	**pcpu** Indicates the ratio of CPU time used to CPU time available, expressed as a percentage. The default header for this field is **%CPU**. **pgid** Indicates the decimal value of the process group ID. The default header for this field is **PGID**. **pid** Indicates the decimal value of the process ID. The default header for this field is **PID**. **ppid** Indicates the decimal value of the parent process ID. The default header for this field is **PPID**. **rgroup** Indicates the real group ID of the process. The textual group ID is displayed. If the textual group ID cannot be obtained, a decimal representation is used. The default header for this field is **RGROUP**. **ruser** Indicates the real user ID of the process. The textual user ID is displayed. If the textual user ID cannot be obtained, a decimal representation is used. The default header for this field is **RUSER**. **scount** Indicates the suspend count for a kernel thread. The default header for this field is **SC**. **sched** Indicates the scheduling policy for a kernel thread. The default header for this field is **SCH**. **shmpgsz** Indicates the shared memory page size of a process. **spgsz** Indicates the stack page size of a process. **tag** Indicates the Workload Manager application tag. The default header for this field is **TAG**. The tag is a character string up to 30 characters long and may be truncated when displayed by **ps**. For processes that do not set their tag, this field displays as a **-** (hyphen). **tcpu** Total CPU time. Indicates the total accumulated CPU time for a single process. The command displays the information when WLM is running either in active or passive mode else, this field displays as a - (hyphen). The default header for this field is **TCPU**. **tctime** Total connect time. Indicates the total amount of time that a login session can be active. This is meaningful only in the case of session leader processes. The default header for this field is **TCTIME**. **tdiskio** Total disk I/O. Indicates the total accumulated blocks of disk I/O for a single process. The default header for this field is **TDISKIO**. **tpgsz** Indicates the text page size of a process. **vmsize** Indicates the WLM virtual memory limits. When this is used, a new header, VMSIZ is displayed. VMSIZ displays the virtual memory used by the process. This value is displayed in 1 MB units. **thcount** Indicates the number of kernel threads owned by the process. The default header for this field is **THCNT**.

Item	Description
-o *Continued*	**THREAD** Indicates the following fields: • User name (the **uname** field) • Process and parent process IDs for processes (the **pid** and **ppid** fields) • Kernel thread ID for threads (the **tid** field) • The state of the process or kernel thread (the **S** field) • The CPU utilization of the process or kernel thread (the **C** field) • The priority of the process or kernel thread (the **PRI** field) • The suspend count of the process or kernel thread (the **scount** field) • The wait channel of the process or kernel thread (the **WCHAN** field) • The flags of the process or kernel thread (the **F** field) • The controlling terminal of the process (the **tty** field) • The CPU to which the process or kernel thread is bound (the **bnd** field) • The command being executed by the process (the **comm** field). Threads are not displayed with the **-o THREAD** flag, unless the **-m** flag is also specified. **Note:** The `ps -o THREAD` flag does not print the scheduler policies. The scheduling policies are displayed only when a **sched** flag is specified. **tid** Indicates the thread ID of a kernel thread. The default header for this field is **TID**. **time** Indicates the cumulative CPU time since the process started. The time is displayed in the following format: [*dd***-**] *hh***:***mm***:***ss* where *dd* specifies the number of days, *hh* specifies the number of hours, *mm* specifies the number of minutes, and *ss* specifies the number of seconds. The default header for this field is **TIME**. **tty** Indicates the controlling terminal name of the process. The default header for this field is **TT**. **user** Indicates the effective user ID of the process. The textual user ID is displayed. If the textual user ID cannot be obtained, a decimal representation is used. The default header for this field is **USER**. **vsz** Indicates, as a decimal integer, the size in kilobytes of the process in virtual memory. The default header for this field is **VSZ**. Otherwise, multiple fields in a specified format can be displayed by the *Format* variable, including field descriptors. If field descriptors are used in the *Format* variable, it must be enclosed in double quotation marks (" "). The following table shows how field descriptors correspond to field specifiers: `Field Descriptors  Field Specifiers  Default Headers` `%a  args  COMMAND` `%c  comm  COMMAND` `%t  etime  ELAPSED` `%D  dpgsz  DPGSZ` `%G  group  GROUP` `%n  nice  NI` `%C  pcpu  %CPU` `%r  pgid  PGID` `%p  pid  PID` `%P  ppid  PPID` `%g  rgroup  RGROUP` `%u  ruser  RUSER` `%S  spgsz  SPGSZ` `%x  time  TIME` `%T  tpgsz  TPGSZ gd` `%y  tty  TTY` `%U  user  USER` `%z  vsz  VSZ` Each field specifier has a default header. The default header can be overridden by appending an equal sign (=) followed by the user-defined text for the header. The fields are written in the order specified on the command-line in column format. The field widths are specified by the system to be at least as wide as the default or user-defined header text. If the header text is null (for example, `-o user=` is specified), the field width is at least as wide as the default header text. If all header fields are null, no header line is written.

Item	Description
	Following is the mapping between the default headers and various field specifiers. Every entry in the Default Header column can be overridden by appending an equal sign (=) to the corresponding entry in the Field specifier followed by the user-defined text for the header.

```
Default Header                Field specifier

ARGS                         "args"
COMM                         "comm"
COMM                         "command"
COMM                         "ucomm"
F_ETIME                      "etime"
GROUP                        "group"
GROUP                        "gname"
GID                          "gid"
NICE                         "nice"
PRI                          "pri"
NICE                         "ni"
PCPU                         "pcpu"
PMEM                         "pmem"
PGID                         "pgid"
PID                          "pid"
PPID                         "ppid"
RGROUP                       "rgroup"
RGROUP                       "rgname"
RGID                         "rgid"
RUSER                        "ruser"
RUSER                        "runame"
RUID                         "ruid"
TIME                         "time"
TIME                         "cputime"
TTY                          "tty"
TTY                          "tt"
TTY                          "tname"
TTY                          "longtname"
USER                         "user"
USER                         "uname"
UID                          "uid"
LOGNAME                      "logname"
STIME                        "start"
VSZ                          "vsz"
VSZ                          "vsize"
RSS                          "rssize"
FLAG                         "flag"
STATUS                       "status"
CP                           "cp"
PAGEIN                       "pagein"
WCHAN                        "wchan"
NWCHAN                       "nwchan"
ST                           "st"
TID                          "tid"
SCOUNT                       "scount"
BIND                         "bnd"
SCHED                        "sched"
THCOUNT                       "thcount"
TAG                          "tag"
CLASS                        "class"
TCPU                         "tcpu"
TDISKIO                      "tdiskio"
TCTIME                       "tctime"
MACLAB                       "mac"
```

-p *Plist*	Displays only information about processes with the process numbers specified for the *Plist* variable. The *Plist* variable is either a comma-separated list of process ID numbers or a list of process ID numbers enclosed in double quotation marks (" ") and separated from one another by a comma or by one or more spaces, or both.
-P	Displays the Project name, Project origin, and subproject identifier for the project. If the stick bit is set for the process, the project name is preceded by an asterisk (*) character. The `Project origin` field designates the currently loaded project repository (LOCAL or LDAP).
-t *Tlist*	Displays only information about processes associated with the controlling ttys listed in the *Tlist* variable. The *Tlist* variable is either a comma-separated list of tty identifiers or a list of tty identifiers enclosed in double quotation marks (" ") and separated from one another by a comma or by one or more spaces, or both.
-T *pid*	Displays the process hierarchy rooted at a given pid in a tree format using ASCII art. This flag can be used in combination with the **-f**, **-F**, **-o**, and **-l** flags.
-u *Ulist*	This flag is equivalent to the **-U** *Ulist* flag. The **-u** flag only applies to the current operating environment unless the **-@** flag is also specified. If the **-@** flag is used to specify a workload partition other than the current operating environment and the **-u** flag is specified, the list of user IDs must be numeric.
-U *Ulist*	Displays only information about processes with the user ID numbers or login names specified for the *Ulist* variable. The *Ulist* variable is either a comma-separated list of user IDs or a list of user IDs enclosed in double quotation marks (" ") and separated from one another by a comma and one or more spaces. The **-U** flag only applies to the current operating environment unless the **-@** flag is also specified. If the **-@** flag is used to specify a workload partition other than the current operating environment and the **-U** flag is specified, the list of user IDs must be numeric. In the listing, the **ps** command displays the numerical user ID unless the **-f** flag is used; then the command displays the login name. This flag is equivalent to the **-u** *Ulist* flag. See also the **u** flag.

Item	Description
-X	Prints all available characters of each user/group name instead of truncating to the first eight characters.
-Z	Displays the page size settings of processes. **DPGSZ** Indicates the data page size of a process. **SHMPGSZ** Indicates the shared memory page size the process allocates. **SPGSZ** Indicates the stack page size of a process. **TPGSZ** Indicates the text page size of a process.
-@ [*WparName*]	Displays the process information that is associated with the workload partition *WparName*. If you do not specify the *WparName* parameter, the process information for all workload partitions is displayed. Workload partition information is displayed for all processes. You must specify other flags to the **ps** command to determine which process information to be displayed.

Options

The following options are not preceded by a minus sign (-):

Item	Description
a	Displays information about all processes with terminals (ordinarily only the own processes of the user are displayed).
c	Displays the command name, as stored internally in the system for purposes of accounting, rather than the command parameters, which are kept in the process address space.
e	Displays the environment as well as the parameters to the command, up to a limit of 80 characters.
ew	Wraps the display from the **e** flag one extra line.
eww	Wraps the display from the **e** flag and displays the **ENV** list until the flag reaches the **LINE_MAX** value.
ewww	Wraps the display from the **e** flag and displays the **ENV** list until the flag reaches the **INT_MAX** value.
g	Displays all processes.
l	Displays a long listing having the **F**, **S**, **UID**, **PID**, **PPID**, **C**, **PRI**, **NI**, **ADDR**, **SZ**, **PSS**, **WCHAN**, **TTY**, **TIME**, and **CMD** fields.
n	Displays numerical output. In a long listing, the **WCHAN** field is printed numerically rather than symbolically. In a user listing, the **USER** field is replaced by a **UID** field.
s	Displays the size (SSIZ) of the kernel stack of each process (for use by system maintainers) in the basic output format. This value is always 0 (zero) for a multi-threaded process.
t *tty*	Displays processes whose controlling tty is the value of the *tty* variable, which should be specified as printed by the **ps** command; that is, `0` for terminal **/dev/tty/0**, `lft0` for **/dev/lft0**, and `pts/2` for **/dev/pts/2**.
u	Displays user-oriented output. This includes the **USER**, **PID**, **%CPU**, **%MEM**, **SZ**, **RSS**, **TTY**, **STAT**, **STIME**, **TIME**, and **COMMAND** fields.
v	Displays the **PGIN**, **SIZE**, **RSS**, **LIM**, **TSIZ**, **TRS**, **%CPU**, **%MEM** fields.
w	Specifies a wide-column format for output (132 columns rather than 80). If repeated, (for example, `ww`), uses arbitrarily wide output. This information is used to decide how much of long commands to print.
x	Displays processes without a controlling terminal in addition to processes with a controlling terminal.
X	Prints the full user name or group name. The name is not truncated.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display all processes, type:

```
ps -e -f
```

   To display all processes with expanded user name, type:

```
ps -X -e -f
```

2. To list processes owned by specific users, type:

```
ps -f -l -ujim,jane,su
```

3. To list processes that are associated with the **/dev/console** and **/dev/tty1** ttys, type:

```
ps -t console,tty/1
```

4. To list processes not associated with a terminal, type:

```
ps -t -
```

5. To display a specified format with field specifiers, type:

```
ps -o ruser,pid,ppid=parent,args
```

   The output is:

```
RUSER   PID     parent  COMMAND
helene  34      12      ps -o ruser,pid,ppid=parent,args
```

6. To display a specified format with field descriptors, type:

```
ps -o "< %u > %p %y : %a"
```

   The output is:

```
< RUSER  >      PID     TT :    COMMAND
< helene >      34      pts/3 : ps -o < %u > %p %y : %a
```

7. To display information about processes and kernel threads controlled by the current terminal, type:

```
ps -lm
```

   The output is like:

```
     F S UID  PID PPID  C PRI NI ADDR  SZ WCHAN   TTY  TIME  CMD
240003 A  26 8984 7190  1  60 20 2974 312       pts/1  0:00  -ksh
   400 S   -    -    -  1  60  -    -   -           -     -  -
200005 A  26 9256 8984 15  67 20 18ed 164       pts/1  0:00  ps
     0 R   -    -    - 15  67  -    -   -           -     -  -
```

8. To display information about all processes and kernel threads, type:

```
ps -emo THREAD
```

   The output is like:

```
USER   PID  PPID  TID S  C PRI SC   WCHAN   FLAG   TTY BND  CMD
jane  1716 19292    - A 10  60  1       * 260801 pts/7   -  biod
   -     -     - 4863 S  0  60  0 599e9d8   8400     -   -  -
   -     -     - 5537 R 10  60  1 5999e18   2420     -   3  -
luke 19292 18524    - A  0  60  0 586ad84 200001 pts/7   -  -ksh
   -     -     - 7617 S  0  60  0 586ad84    400     -   -  -
luke 25864 31168    - A 11  65  0       - 200001 pts/7   -  -
   -     -     - 8993 R 11  65  0       -      0     -   -  -
```

9. To list all the 64-bit processes, type:

```
ps -M
```

10. To display the project assignment details for the processes, type:

```
ps -P
```

11. To display the page size settings of the processes, type:

```
ps -Z
```

The output is like:

```
          PID    TTY TIME DPGSZ SPGSZ TPGSZ SHMPGSZ CMD
        41856 pts/15 0:00    4K    4K    4K    64K    ps
        84516 pts/15 0:00    4K    4K    4K    64K    ksh
```

Files

Item	Description
/**usr**/**bin**/**ps**	Contains the **ps** command.

Using the ps command in *Performance management*.

System V ps command

Syntax (System V)

/usr/sysv/bin/ps [**-a**] [**-A**] [**-c**] [**-d**] [**-e**] [**-f**] [**-j**] [**-l**] [**-L**] [**-P**] [**-y**] [**-g** *pgrplist*] [**-o** *format*] [**-p** *proclist*] [**-s** *sidlist*] [**-t** *termlist*] [{ **-u** | **-U** } *uidlist*] [**-G** *grplist*] [-X]

Description (System V)

The **ps** command prints information about active processes. Without flags, **ps** prints information about processes associated with the controlling terminal. The output contains the process ID, terminal identifier, cumulative runtime, and the command name. The information displayed with flags varies accordingly.

Output

Depending on the flags used with the **ps** command, column headings vary for the information displayed. The headings are defined in the following list (flags that cause these headings to appear are shown in parentheses):

F (-l)
Flags (hexadecimal and additive) associated with the process, or the thread if the **-L** option is specified. Some of the more important F field flags (hexadecimal and additive) associated with processes and threads are shown below:

F Field Table		
Flags	**Hexadecimal Value**	**Definition**
SLOAD	0x00000001	Indicates that the process is operating in core memory.
SNOSWAP	0x00000002	Indicates that the process cannot be swapped out.
STRC	0x00000008	Indicates that the process is being traced.
SKPROC	0x00000200	Indicates a Kernel process.
SEXIT	0x00010000	Indicates that the process is exiting.
SEXECED	0x00200000	Indicates that process has been run.
SEXECING	0x01000000	Indicates that the process is execing (performing an exec).
TKTHREAD	0x00001000	Indicates that the thread is a kernel only thread.

Note: You can see the definitions of all process and thread flags by referring to the **p_flags** and **t_flags** fields in the **/usr/include/sys/proc.h** and **/usr/include/sys/thread.h** files respectively.

S (-l)
The state of the process or kernel thread :

For processes:

O
Nonexistent

A
Active

W
Swapped

I
Idle

Z
Canceled

T
Stopped

For kernel threads:

O
Nonexistent

R
Running

S
Sleeping

W
Swapped

Z
Canceled

T
Stopped

UID (-f,- l)
The user ID number of the process (the login name is printed under the -f option).

PID (all)
The process ID of the process.

PPID (-f,-l)
The process ID of the parent process.

CLS (-c)
Scheduling class for the process. Printed only when the **-c** flag is used.

NI (-l)
The nice value of the process used in calculating priority for the **sched_other** policy.

PRI (-c, -l)
The priority of the process or kernel thread. Higher numbers mean lower priority.

ADDR (-l)
Contains the segment number of the process stack, if normal; if a kernel process, the address of the preprocess data area.

SZ (-l)
The size in pages of the core image of the process.

WCHAN(-l)
The event for which the process or kernel thread is waiting or sleeping. For a kernel thread, this field is blank if the kernel thread is running. For a process, the wait channel is defined as the wait channel of the sleeping kernel thread if only one kernel thread is sleeping; otherwise a star is displayed.

STIME (-f,-u)
The starting time of the process. The **LANG** environment variables control the appearance of this field.

TTY (all)
The controlling terminal for the process:

-
The process is not associated with a terminal.

?
Unknown

TIME (all)
The total runtime for the process. The time is displayed in the format of *mm:ss* or *mmmm:ss* if the runtime reaches 100 minutes, which is different from the displayed format if you use the **-o time** flag.

LTIME (-L)
The runtime for an individual LWP.

CMD (all)
Contains the command name. The full command name and its parameters are displayed with the **-f** flag.

LWP (-L)
The tid of the kernel thread.

NLWP(-Lf)
The number of kernel threads in the process.

PSR (-P)
The logical processor number of the processor to which the kernel thread is bound (if any). For a process, this field is shown if all its threads are bound to the same processor.

RSS (-ly)
The real memory (resident set) size of the process (in 1 KB units).

Format

The following list describes the field specifiers recognized by the system. These field specifiers can be used with the **-o** flag to specify the format for the output of the **ps** command.

The field specifiers recognized by the system are:

addr
Indicates the segment number of the process stack, if normal; if a kernel process, the address of the preprocess data area. The default header for this field is **ADDR**.

args
Indicates the full command name being executed. All command-line arguments are included, though truncation may occur. The default header for this field is **COMMAND**.

c
CPU utilization of process or thread, incremented each time the system clock ticks and the process or thread is found to be running. The value is decayed by the scheduler by dividing it by 2 once per second. For the **sched_other** policy, CPU utilization is used in determining process scheduling priority. Large values indicate a CPU intensive process and result in lower process priority whereas small values indicate an I/O intensive process and result in a more favorable priority. The default header for this field is **C**.

class
Indicates the scheduling policy for a kernel thread. The policies are sched_other, sched_fifo and sched_rr. The default header for this field is **CLS**.

comm
Indicates the short name of the command being executed. Command-line arguments are not included. The default header for this field is **COMMAND**.

etime
Indicates the elapsed time since the process started. The elapsed time is displayed in the format

```
[[ dd -] hh: ]mm :ss
```

where *dd* specifies the number of days, *hh* specifies the number of hours, *mm* specifies the number of minutes, and *ss* specifies the number of seconds.

The default header for this field is **ELAPSED**.

f
Indicates flags (hexadecimal and additive) associated with the process. The default header for this field is **COMMAND**.

fname
Indicates the first 8 bytes of the base name of the process's executable file. The default header for this field is **COMMAND**.

gid
Indicates the effective group ID number of the process as a decimal integer. The default header for this field is **GID**. The login name is printed under the **-f** option.

group
Indicates the effective group ID of the process. The textual group ID is displayed. If the textual group ID cannot be obtained, a decimal representation is used. The default header for this field is **GROUP**.

lwp
Indicates the thread ID of the kernel thread. The default header for this field is **TID**.

nice
Indicates the decimal value of the process nice value. The default header for this field is **NI**.

nlwp
Indicates the number of kernel threads owned by the process. The default header for this field is **THCNT**.

pcpu
Indicates the ratio of CPU time used to CPU time available, expressed as a percentage. The default header for this field is **%CPU**.

pgid
Indicates the decimal value of the process group ID. The default header for this field is **PGID**.

pid
Indicates the decimal value of the process ID. The default header for this field is **PID**.

pmem
Indicates the percentage of real memory used by this process. The default header for this field is **%MEM**.

ppid
Indicates the decimal value of the parent process ID. The default header for this field is **PPID**.

pri
Indicates the priority of the process or kernel thread ; higher numbers mean lower priority. The default header for this field is **PRI**.

psr
Indicates the logical processor number of the processor to which the kernel thread is bound (if any). The default header for this field is **PSR**.

rgid
Indicates the real group ID number of the process as a decimal integer. The default header for this field is **RGID**.

rgroup
Indicates the real group ID of the process. The textual group ID is displayed. If the textual group ID cannot be obtained, a decimal representation is used. The default header for this field is **RGROUP**.

rss
Indicates the real memory (resident set) size of the process (in 1 KB units). The default header for this field is **RSS**.

ruid
Indicates the real user ID number of the process as a decimal integer. The default header for this field is **RUID**.

ruser
Indicates the real user ID of the process. The textual user ID is displayed. If the textual user ID cannot be obtained, a decimal representation is used. The default header for this field is **RUSER**.

s
Indicates the state of the process. The default header for this field is **S**.

sid
Indicates the process ID of the session leader. The default header for this field is **SID**.

stime
Indicates the starting time of the process. The LANG environment variables control the appearance of this field. The default header for this field is **STIME**.

time
Indicates the cumulative CPU time since the process started. The time is displayed in the same format as in **etime**. The default header for this field is **TIME**.

tty
Indicates the controlling terminal name of the process. The default header for this field is **TT**.

uid
Indicates the effective user ID number of the process as a decimal integer. The default header for this field is **UID**.

user
Indicates the effective user ID of the process. The textual user ID is displayed. If the textual user ID cannot be obtained, a decimal representation is used. The default header for this field is **USER**.

vsz
Indicates, as a decimal integer, the size in kilobytes of the core image of the process. The default header for this field is **VSZ**.

wchan
Indicates the event for which the process or kernel thread is waiting or sleeping. For a kernel thread, this field is blank if the kernel thread is running. For a process, the wait channel is defined as the wait channel of the sleeping kernel thread if only one kernel thread is sleeping; otherwise a star is displayed.

The default header for this field is **WCHAN**.

Flags (System V)

Some flags accept lists as arguments. Items in a list can be either separated by commas or else enclosed in double quotes and separated by commas or spaces. Values for *proclist* and *pgrplist* must be numeric.

Item	Description
-a	Writes to standard output information about all processes, except the session leaders and processes not associated with a terminal.
-A	Writes to standard output information about all processes.
-c	Prints information in a format that reflects scheduler properties. The **-c** flag affects the output of the **-f** and **-l** flags, as described below.
-d	Writes to standard output information about all processes, except the session leaders.
-e	Writes to standard output information about all processes, except kernel processes.

Item	Description
-f	Generates a full listing.
-g *pgrplist*	Writes to standard output information only about processes that are in the process groups specified by *pgrplist*. Values for *pgrplist* must be numeric.
-G *grplist*	Writes to standard output information only about processes that are in the process groups specified by *grplist*. The **-G** flag accepts group names.
-j	Displays session ID and process group ID.
-l	Generates a long listing.
-L	Prints status of active threads within a process.
-o *format*	Displays information in the format specified by *format*. Multiple field specifiers can be specified for the format variable. The field specifiers that can be used with the **-o** flag are described above in the Format section.
-p *proclist*	Displays information only about processes with the process numbers specified by *proclist*. Values for *proclist* must be numeric.
-P	Displays the logical processor number of the processor to which the primary kernel thread of the process is bound (if any).
-s *sidlist*	Displays all processes whose session leader's IDs are specified by *sidlist*.
-t *termlist*	Displays information only about processes associated with the terminals specified by *termlist*.
-u *uidlist*	Displays information only about processes with the user ID numbers or login names specified by *uidlist*.
-U *uidlist*	Displays information only about processes with the user ID numbers or login names specified by *uidlist*.
-X	Prints all available characters of each user and group name instead of truncating to the first 8 characters.
-y	When combined with the **-l** option, changes the long listing so that it prints the "RSS'' and "SZ'' fields in kilobytes and does not print the "F'' and "ADDR'' fields.

Exit Status (System V)

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security (System V)

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples (System V)

1. To display all processes, enter:

```
ps -e -f
```

2. To list processes owned by the user 'guest', enter:

```
ps -f -l -u guest
```

3. To list processes that are associated with the **/dev/pts/0** and **/dev/pts/1** terminals, enter:

```
ps -t pts/0,pts/1
```

4. To list processes not associated with a terminal, enter:

```
ps -t -
```

5. To display a specified format with field specifiers, enter:

```
ps -o ruser,pid,ppid,args
```

6. To display information about all kernel threads in a process, enter:

```
ps -L
```

7. To display session ID and process group IDs of all the processes, enter:

```
ps -jA
```

8. To display the scheduling class and priority of processes, enter:

```
ps -c -l
```

9. To display information about kernel threads and the number of kernel threads in a process, enter:

```
ps -L -f
```

10. To display the processor to which the process or kernel thread is bound to, enter:

```
ps  -P
```

11. To print an ASCII art for a given process (`inetd` in the example below), enter :

```
ps -T 14220
```

Output will look like the following:

```
    PID    TTY  TIME CMD
  14220      -  0:00 inetd
  16948      -  0:00    |\--telnetd
  32542  pts/4  0:00    |    \--ksh
  26504      -  0:00    |\--telnetd
  41272  pts/5  0:00    |    \--ksh
  26908  pts/5  0:00    |        \--vi
  28602      -  0:00    |\--telnetd
  24830  pts/0  0:00    |    \--ksh
 676416  pts/0  0:00    |        \--ksh
  29984      -  0:00    |\--telnetd
  38546  pts/6  0:00    |    \--ksh
  32126      -  0:00    |\--telnetd
  11162  pts/7  0:00    |    \--ksh
  34466      -  0:00    |\--rpc.ttdbserver
  35750      -  0:00    |\--telnetd
  23612  pts/3  0:00    |    \--ksh
  36294      -  0:00    |\--telnetd
  38096  pts/8  0:00    |    \--ksh
  39740      -  0:00    |\--telnetd
  42226  pts/9  0:01    |    \--ksh
  40632      -  0:00    |\--telnetd
  40232  pts/2  0:00    |    \--ksh
  32910  pts/2  0:00    |        \--dbx
 987990  pts/2  0:00    |            \--a.out
  40722      -  0:00    |\--telnetd
  16792 pts/10  0:00    |    \--ksh
  37886 pts/10  0:00    |        \--ps
 105716      -  0:00     \--telnetd
  29508  pts/1  0:00         \--ksh
  39478  pts/1  0:00             \--ksh
  38392  pts/1  0:00                 \--vi
```

12. To print information about all processes rooted at a given pid, enter:

```
ps -fL 14220
```

Output will look like the following:

```
  UID   PID  PPID   C    STIME    TTY  TIME CMD
root 14220  8676   0   Apr 07      -  0:00 /usr/sbin/inetd
root 16948 14220   0   Apr 06      -  0:00 telnetd -a
root 23612 35750   0   Apr 10  pts/3  0:00 -ksh
root 24830 28602   1 18:30:56  pts/0  0:00 -ksh
root 28602 14220   0 18:30:55      -  0:00 telnetd -a
```

```
root 32542 16948   0   Apr 06  pts/4  0:00 -ksh
root 34466 14220   0   Apr 10      -  0:00 rpc.ttdbserver 100083 1
root 35750 14220   0   Apr 10      -  0:00 telnetd -a
root 40228 24830   8 18:36:01  pts/0  0:00 ps -fL 14220
```

13. To display all processes with expanded user name, type:

```
ps -X -e -f
```

14. To display the scheduling policies of a thread, enter the following command:

```
#ps -m -o THREAD,sched
 USER     PID    PPID      TID ST  CP PRI SC WCHAN   F     TT BND COMMAND     SCH
suresana 1609830 4227284        - A   16  68  1   - 200001 pts/144   - ps -m
                                                              -o THREAD sched 0
       -       -       -  6381739 R   16  68  1   -400000      -   - -         0
suresana 4227284 4239476        - A    1  60  1   -200801 pts/144   - bash    0
       -       -       -  4177981 S    1  60  1   -410400      -   - -         0
suresana 4239476  921694        - A    0  60  1   -240001 pts/144   - -ksh    0
       -       -       -  5554385 S    0  60  1   -10400       -   - -         0
```

Files (System V)

Item	Description
/usr/sysv/bin/ps	Contains the System V R4 **ps** command.
/etc/passwd	Contains the user ID information.
/dev/pty*	Indicates terminal (PTY) names.
/dev/tty*	Indicates terminal (TTY) names.

ps4014 Command

Purpose

Converts a Tektronix 4014 file to PostScript format.

Syntax

ps4014 [**-m**] [**-C**] [**-N**] [**-R**] [**-s***Width*,*Height*] [**-l***Left*,*Bottom*] [**-S***Width*] [**-p***OutFile*] [*File*]

Description

The **ps4014** command reads in a Tektronix 4014 format file and converts it to PostScript format for printing on a PostScript printer. If no file is specified, the standard input is used. The resulting PostScript file can be directed to standard output or to a named file.

> **Note:** By default, the 4014 image is scaled to occupy nearly the entire page in a landscape orientation.

Flags

> **Note:** The **-m**, **-C**, and **-N** flags specify values for 4014 hardware options that affect the interpretation of 4014 commands.

Item	Description
-C	Causes a carriage return to move the pen position to the left margin but not down to the next line. By default, a carriage return command moves the pen down to the next line and over to the left margin.

Item	Description
-l*Left*,*Bottom*	Specifies the location on the printed page of the bottom left corner of the converted raster image. The values specified by the *Left* and *Bottom* parameters are the distances (in inches) from the bottom left corner of the printed page to the bottom left corner of the image.
-m	Enables the "Margin 2" mode for the 4014.
-N	Causes line feed to move the pen position down to the next line but not to the left margin. By default, a line feed command moves the pen down to the next line and over to the left margin.
-p*OutFile*	Causes the PostScript file to be written to the file specified by the *OutFile* parameter rather than the standard output.
-R	Rotates the image 90 degrees on the page for portrait orientation. The default is landscape orientation.
-s*Width*,*Height*	Specifies the size of the converted raster image on the printed page. The *Width* and *Height* parameters specify the dimensions (in inches) of the resulting image on the printed page.
-S*Width*	Allows you to scale the image without distorting its shape. The *Width* parameter specifies the width, in inches, of the resulting image on the printed page. The height of the image is computed to maintain the same ratio of height to width on the output image as on the input raster-format file.

International Character Support

See the html

ps630 Command

Purpose

Converts Diablo 630 print files to PostScript format.

Syntax

ps630 [**-f***Bodyfont*] [**-p***File*] [**-s***Pitch*] [**-F***Boldfont*] [*File* ...]

Description

The **ps630** command converts Diablo 630 format print files to PostScript format for printing on a PostScript printer. If no *File* variable is specified, the **ps630** command reads from standard input. By default, the PostScript file is sent to the standard output.

The **ps630** command can convert **nroff** files generated with the **-Txerox** flag. Typewheel emulation information can be specified as options. Font specifications (for bold and regular) are PostScript font names (such as Times-Roman, Times-Bold, Courier-Bold, Courier-BoldOblique). You can select 10, 12, or 15 characters per inch.

Some applications produce bold type by double-striking a character. This type of bolding is not translated into PostScript format. Only the bold effect produced by issuing the proper Diablo command sequence (Esc-O) results in bold characters.

The output of the **ps630** command cannot be page-reversed. Times-Roman and Helvetica are narrow fonts that may look squeezed if no adjustment to the page width is made by the application.

The following Diablo 630 commands are not supported:

- Print suppression

- HY-Plot
- Extended character set
- Downloading print wheel information or program mode
- Page lengths other than 11 inches
- Paper feeder control
- Hammer energy control
- Remote diagnostic
- Backward printing control.

  Note: The Diablo 630 command for reverse printing is supported.

Flags

Item	Description
-f*Bodyfont*	Sets the font to be used for normal printing. The default is Courier.
-p*File*	Causes the PostScript file to be written to the file specified by the *File* parameter rather than to the standard output.
-s*Pitch*	Selects type size for printing (both the regular and bold fonts are scaled to this size). Pitch is in characters per inch and must be one of 10, 12, or 15. The default is 12.
-F*Boldfont*	Sets the font to be used for bold type. The default is Courier-Bold.

International Character Support

See the html

psc or psdit Command

Purpose

Converts **troff** intermediate format to PostScript format.

Syntax

{ **psc** | **psdit** } [**-f1** *CodeSet***:***Font*] [**-F***FontDirectory*] [**-M***MediaName*] [**-p***Prologue*] [**-o***List*] [*File*]

Description

The **psc** and **psdit** commands translate a file created by device-independent **troff** to PostScript format for printing with a PostScript printer. If no file is specified, the standard input is used. The PostScript file is sent to the standard output.

Note: The input for the **psc** and **psdit** commands should be prepared with the corresponding **-Tpsc** option, such as the **troff** or **pic** command.

The **psc** and **psdit** commands can handle extended characters created by modifying the printer code field in the font file (**/usr/lib/font/devpsc/R**). The modified field contains a string surrounded by double quotation marks. The string contains a **\b** (backslash b) followed by a sequence of characters from the standard font that is composed into a new character by overstriking.

The **psc** and **psdit** commands allow users to cause the **troff** command to include arbitrary PostScript code in the generated PostScript file. The **psc** and **psdit** commands recognize the undefined **%** (percent) command in the **troff** intermediate file format to signal the start of raw PostScript code to be placed as is in the output file. Everything between (but not including) the **%** (percent sign) and a line containing a **.** (period) will be placed in the generated PostScript output.

This PostScript output is not insulated from the **troff** command coordinate system or the state of the generated PostScript output. However, two functions are defined in the prologue so that users can insulate themselves if so desired. The **PB** (picture begin) function performs a PostScript save operation, translates the PostScript coordinate system to **troff**'s idea of the current position on the page, and changes the scale and orientation of the coordinate system axes to the standard PostScript 72 units per inch. The **PE** (picture end) macro ends this protected environment.

Several methods can be used to incorporate such included PostScript code into the **troff** intermediate file. For example, the **.sy**, **\!**, and **.cf** subcommands of the **troff** command use the following example to include the PostScript language description of a completely separate, printable document. In this example, the **showpage** operator is redefined to include `mypic.ps` as an illustration:

```
standard troff input
\&
.fl
\!%PB
\!/showpage{}def
.fl
.sy cat mypic.ps
\!PE
\!.
more standard troff input
```

Information containing various media sizes for the **psdit** command and the **enscript** command are contained in the file **/usr/lib/ps/MediaSizes**.

The information required for each entry in the **MediaSizes** file can be obtained from the **PostScript Printer Description**, or **PPD**, file that matches the PostScript printer used with TranScript. The **PPD** files are available from Adobe Systems Incorporated. The measurements extracted form the **PPD** files are in points. A printer's point is 1/72 of an inch.

Any line in the **MediaSizes** file beginning with an ASCII ***** (asterisk) is ignored when matching media size names provided on the command line to the **enscript** command and the **psdit** command.

Each entry in the **MediaSizes** file contains either eight or nine fields. The first eight fields are required for all entries. The ninth field is optional. Fields are separated by white space. The fields for each entry are as follows:

Field Name	Description
EntryName	Character string to match against a media name provided with the **-M** option with the **enscript** command or the **psdit** command.
MediaWidth	Media width in points.
MediaDepth	Media depth in points.
ImageableLLX	Imageable lower left-hand corner x coordinate in points.
ImageableLLY	Imageable lower left-hand corner y coordinate in points.
ImageableURX	Imageable upper right-hand corner x coordinate in points.
ImageableURY	Imageable upper right-hand corner y coordinate in points.
PageRegionName	PostScript sequence for the particular printer to identify the size of the imageable area.
PaperTrayName	PostScript sequence for the particular printer to select a particular paper/media tray. This field is optional.

Note: The sequence can be multiple PostScript operators or words for both the **PageRegionName** field and the **PaperTrayName** field. To specify such a sequence, use the ASCII " (double quotation mark character) to delimit the entire sequence.

The following are examples of field entries in the **MediaSizes** file:

Name	Entries
Letter	**Width** 612 **Depth** 792 **llx** 18 **lly** 17 **urx** 597 **ury** 776 **Page- Region- Name** Letter **Page- Tray- Name**
Legal	**Width** 612 **Depth** 1008 **llx** 18 **lly** 17 **urx** 597 **ury** 992 **Page- Region- Name** Legal **Page- Tray- Name**

Flags

-f1 *CodeSet*:*Font*

Item	Description
-F*FontDirectory*	Takes font information from *FontDirectory* instead of the default.
-M*MediaName*	Specifies a media name to use to determine the amount of imageable area on the paper. The name provided is matched against entries in the **MediaSizes** file. For instance, `-M legal` would request a legal size of paper as the imageable area. If this option is not used, the default size is letter size, which is 8.5 inches wide by 11.0 inches deep.
-p*Prologue*	Uses the contents of *Prologue* instead of the default PostScript prologue.

Item	Description
-o*List*	Prints pages whose numbers are given in the list separated by commas. The list contains single numbers and ranges in the format *N1*-*N2*, where *N1* and *N2* represent page numbers. A missing *N1* means the range begins with the lowest-numbered page; a missing *N2* means the range ends with the highest-numbered page.

Examples

The following statements are equivalent:

```
pic -Tpsc File | troff -Tpsc | psc
pic -Tpsc File | troff -Tpsc | psdit
```

Environment Variables

Item	Description
PSLIBDIR	Path name of a directory to use instead of the **/usr/lib/ps** file for the **psc** and **psdit** command prologue.
TRANSCRIPT	Absolute path name of a file to use instead of **/usr/lib/ps/transcript.conf** for the MBCS handling.

Files

Item	Description
/usr/lib/font/devpsc/*	Contains the **troff** default description files for a PostScript virtual device.
/usr/lib/ps/psdit.pro	Contains the default PostScript prologue.
/usr/lib/ps/MediaSizes	Contains the default file used for media sizes.
/usr/lib/ps/transcript.conf	Contains the default value used for PostScript codeset and font name.

pshare Command

Purpose

Enables or reports the availability of shared login ports.

Syntax

pshare [**-a**] [*Device*]

Description

The **pshare** command enables shared ports. Shared ports are bidirectional. If you do not specify a *Device* parameter, the **pshare** command reports the names of all currently enabled shared ports. To enable a shared port, the **getty** command attempts to create a **lock** file in the **/etc/locks** directory that contains the ASCII process ID of the process. If another process is already using the port, the **getty** command waits until the port is available and tries again. The system enables a port by updating an entry in the **/etc/inittab** file and then sending a signal to the **init** process. After receiving the signal and reading the updated status entry, the process takes the appropriate action.

Use the *Device* parameter to specify the ports to be enabled. Permitted values include:

- Full device name, such as the **/dev/tty1** device
- Simple device name, such as the **tty1** device
- A number (for example, 1 to indicate the **/dev/tty1** device)

Note: You must have root user authority to run this command.

Flags

Item	Description
-a	Enables all ports as shared.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To enable the workstation attached to the **/dev/tty2** port as a shared port, enter:

```
pshare /dev/tty2
```

Item	Description
/etc/inittab	Controls system initialization.

Files

Item	Description
/etc/locks	Contains **lock** files for the **pshare** and **pdelay** commands.
/usr/sbin/pshare	Contains the **pshare** command.

psplot Command

Purpose

Converts files in plot format to PostScript format.

Syntax

psplot [**-g** *Prologue*] [*File...*]

Description

The **psplot** command reads files in plot format and converts them to PostScript format on the standard output. If no files are specified, the standard input is used. The conversion is almost one-to-one, with one PostScript function call for each plot primitive. You can modify the behavior of the file by changing the definitions of the PostScript functions in the prologue.

Flags

Item	Description
-g*Prologue*	Uses the contents of the *Prologue* file instead of the default PostScript prologue. If this flag is not specified, the default prologue file is used.

International Character Support

The html

psrasc Command

Purpose

Collects centralized RAS data.

Syntax

psrasc type [**-d**] [**-n** *number*] **-o** *outputFile* **logSpace/logStream**

Description

The **psrasc** command extracts the Reliability/Availability/Serviceability (RAS) data log records centralized on a PowerHA pureScale log stream and builds a file in the RAS data AIX format. The PowerHA pureScale service name is **CentralizedLogService**. Binding information for that service name must be setup before using the **psrasc** command.

RAS data types

When the specified type is **syslog**, the log records contain system log messages, including message initiator hostname. The format of the generated file is similar to the system log destination files. When the specified type is **errlog**, the log records contain error log entries. The generated file is an error log file that can be later exploited by the **errpt** command.

Flags

Item	Description
type	Specifies the type of RAS data contained in the log records. This must be the first parameter. Supported RAS data types are: **syslog** and **errlog**. From this type, depends the format of the output file.
-d	Specifies that the collected log record are deleted.
-n *number*	Specifies the number of log records to collect. Oldest log records are collected. When this parameter is not specified, all the log records are collected.
-o *outputFile*	Specifies the relative or absolute pathname of the output file. If the file already exists, it is overwritten.
log_space/ log_stream	Specifies the fullname of the log stream from which system log messages are collected. Fullname is made of the parent log space name and the log stream name separated by a **/** (slash).

Exit Status

This command returns the following exit values:

Item	Description
`0`	Successful completion.
`> 0`	An error occurred.

Examples

1. To collect log records of the log stream named **CentralizedRAS/Syslog** into the **syslog.out** file on the

   PowerHA pureScale server identified by the **CentralizedLogService** service name, enter:

   ```
   psrasc syslog -o syslog.out CentralizedRAS/Syslog
   ```
2. To collect the 100 oldest log records of the log stream named **CentralizedRAS/Syslog** into the **/var/adm/ras/cluster_syslog** file on the PowerHA pureScale server

   identified by the **CentralizedLogService** service name and delete them, enter:

   ```
   psrasc syslog -d -n 100 -o /var/adm/ras/cluster_syslog CentralizedRAS/Syslog
   ```
3. To collect log records of the log stream named **CentralizedRAS/Errlog** into the file **centralizedRAS_errlog** on the PowerHA pureScale server identified by the CentralizedLogService service name, enter:

   ```
   psrasc errlog -o centralizedRAS_errlog CentralizedRAS/Errlog
   ```
4. To collect and delete the 100 oldest log records of the log stream named **CentralizedRAS/Errlog** into the **/var/adm/ras/cluster_errlog** error log file on

   the PowerHA pureScaleserver identified by the **centralizedRAS_error** service, enter:

   ```
   psrasc errlog -d -n 100 -o /var/adm/ras/cluster_errlog CentralizedRAS/Errlog
   ```

psrev Command

Purpose

Reverses the page order of a PostScript file and selects a page range for printing.

Syntax

psrev [**-R**] [**-s***Pagespec*,...] [*File*]

Description

The **psrev** command reverses the page order of the file specified by the *File* variable and prints the pages specified by the *Pagespec* parameter. The file must conform to PostScript file structuring conventions. If no value for the *File* is specified, the **psrev** command reads from standard input. The **psrev** command writes the resulting file to the standard output.

Flags

Item	Description
-R	Does not reverse the page order (but subsets the pages if specified).
-s*Pagespec*	Specifies a range (or several ranges) of pages to be printed. The *Pagespec* parameter is a string with no spaces. The *Pagespec* parameter can be a single page number or a range of the form *N-M*, which prints pages *N* through *M*. *-N* prints from the beginning of the document to page *N*. *M-* prints from page *M* to the end of the document.

Examples

The following are examples of using the **psrev** command showing page ranges and an individual page in nonreversed order:

```
psrev -R -s2-4,6
```

```
psrev -R -s2-4,6-8
```

Files

Item	Description
/var/tmp/RV*	Contains the temporary file if the input is a pipe.

psroff Command

Purpose

Converts files from **troff** format to PostScript format.

Syntax

psroff [**-t**] [**-d***Queue*] [**-n***Number*] [**-t***Title*] [**-D***FontDirectory*] [**-F***FontFamily*] [**-P***Flag*] [*troffFlags*] [*File ...*]

Description

The **psroff** command is a shell script that runs the **troff** command in an environment to produce output on a PostScript printer. It uses the **psdit** command to convert **troff** intermediate output to PostScript format, and spools this output for printing. If no files are specified, the standard input is used.

To include arbitrary PostScript language commands or files in a **troff** document, see the **psdit** command.

PostScript Font Information

The PostScript Fonts for Transcript table shows the fonts available for the TranScript commands. The fonts are available by long name when using the **enscript** command, and by short name when using the **psroff** or **troff** commands. The following table shows the **psroff** commands (short names) used to declare a default set of fonts. The alphabetic characters are case-sensitive:

PostScript Fonts for Transcript	
Long Name (Short Name)	**Font Family**
AvantGarde-Book (ag)	AvantGarde
AvantGarde-Demi (Ag)	AvantGarde
AvantGarde-DemiOblique (AG)	AvantGarde
AvantGarde-BookOblique (aG)	AvantGarde
Bookman-Demi (Bo)	Bookman
Bookman-DemiItalic (BO)	Bookman
Bookman-Light (bo)	Bookman
Bookman-LightItalic (bO)	Bookman
Courier (C)	Courier

PostScript Fonts for Transcript *(continued)*	
Long Name (Short Name)	**Font Family**
Courier-Bold (CB)	Courier
Courier-BoldOblique (CO)	Courier
Courier-Oblique (CO)	Courier
Garamond-Bold (Ga)	Garamond
Garamond-BoldItalic (GA)	Garamond
Garamond-Light (ga)	Garamond
Garamond-LightItalic (gA)	Garamond
Helvetica (H)	Helvetica
Helvetica-Bold (HB)	Helvetica
Helvetica-Oblique (HO)	Helvetica
Helvetica-BoldOblique (HD)	Helvetica
Helvetica-Narrow (hn)	Helvetica
Helvetica-Narrow-Bold (Hn)	Helvetica
Helvetica-Narrow-BoldOblique (HN)	Helvetica
Helvetica-Narrow-Oblique (hN)	Helvetica
LubalinGraph-Book (lu)	Lubalin
LubalinGraph-BookOblique (lU)	Lubalin
LubalinGraph-Demi (Lu)	Lubalin
LubalinGraph-DemiOblique (LU)	Lubalin

Item	**Description**
NewCenturySchlbk (NC)	NewCentury
NewCenturySchlbk-Bold (Nc)	NewCentury
NewCenturySchlbk-Italic (nC)	NewCentury
NewCenturySchlbk-Roman (nc)	NewCentury
Optima (op)	Optima
Optima-Bold (Op)	Optima
Optima-BoldOblique (OP)	Optima
Optima-Oblique (oP)	Optima
Palatino-Bold (PB)	Palatino
Palatino-BoldItalic (PX)	Palatino
Palatino-Italic (PI)	Palatino
Palatino-Roman (PA)	Palatino
Souvenir-Demi (Sv)	Souvenir
Souvenir-DemiItalic (SV)	Souvenir

Item	Description
Souvenir-Light (sv)	Souvenir
Souvenir-LightItalic (sV)	Souvenir
Times-Bold (TB)	Times
Times-BoldItalic (TD)	Times
Times-Italic (TI)	Times
Times-Roman (TR)	Times
Symbol (S)	(none)
ZapfChancery-MediumItalic (ZC)	Zapf
ZapfDingbats	(none)

Flags

Item	Description
-D*FontDirectory*	Finds font family directories in the specified font directory, rather than the standard font directory, which was configured in the installation procedure. It may be necessary to use both this flag and the **-F** flag to imitate the **-F** flag in the **troff** command.
-d*Queue*	Causes the output to be queued to the queue specified by the *Queue* parameter. If the **-d** flag is not used, the **psroff** command queues output on the default queue, the first queue known to the **qdaemon**. This flag is recognized by the spooler print.
-F*FontFamily*	Uses the specified font family for the R/I/B/BI fonts, rather than the Times default family. The Times, Courier, and Helvetica font families are defined at your site, and others are available as well. Ensure that the printer you use contains the font family you pick. This flag overrides the **troff** command **-F** flag. If you want to use the **troff** command **-F** flag, you should run the **troff** command directly or use the **-D** flag instead.
-n*Number*	Causes the number of output copies specified by the *Number* parameter to be produced. The default is one. This flag is recognized by the spooler print.
-P*Flag*	Passes the *Flag* parameter to the spooler. This flag is useful when a conflict exists between a spooler flag and a flag with the **psroff** command.
-t	Sends the PostScript output to the standard output, rather than spooling it to a printer. This flag overrides the **troff** command **-t** flag. If you want the **troff** command **-t** flag, you should run the **troff** command directly.
-t*Title*	Sets the job name for use on the first banner page. The default is to use the name of the first input file. This flag is recognized by the spooler print.

Parameters

Item	Description
troffFlags	Specifies standard flags available with the **troff** command.
File	Specifies the **troff** intermediate output file. The default is the standard input.

Files

Item	Description
/usr/share/lib/tmac/tmac.*	Contains the standard macro files.
/usr/lib/font/devpsc/*	Contains the **troff** description files for PostScript virtual device.
/usr/lib/ps/*.afm	Contains Adobe Font Metrics (AFM) files for use with the **enscript** command.
/usr/lib/ps/font.map	Contains the list of font names with their abbreviations.
/usr/lib/ps/ditroff.font	Contains font family files for the **troff** command.

pstart Command

Purpose

Enables or reports the availability of login ports (normal, shared, and delayed).

Syntax

pstart [**-a**] [*Device*]

Description

The **pstart** command enables all ports (normal, shared, and delayed) listed in the **/etc/inittab** file. The system enables a port by updating an entry in the **/etc/inittab** file and then sending a signal to the **init** process. When the **init** process receives the signal and reads the updated status entry, it takes the appropriate action.

Use the *Device* parameter to specify the ports to be enabled. Permitted values include:

- A full device name, such as the **/dev/tty1** device
- A simple device name, such as the **tty1** device
- A number (for example, 1 to indicate the **/dev/tty1** device)

If you do not specify a *Device* parameter, the **pstart** command reports the names of all enabled ports and whether they are currently enabled as normal, shared, or delayed.

Note: You must have root user authority to run this command.

Flag

Item	Description
-a	Enables all ports (normal, shared, and delayed ports).

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the names of all ports (normal, shared, and delayed) currently enabled and how they are enabled, enter:

```
pstart
```

2. To enable all normal, shared, and delayed ports listed in the **/etc/inittab** file, enter:

```
pstart -a
```

Files

Item	Description
/etc/locks	Contains **lock** files for the **pshare** and **pdelay** commands.
/usr/sbin/pstart	Contains the **pstart** command file.

pstat Command

Purpose

Interprets the contents of the various system tables and writes it to standard output.

Syntax

pstat [**-a**] [**-A**] [**-f**] [**-i**] [**-p**] [**-P**] [**-s**] [**-S**] [**-t**] [**-u***ProcSlot*] [**-T**] [**-U** *ThreadSlot*] [[*KernelFile*] *CoreFile*]

Description

The **pstat** interprets the contents of the various system tables and writes it to standard output. You must have root user or **system** group authority to run the **pstat** command.

Flags

Item	Description
-a	Displays entries in the process table.
-A	Displays all entries in the kernel thread table.
-f	Displays the file table.
-i	Displays the i-node table and the i-node data block addresses.
-p	Displays the process table.
-P	Displays runnable kernel thread table entries only.
-s	Displays information about the swap or paging space usage.
-S	Displays the status of the processors.
-t	Displays the tty structures.
-u *ProcSlot*	Displays the user structure of the process in the designated slot of the process table. An error message is generated if you attempt to display a swapped out process.
-T	Displays the system variables. These variables are briefly described in var.h.
-U *ThreadSlot*	Displays the user structure of the kernel thread in the designated slot of the kernel thread table. An error message is generated if you attempt to display a swapped out kernel thread.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the i-nodes of the system dump saved in the **dumpfile** core file, enter:

   ```
   pstat -i dumpfile
   ```

   Symbols are retrieved from the **/usr/lib/boot/unix** file.
2. To display the file table and the user structure for the process in process table slot 0 (zero) of the system currently running, enter:

   ```
   pstat -f -u 0
   ```
3. To display the tty structures for a system dump, whose core file is **dumpfile** and whose kernel is the **/usr/lib/boot/unix.back** file, enter:

   ```
   pstat -t /usr/lib/boot/unix.back dumpfile
   ```
4. To display all threads in the kernel thread table and the user structure of the thread in thread table slot 2, enter:

   ```
   pstat -A -U 2
   ```

Files

Item	Description
/**usr**/**sbin**/**pstat**	Contains the **pstat** command.
/dev/mem	Default system-image file.
/usr/lib/boot/unix	Default kernel-image file.
/usr/include/sys/*.h	Contains header files for table and structure information.

ptpd Daemon

Purpose

Starts the Precision Time Protocol (1588-2008) daemon (**ptpd**).

Syntax

/usr/sbin/ptpd [**-?**] [**-h**] [**-H**] [**-e** *setting*] [**-k**] [**-v**] [**-O**] [**-L**] [**-A**] [**-s**] [**-m**] [**-M**] [**-y**] [**-E**] [**-P**] [**-a**] [**-n**] [**-C**] [**-V**] [**-c** *file*] [**-R** *dir*] [**-f** *file*] [**-S** *file*] [**-d** *domain_number*] [**-u** *IP_address*] [**-r** *number*] [**-l** *file*] [**-i** *dev*]

Description

The **ptpd** daemon implements the Precision Time Protocol (PTP) version 2 as defined by the IEEE 1588-2008 standard. PTP provides precise time coordination of LAN-connected computers. You must run this daemon with root authority to manipulate the system clock and use lower port numbers. The **ptpd** daemon supports IPv4 multicast, unicast, hybrid mode (mixed), and Ethernet mode operations. The

ptpd daemon can achieve and maintain submicrosecond level timing precision, even without hardware assistance.

Configure the **ptpd** daemon by using the `/etc/ptpd2.conf` configuration file (default file). The short (**-x**) and long (**--xxxxx**) flags provide basic control over the daemon operation, and provide only the basic PTP protocol settings. Other settings can be displayed by the **-h**, **-H**, and **-e** flags.

The **ptpd** daemon can be started either from System Resource Controller (SRC) or from the command line.

Use the following SRC commands to manipulate the **ptpd** daemon:

startsrc
Starts a subsystem, group of subsystems, or a subserver.

stopsrc
Stops a subsystem, group of subsystems, or a subserver.

refresh
Causes a subsystem or group of subsystems to read the appropriate configuration file again.

lssrc
Gets the status of a subsystem, group of subsystems, or a subserver.

Note: The **ptpd** daemon does not support Network Time Protocol (NTP) and Simple Network Management Protocol (SNMP) related functions.

The **ptpd** daemon uses the following terms:

slave device
A system running the **ptpd** daemon that accepts commands from a *master* device and synchronizes its system time to match the associated boundary clock time.

master device
A device bound to a boundary clock that synchronizes time with a set of one or more PTP *slave* devices on the same network.

grandmaster device
A *master* device that has the best clock provided by the Best Master Clock algorithm. It synchronizes all of the other *master* devices (also known as the boundary clocks), which in turn update and synchronize all of the associated *slave* devices. The grandmaster clock is also known as best master clock or best clock.

Flags

Short flag	Long flag	Description
-a	**--delay-override**	Overrides delay request interval announced by master in worker state. This flag sets the `ptpengine` attribute to the following setting: `ptpengine:log_delayreq_override`
-A	**--auto-lock**	Uses preset or port mode-specific lock-file names. This flag is useful when you run multiple instances of the **ptpd** daemon.
-c *file*	**--config-file** *file*	Specifies the path of the configuration file.
-C	**--foreground**	Specifies the command to run in foreground. This flag sets the `global` attribute as follows: `global:foreground=Y` **Note:** This option is ignored if the **ptpd** daemon is started from SRC.

Short flag	Long flag	Description
-d *domain_number*	**--domain** *domain_number*	Specifies the PTP domain number to become part of. This flag sets the `ptpengine` attribute as follows: `ptpengine:domain`
-D [DD]	**--debug**	Specifies the debug level. This flag sets the `global` attribute as follows: `global:debug_level` You can specify multiple instances to increase the debug level information. For example, the **-D** option specifies minimal debug information. The **-DD** option specifies detailed debug information. The **-DDD** option specifies maximum debug information.
-e *setting*	**--explain** *setting*	Shows help information for a single setting. This flag sets the `section` attribute as follows: `section:key`
-E	**--e2e**	Specifies end-to-end delay detection. This flag sets the `ptpengine` attribute as follows: `ptpengine:delay_mechanism=E2E`
-f *file*	**--log-file** *file*	Specifies the path of the log file. This flag sets the `global` attribute as follows: `global:logfile`
-h	**--help**	Shows help screen.
-H	**--long-help**	Shows detailed help for all settings and behaviors.
-i *dev*	**--interface** *dev*	Specifies the interface to be used for **ptpd** implementation. For example, `en0`. This flag sets the `ptpengine` attribute as follows: `ptpengine:interface`
-k	**--check-config**	Checks the PTP configuration and exits. Returns 0 if the configuration is correct.
-l *file*	**--lockfile** *file*	Specifies the path of lock-file. This flag sets the `global` attribute as follows: `global:lock_file`
-L	**--ignore-lock**	Skips checking and locking the lock-file. This flag sets the `global` attribute as follows: `global:ignore_lock`
-m	**--masterslave**	Specifies the full IEEE 1588 implementation: master, slave when not gandmaster (best master). This flag sets the `ptpengine` attribute as follows: `ptpengine:preset=masterslave`

Short flag	Long flag	Description
-M	**--masteronly**	Specifies master only mode: passive when not best GM. This flag sets the `ptpengine` attribute as follows: `ptpengine:preset=masteronly`
-n	**--clock:no_adjust**	Specifies not to adjust the clock. This flag sets the `clock` attribute as follows: `clock:no_adjust`
-O	**--default-config**	Shows default configuration and exits. The output can be used as a configuration file.
-p	**--print-lockfile**	Prints path of the lock-file and exits. This flag is useful for `init` scripts in combination with auto lock-files.
-P	**--p2p**	Specifies peer-to-peer delay detection. This flag sets the `ptpengine` attribute as follows `ptpengine:delay_mechanism=P2P`
-r *number*	**--delay-interval** *number*	Specifies the interval of delay request message (log 2). This flag sets the `ptpengine` attribute as follows: `ptpengine:log_delayreq_interval`
-R *dir*	**--lock-directory** *dir*	Specifies the directory to store lock-files. This flag sets the `global` attribute as follows: `global:lock_directory`
-S *file*	**--statistics-file** *file*	Specifies the path of statistics file. This flag sets the `global` attribute as follows: `global:statistics_file`
-s	**--slaveonly**	Turns on the slave-only mode. This flag sets the `ptpengine` attribute as follows: `ptpengine:preset=slaveonly`
-u	**--unicast**	Specifies unicast mode (no unicast negotiation) and sends all messages to IP. This flag sets the `ptpengine` attribute as follows: `ptpengine:ip_mode=unicast + ptpengine:unicast_address`
-v	**--version**	Prints the version string and exits.
-V	**--verbose**	Specifies the command to run in foreground and to log all the messages to the standard output. This flag sets the `global` attribute as follows: `global:verbose_foreground=Y` **Note:** This option is ignored if the **ptpd** daemon is started from SRC.

Short flag	Long flag	Description
-y	**--hybrid**	Specifies hybrid mode: mixed multicast and unicast operation. Multicast for sync and announce, unicast for delay request and response. This flag sets the `ptpengine` attribute as follows: `ptpengine:ip_mode=hybrid`

PTP daemon port states

The **ptpd** port can have the following states:

State	Description
`init`	Initializing
`flt`	Faulty
`lstn_init`	Listening (first time)
`lstn_reset`	Listening (subsequent reset)
`pass`	Passive (not best master, not announcing)
`uncl`	Uncalibrated
`slv`	Worker
`pmst`	Pre-master
`mst`	Master (active)
`dsbl`	Disabled
`? (unk)`	Unknown state

Statistics log file format

The following options are available when you enable the **ptpd** statistics log:

ptpengine:log_statistics
Updates the login information for each received PTP packet.

ptpengine:statistics_file
Specifies the location path of the statistics log file.

Note: This option enables statistics gathering.

When the statistics logging is enabled, a **ptpd** worker logs clock sync information when sync and delay response message are received. When the **ptpd** daemon starts up or flushes the log, a comment line (starting with #) is logged, containing the names of all columns. The log file is in the comma-separated values (CSV) format and can be easily imported into statistics tools and spreadsheet software packages for analysis and creating graphs. The size of the log files increases when you run the **ptpd** daemon for longer duration and with high message rates. Therefore, to reduce the number of messages logged, the `global:statistics_log_interval` setting can be used to limit the log output to one message per configured interval. The size and maximum number of the statistics log files can also be controlled.

The description of the columns in the statistics log file follows:

Timestamp
Time when the message was received. The date and time information are represented as text, UNIX time stamp (with fractional seconds), or both forms (in this case, an extra field is added), depending on the `global:statistics_timestamp_format` setting. When you import the log file to plotting software, if the software can understand UNIX time, set the time stamp format to `unix` or `both`, because some software do not interpret the fractional part of the second when it converts the date and time from text.

State
The state of the port. For more information about various port states, see "PTP daemon port states" on page 3261.

Clock ID
Port identity of the current best master, as defined by IEEE 1588 standard. This ID is the local clock's ID if the local clock is the best master. This parameter is displayed as `clock_id` or `port` (host). Port is the PTP clock port number, not the User Datagram Protocol (UDP) port numbers. The clock ID is an Extended Unique Identifier (EUI)-64 64-bit ID, converted from the 48-bit MAC address, by inserting `0xfffe` at the middle of the MAC address.

One-way delay
Current value of one-way delay (or mean-path delay) in seconds, calculated by the **ptpd** daemon that is in the worker state from the delay request and delay response message exchange.

Note: If this value remains at zero, it means that no delay response messages are being received, which might be because of a network issue.

Offset from master
Current offset value from master device in seconds. It is the main output of the PTP engine that is in the worker state. This value is the input for clock corrections in the clock servo algorithms. This value is typically measured when estimating the performance of the worker device.

Slave to master
Intermediate offset value (seconds) extracted from the delay request and delay response message exchange. This value is used for computing one-way delay. If the last value was rejected by a filter, the previous value is shown in the log file. This value is zero (0) if the delay response messages are not received.

Master to slave
Intermediate offset value (seconds) extracted from the sync messages. This value is used for computing the offset value from the master devices. If the last value was rejected by a filter, the previous value is shown in the log file.

Observed drift
The frequency difference between the worker clock and the master clock as measured by the integral accumulator of the clock control proportional integral (PI) servo model. This value stabilizes when the clock offset value is stabilized, and this value is used to detect clock stability.

Last packet received
This field shows which message was received last. It displays S for sync messages and D for delay response messages. If a worker device logs no D entries, it means that the worker device is not receiving delay response messages because of network issue.

One-way delay mean
One-way delay mean computed over the last sampling window.

One-way delay std dev
One-way delay standard deviation computed over the last sampling window.

Offset from master mean
Offset from master mean computed over the last sampling window.

Offset from master std dev
Offset from master standard deviation computed over the last sampling window.

Observed drift mean
Observed drift or local clock frequency adjustment mean computed over the last sampling window.

Observed drift std dev
Observed drift or local clock frequency adjustment standard deviation computed over the last sampling window. A lower value indicates that the clock is controlled less aggressively. Therefore, the value is more stable.

Note: All the statistical measures (mean and standard deviation) are computed and displayed only if the **ptpd** daemon was created by using the **--enable-statistics** flag. The duration of the sampling period is controlled with the `global:statistics_update_interval` setting.

Handling signals

The **ptpd** daemon handles the following signals:

Item	Description
SIGHUP	Reloads the configuration file (if used by the daemon) and reopens log files. The **refresh** subcommand of the SRC performs the same task.
SIGUSR1	When the subsystem is in the worker state, the **ptpd** daemon forces the clock to step to a current offset value from a master value.
SIGUSR2	Dumps all PTP protocol counters to current log target (and clears the counters if the `ptpengine:sigusr2_clears_counters` attribute is set).
SIGINT| SIGTERM	Closes log files and other open files. It also cleans up lock file and exits.
SIGKILL	Forces an unclean exit.

Exit status

Upon exit, the **ptpd** daemon returns 0 on success, either successfully started in daemon mode, or exited cleanly. The value of 0 is also returned when the **-k** (**--check-config**) option is used and the configuration was correct. A nonzero exit code is returned on errors. The value of 127 is returned if the **ptpd** daemon is started by a non-root user. The value of 3 is returned on lock-file errors and when the **ptpd** daemon cannot be started as daemon. The value of 2 is returned on memory allocation errors when the daemon is started. For all other error conditions such as configuration errors, running the **ptpd** daemon in help mode or without any parameters, self shutdown of subsystems, and network startup errors, the value of 1 is returned.

Examples

1. To start the **ptpd** daemon with the SRC, enter the following command:

   ```
   startsrc -s ptpd
   ```

2. To stop the **ptpd** daemon with the SRC, enter the following command:

   ```
   stopsrc -s ptpd
   ```

3. To refresh the **ptpd** daemon with the SRC, enter the following command:

   ```
   refresh -s ptpd
   ```

   The **ptpd** daemon reloads the configuration file (if used by the daemon) and reopens log files.

4. To check whether the configuration file in the `/etc/ptpd2.conf` path is configured correctly, enter the following command:

   ```
   ptpd -k
   ```

5. To view the meaning of a single setting, enter the following command:

   ```
   ptpd -e ptpengine:interface
   ```

   The output explains the meaning of the `ptpengine:interface` setting.

Files

Item	Description
`/etc/ptpd2.conf`	Default path of the **ptpd** daemon configuration file.

Item	Description
`/usr/samples/tcpip/ptpd2/ptpd2.conf`	Sample file of the `ptpd2.conf` configuration file.

ptsc Command

Purpose

Collects information from a trusted platform module (TPM) in preparation for an attestation request from an openpts verifier.

Syntax

ptsc [**options**] [**commands**]

Description

The **ptsc** command is the openpts collector. The command is used to gather measurements and events from the TPM (through the **tscd** interface), construct reference manifests (RMs) and convey them when requested to the openpts verifier. When a system is first configured for trusted boot, the collector must be initialized by using the **-i** option. This option generates a UUID and an associated RM stored in the /var/ptsc/<UUID>/rm0.xml file. If the system is changed and a new RM is required, the **-u** option is used and the verifier must be reinitialized.

Flags

Item	Description
Commands	
-i	Initializes the openpts collector.
-s	Specifies the startup (both self-test and the timestamp).
-t	Indicates the self-test.
-u	Updates the RM.
-U	Updates the RM automatically.
-D	Displays the configuration settings of the target or ALL the options. This is the default setting.
-m	If **-M** mode
Options	
-c *configfile*	Changes the location of the configuration file. The default is /etc/ptsc.conf.
-P *name=value*	Sets the properties.
-R	Removes the RM.
-Z	Uses an SRK secret of all zeros.
Miscellaneous	
-h	Displays the command usage information.
-V	Displays the information in verbose mode. Multiple **-V** options increase the verbosity and is used for debugging.

Files

Item	Description
/etc/ptsc.conf	The configuration file. This is the default location of the configuration file.
/var/ptsc/rm-uuid	The UUID of the current RM.
/var/ptsc/uuid	The UUID of the collector.
/var/ptsc/<UUID>/rm0.xml	The reference manifest.

ptsevt Command

Purpose

Manages the notifications of updates to the AIX system boot image.

Syntax

ptsevt [**-a**] [**-r**] [***host port***]

ptsevt -c

ptsevt [**-u uuid**] **-e**

Description

The **ptsevt** utility delivers events, by using the **-e** option about the boot image updates to which the attestation software known as listeners can subscribe. The optional **-u** argument can be used to specify the universally unique identifier (UUID) of the collector of the AIX system that is being updated. If the **-u** argument is not specified, the **ptsevt** command uses the default value found in the `/var/ptsc/uuid` file.

Subscribers can be added or removed by using the **-a** and **-r** options, respectively. The host can be a symbolic address or an IP or IPv6 number, and the TCP port must be a decimal number.

The **-c** option is used to clear the subscription list.

Flags

Item	Description
-a	Adds the listener specified by the host and port arguments to the destinations mentioned in the subscriber list.
-c	Clears the list of subscribers.
-e	Sends an event notification to all subscribers in the list.
-r	Removes the listener specified by the host and port arguments from the list of subscribers.
-u	Specifies the UUID that is sent as part of the notification. By default, the **ptsevt** command uses the value found in the **/var/ptsc/uuid** file.

Files

Item	Description
/var/ptsc/subscribers	The subscribers list.
/var/ptsc/subscribers.lo ck	The subscribers list lock file.
/var/ptsc/uuid	The default UUID sent as part of the notification.

ptsevtd Command

Purpose

Manages the notifications of updates to the AIX system boot image.

Syntax

ptsevtd [**-c** command] [**-d**] [**-f**] [**-p** ***port name***]

Description

The **ptsevtd** daemon listens to the events delivered by the **ptsevt** command when an attested system is being updated. By default, whenever an event is received, the **ptsevtd** command calls the **openpts** command with the universally unique identifier (UUID) of the system that is sending the event as the first argument. This process updates the corresponding reference manifest with the latest or the expected measurements. The -c option can be used to specify an alternative command that is called when a notification is received.

Use the -f option to run the daemon in the foreground. The -d option is specified multiple times to make the output more verbose. The -p argument specifies the port to be used to listen for event notifications.

Flags

Item	Description
-c	Specifies the command to call when a notification is received. If the option is not specified, the **openpts** command is used by default.
-d	Specifies the level to increase the verbosity of the output.
-f	Runs the listener in the foreground. The output is sent to the **stderr** console.
-p	Specifies the TCP port to use for event notifications. The default is 34185.

ptx Command

Purpose

Generates a permuted index.

Syntax

ptx [**-f**] [**-r**] [**-t**] [**-b** *Breakfile*] [**-g** *Number*] [**-w** *Number*] [**-i** *Ignore* | **-o** *Only*] [**—**] [*Infile* [*Outfile*]]

Description

The **ptx** command reads the specified English-language text (the *Infile* parameter), creates a rearranged index from it, and writes to the specified file (*Outfile*). Standard input and standard output are the defaults.

The **ptx** command searches the specified file (*Infile*) for keywords, sorts the lines, and generates the file *Outfile*. The *Outfile* file can then be processed with the **nroff** or **troff** command to produce a rearranged index.

The **ptx** command follows three steps:

1. Performs the permutation, generates one line for each keyword in an input line, and rotates the keyword to the front of the line.
2. Sorts the permuted file.

3. Rotates the sorted lines so that the keyword comes at the middle of each line.

The resulting lines in the *Outfile* file are in the following form:

```
.xx "" "before keyword" "keyword" "after keyword"
```

where `.xx` is an **nroff** or **troff** macro provided by the user or by the **ptx** command. The **mptx** macro package provides the `.xx` macro definition.

The `before keyword`, and `keyword`, and `after keyword` fields incorporate as much of the line as can fit around the keyword when it is printed. The first field and last field, at least one of which is always the empty string, are wrapped to fit in the unused space at the opposite end of the line.

Notes:

1. Line-length counts do not account for overstriking or proportional spacing.
2. Lines that contain a ~ (tilde) do not work, because the **ptx** command uses that character internally.
3. The **ptx** command does not discard non-alphanumeric characters.

Flags

Item	Description
-b *BreakFile*	Uses the characters in the specified break file to separate words. Tab characters, new-line characters, and spaces are always used as break characters.
-f	Folds uppercase and lowercase characters for sorting.
-g *Number*	Uses the specified number as the number of characters that the **ptx** command reserves for each gap among the four parts of the line as it is printed. The default *Number* variable value is 3.
-i *Ignore*	Does not use any words specified in the *Ignore* file as keywords. If the **-i** and **-o** flags are not used, the **/usr/lib/eign** file is the default *Ignore* file.
-o *Only*	Uses only the words specified in the *Only* file as keywords.
-r	Considers any leading non-blank characters of each input line as reference identifiers separate from the text of the line. Attaches the identifier as a fifth field on each output line.
-t	Prepares the output for the phototypesetter.
-w *Number*	Uses the specified number as the length of the output line. The default line length is 72 characters for the **nroff** command and 100 for the **troff** command.
—	(double dash) Indicates the end of flags.

Parameters

Item	Description
Infile	Specifies the English-language text. Standard input is the default file. The **ptx** command searches the specified file for keywords, sorts the lines, and generates the file *Outfile*.
Outfile	Specifies the file to which the **ptx** command writes the index created from the *Infile* file. Standard output is the default file. The *Outfile* file can be processed with the **nroff** or **troff** command to produce a rearranged index.

Files

Item	Description
/usr/lib/eign	Contains the default *Ignore* file.
/usr/share/lib/tmac/tmac.ptx	Contains the macro file.

pvcauth command

Purpose

The **pvcauth** command is used to authenticate with an IBM Power Virtualization Center (PowerVC) and get a token. This token is required to use the PowerVC services for the AIX Live Update operation. This token is valid only for a set time period. This command can also be used to invalidate a token.

Syntax

To authenticate with PowerVC and to get a token, use the following syntax:

```
pvcauth [ -u user_name ] [ -p password ] -a pvc [ -o project ][ -P port ]
```

To invalidate and remove a previously generated token, use the following syntax:

```
pvcauth -r -a pvc
```

To list all the known PowerVC authentication tokens, use the following syntax:

```
pvcauth -l
```

Description

You can use the **pvcauth** command if you have access to all types of object and if you have appropriate PowerVC administrative authority. The **pvcauth** command generates a token that can be used by an AIX partition administrator to perform the Live Update operation. If the command succeeds, a token is stored in the kernel. You can now use the **geninstall** command to perform the Live Update operation.

To use this command, you must have authority to perform the following tasks:

- Power on a managed partition.
- Shut down a managed partition.
- Create a managed partition.
- Remove a managed partition.
- Manage storage volumes.
- Manage network adapters.

Parameters

password
A string of up to 64 characters that specifies a password.

port
A string of up to 16 characters that specifies a port number to contact PowerVC. The default value of this parameter is 5000.

project
A string of up to 64 characters that specifies the PowerVC project name.

pvc
A string of up to 64 characters that specifies either the host name or the IP address of the PowerVC for authentication.

user_name
A string of up to 64 characters that specifies the PowerVC user name.

Flags

-a *pvc*
Specifies the host name or the IP address of PowerVC for authentication.

-o *project*
Specifies a PowerVC project name that is used to authenticate with PowerVC. If you do not specify the **-o** flag, the project name is set to the default name as **ibm-default**.

-l
Lists all the known PowerVC authentication tokens. The information that is listed includes the current Time To Live (TTL) value for the token.

-p *password*
Specifies the PowerVC password for authentication. If you do not specify the **-p** flag, you are prompted for the password after you run the **pvcauth** command.

-P *port*
Specifies a port number that can be used to contact PowerVC.

-r
Removes the token that is generated by PowerVC.

-u *user_name*
Specifies the PowerVC user name that can be used for authentication. You must have access to all types of objects and appropriate PowerVC administrative authority.

Examples

1. To authenticate with an HMC, called `apollo`, which has a firewall and in which the PowerVC port 5000 is not accessible, a rebound proxy node can be set up to use a different port that is open. To authenticate a logical partition called `mylpar` and to use the SSH client with port 14111 on a proxy node that is called `proxy1`, enter the following commands:

   ```
   root @ proxy1: /
   # ssh -R localhost:14111:apollo:5000 root@mylpar

   root @ mylpar: /
   # pvcauth -a localhost -u hscroot -P 14111
   Enter HMC password:
   ```

   You can specify the `management_console` attribute as `localhost` in the `pvc` stanza of the **lvupdate.data** file to initiate the Live Update operation.
2. To authenticate with PowerVC that has an IP address 5.5.55.121 with password prompt, enter the following command:

   ```
   # pvcauth -a 5.5.55.121 -u root
        Enter password for root:
   ```

3. To invalidate a previous authentication with PowerVC that has an IP address 5.5.55.121, enter the following command:

   ```
   # pvcauth -r -a 5.5.55.121
   ```

pvi Command

Purpose

Provides a privileged editor so that you can access privileged files.

Syntax

pvi [**-l**] [**-R**] [**-w** *Number*] [**-c** | **+** [*Subcommand*]] [*File*]

Description

The **pvi** command calls the **pvi** editor, a privileged version of the **vi** editor, to edit the file specified by the *File* parameter. Only one file can be opened at a time, and this file must have the security attributes that are defined in the privileged file database. You can display the file in the editor only when at least one of the authorizations matches at least one of the authorizations in the **readauths** or the **writeauths** attribute for the file. The contents of the buffer can then be modified. You can write to the file using the editor only when at least one of the authorizations matches at least one of the authorizations in the **writeauths** attribute for the file. Files opened by the **pvi** command can only be written to the same path they were opened from.

You enter and leave the **pvi** editor in command mode, but to add or change text, you must enter the text input mode. See the text input mode for information about the subcommands that initiate the text input mode. You can save the text to a file with one of the **:w** commands, and exit the **pvi** editor using the **:q** command.

The full-screen display editor, which is started by the **pvi** command, is based on the **ex** editor. You can use the **ex** subcommands within the **pvi** editor. Subcommands function at the cursor position on the display screen.

The **pvi** editor makes a copy of the file that you are editing in an edit buffer. The contents of the file are not changed until you save the changes.

Note: There are several functions of the **vi** editor that you cannot use with the **pvi** editor. If you refer to the information on the **vi** editor, be aware that the **-r** flag, the **-t** flag, shell escapes, user-defined macros, key mapping, and setting **vi** options permanently are not supported by the **pvi** editor. Only one buffer is opened at a time and a file can only be written to the same path from which it was opened.

Editor Limitations

The maximum limits of the **pvi** editor assume single-byte characters:

- 256 characters per a global command list
- 2048 characters in a shell escape command
- 128 characters in a string-valued option
- 30 characters in a tag name
- 524,230 lines silently enforced
- 128 map macros with 2048 characters total

Editing Modes

The **pvi** editor operates in the following modes:

Item	Description
command mode	The **pvi** editor starts in the command mode. Any subcommand can be called except those that only correct text during the text input mode. To see a description of the subcommands, refer to the topics in "Subcommands for the **pvi** editor". To identify the subcommands that cannot be called from the command mode, refer to "Changing Text While in Input Mode". The **pvi** editor returns to the command mode when the subcommands and other modes end. Press the **Esc** key to cancel a partial subcommand.
text input mode	The **pvi** editor enters the text input mode when you use a permitted command that adds or changes text. To see a list of subcommands that initiate text input mode, refer to "Adding Text to a File" and the subcommands that change text from the command mode, the **C** subcommand and the **cx** subcommands. After entering one of these subcommands, you can edit text with any of the subcommands that function in the text input mode. To see a list of the subcommands, refer to the topics in "Subcommands for the pvi Editor". To return to command mode from text input mode, press **Esc** for a typical exit or press the **Ctrl + C** keys to create an **INTERRUPT** signal.
last line mode	Some subcommands read input on a line displayed at the bottom of the screen. These subcommands include those with the prefix colon (:), slash (/), and question mark (?). When you enter the initial character, the **pvi** editor places the cursor at the bottom of the screen so you can enter the remaining command characters. To run the subcommand, press **Enter**. To cancel the subcommand, press **Ctrl** + **C** to create an **INTERRUPT** signal. When you use the colon (:) to enter the last line mode, the following characters have special meaning when used before the commands that specify counts: **%** All lines regardless of the cursor position **$** Last line . Current line

Customizing the pvi Editor

You can customize the **pvi** editor on a temporary basis by following the directions in "Setting vi Editor Options".

Subcommands for the pvi Editor

You can find information about the **vi** editor subcommands that are applicable to the **pvi** editor in the following list:

- **vi** General Subcommand Syntax.
- **vi** Subcommands for Adjusting the Screen.
- Editing Text with the **vi** Editor.
- Manipulating Files with the **vi** Editor.
- Subcommands for Interrupting and Ending the **vi** Editor.

Flags

Item	Description
-c [*Subcommand*]	Carries out the **ex** editor subcommand before the editing begins. This provides a line-oriented text editor. When you specify a null operand for the *Subcommand* parameter, for example, `-c ''`, the editor places the cursor on the last line of the file.
-l	Enters the editor in the list processing (LISP) mode. In this mode, the editor indents appropriately for LISP mode, and the **(,)**, **{, }**, **[[,** and **]]** subcommands are modified to act in LISP. These subcommands place the cursor at the specified LISP function. For more information on the LISP subcommands, refer to "Moving to Sentences, Paragraphs, and Sections".
-R	Sets the **readonly** option to protect the file against overwriting.
-w *Number*	Sets the default window size to the value specified by the *Number* parameter. This is useful when you use the editor over a low-speed line.
+ [*Subcommand*]	Same as the **-c** Subcommand.

Security

Access Control: This command grants the execute (x) access to all users.

Role-Based Access Control: The command grants read access to a file if the user has an authorization that matches one in the **readauths** or the **writeauths** authorization list in the privileged file database. The command only grants the write access to a file if the user has an authorization that matches one in the **writeauths** authorization list in the privileged file database.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To call a privileged editor to edit the **plans** file, enter:

   ```
   pvi plans
   ```

   This command puts the **pvi** editor into the command mode. To add or change text, you must enter the text input mode or use a command accepted in the command mode.
2. To save the text that you create with the **pvi** editor, leave the text input mode by pressing **Esc**, and then enter:

   ```
   :w
   ```
3. To exit the **pvi** editor from the text input mode, press **Esc** to enter the command mode, and then enter:

   ```
   :q!
   ```

   If the editor is already in the command mode, you do not need to press **Esc** before giving the quit (**q!**) command.

File

Item	Description
/usr/bin/pvi	Contains the **pvi** command.

Item	Description
/etc/security/privfiles	Contains the security attributes for the privileged files.

pwchange Command

Purpose

Change user authentication and privacy keys dynamically.

Syntax

pwchange [**-e**] [**-d** *DebugLevel*] [**-p** *Protocol*] [**-u** *KeyUsage*] [**-s**] [*OldPassword* *NewPassword*] [*IPAddress* | *HostName* | *EngineID*]

Description

The **pwchange** command is provided to facilitate dynamic changes of user authentication and privacy keys. Dynamic configuration of authentication and privacy keys is done by doing **set** commands to objects of syntax keyChange. The keyChange syntax provides a way of changing keys without requiring that the actual keys (either new or old) be flowed directly across the wire, which would not be secure. Instead, if an object,such as **usmUserAuthKeyChange** (for example) is to be set, the keyChange value must be derived from the old and new passwords and the engineID of the agent at which the key will be used. The **pwchange** command is used to generate the keyChange values.

The **pwchange** command generates different output, depending on which protocol and what key usage is selected. Keychange values are typically twice as long as the key to be changed.

Flags

Item	Description
-d *DebugLevel*	This flag indicates what level of debug information is desired. Debug tracing is either on or off: 1 causes debug tracing to be generated to the screen of the command issuer (sysout). Debug tracing is off (0) by default.
-e	This flag indicates that the agent for which the keychange value is being defined is identified by engineID rather than by IP address or host name.
-p *Protocol*	This flag indicates the protocols for which the keychange values should be generated. Valid values are: **HMAC-MD5** Generates keychange values for use with the HMAC-MD5 authentication protocol. **HMAC-SHA** Generates keychange values for use with the HMAC-SHA authentication protocol. **all** Generates both HMAC-MD5 and HMAC-SHA keychange values. The default is that keychange values for the HMAC-MD5 protocol are generated.

Item	Description
-s	This flag indicates that output should be displayed with additional spaces to improve readability. By default, data is displayed in a condensed format to facilitate cut-and-paste operations on the keychange values onto command lines in shell scripts.
-u *KeyUsage*	This flag indicates the usage intended for the keychange value. Valid values are: **auth** An authentication keychange value. **priv** A privacy keychange value. **all** Both authentication and privacy keychange values. **Note:** There is no difference between a keychange value generated for authentication and a keychange value generated for privacy. However, the length of privacy keychange values depends on whether the keychange value is localized.

Parameters

Item	Description
EngineID	Specifies the engineID (1-32 octets, 2-64 hex digits) of the destination host at which the key is to be used. The engineID must be a string of 1-32 octets (2-64 hex digits). The default is that the agent identification is not an engineID.
HostName	Specifies the destination host at which the key is to be used.
IPAddress	Specifies an IPv4 or an IPv6 address of the agent at the destination host at which the key is to be used.
NewPassword	Specifies the password that will be used in generating the new key. The password must be between eight and 255 characters long.
OldPassword	Specifies the password that was used in generating the key originally. The password must be between eight and 255 characters long.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

The **pwchange** command generates different output depending on which protocol and what key usage is selected. Key change values are typically twice as long as the key to be changed.

1. The following command demonstrates how the **pwchange** command can be used:

   ```
   pwchange oldpassword newpassword 9.67.113.79
   ```

   The output of this command looks similar to:

   ```
   Dump of 32 byte HMAC-MD5 authKey keyChange value:
     3eca6ff34b59010d262845210a401656
     78dd9646e31e9f890480a233dbe1114d
   ```

   The value to be set should be passed as a hex value with the **clsnmp** command (all on one line):

   ```
    clsnmp set usmUserAuthKeyChange.12.0.0.0.2.0.0.0.0.9.67.113.79.2.117.49
    \'3eca6ff34b59010d262845210a40165678dd9646e31e9f890480a233dbe1114d\'h
   ```

   Note: The backslash in the preceding example is required before the single quotation mark to enable AIX to correctly interpret the hexadecimal value.

   The index of the usmUserTable is made up of the EngineID and the ASCII representation of the user name. In this case it is 2 characters long and translates to 117.49.

   Note: pwchange incorporates a random component in generating keys and keyChange values. The output from multiple commands with the same input does not produce duplicate results.

2. The following command demonstrates how the **pwchange** command can be used with IPv6 address:

   ```
   pwchange oldpassword newpassword 2000:1:1:1:209:6bff:feae:6d67
   ```

   The output of this command looks similar to:

   ```
   Dump of 32 byte HMAC-MD5 authKey keyChange value:
     0000774adc53ba4b0427dc2f65568435
     721847d1b5cb597daa85d003033afba3
   ```

   The value to be set should be passed as a hex value with the **clsnmp** command (all on one line):

   ```
   clsnmp set usmUserAuthKeyChange.21.128.0.0.2.2.32.0.0.1.0.1.0.1.2.9.107.255.254.174.
   109.103.6.105.112.118.54.117.49
   \'36133c69415502662063761f835ef616de294f37f758c74ff1544ca3de279b8\'h
   ```

   Note: The backslash in the preceding example is required before the single quotation mark to enable AIX to correctly interpret the hexadecimal value.

   The index of the usmUserTable is made up of the EngineID, in this case 21 octets: 128.0.0.2.2.32.0.0.1.0.1.0.1.2.9.107.255.254.174.109.103; And the ASCII representation of the user name, in this case it is 6 characters long and translates to 105.112.118.54.117.49.

   Note: The **pwchange** command incorporates a random component in generating keys and keyChange values. The output from multiple commands with the same input does not produce duplicate results.

pwck Command

Purpose

Verifies the correctness of local authentication information.

Syntax

pwck

Description

The **pwck** command verifies the correctness of the password information in the user database files by checking the definitions for all users. The **pwck** command internally calls the **pwdck** command with **-n** and **ALL** options.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To verify that all the users and administrators exist in the user database, and have any errors reported (but not fixed), enter:

```
pwck
```

Files

Item	Description
/usr/bin/pwck	Contains the **pwck** command.

pwd Command

Purpose

Displays the path name of the working directory.

Syntax

pwd [**-L** | **-P**]

Description

The **pwd** command writes to standard output the full path name of your current directory (from the root directory). All directories are separated by a / (slash). The root directory is represented by the first /, and the last directory named is your current directory.

Flags

-L
Displays the value of the PWD environment variable if the PWD environment variable contains an absolute path name of the current directory that does not contain the file names **.** (dot) or **..** (dot-dot). Otherwise, the **-L** flag behaves the same as the **-P** flag.

-P
Displays the absolute path name of the current directory. The absolute path name displayed with the **-P** flag does not contain file names that, in the context of the path name, refer to files of type symbolic link.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

Entering:

```
pwd
```

displays the current directory as:

```
/home/thomas
```

Files

Item	Description
/usr/bin/pwd	Contains the **pwd** command.

pwdadm Command

Purpose

Administers users' passwords.

Syntax

pwdadm [**-R** *load_module*] [**-f** *Flags* | **-q** | **-c**] *User*

Description

The **pwdadm** command administers users' passwords. The root user or a member of the security group can supply or change the password of the user specified by the *User* parameter. The invoker of the command must provide a password when queried before being allowed to change the other user's password. When the command executes, it sets the **ADMCHG** attribute. This forces the user to change the password the next time a **su** command is given for the user.

Note: The behavior described for this command is for a local user. For users defined in a remote domain, attributes will be retrieved and stored in the remote domain rather than in the local files.

Root users and members of the security group should not change their personal password with this command. The **ADMCHG** attribute would require them to change their password again the next time a **login** command or an **su** command is given for the user. Only the root user or a user with PasswdAdmin authorization can change password information for administrative users, who have the **admin** attribute set to true in the **/etc/security/user** file.

Only the root user, a member of the security group, or a user with PasswdManage authorization can supply or change the password of the user specified by the *User* parameter.

When this command is executed, the `password` field for the user in the **/etc/passwd** file is set to ! (exclamation point), indicating that an encrypted version of the password is in the **/etc/security/passwd** file. The **ADMCHG** attribute is set when the root user or a member of the security group changes a user's password with the **pwdadm** command.

A new password must be defined according to the rules in the **/etc/security/user** file, unless the **-f NOCHECK** flag is included. Only 7-bit characters are supported in passwords. By including the **-f** flag with the **pwdadm** command, the root user or a member of the security group can set attributes that change the password rules. If there is no password entry in the **/etc/security/passwd** file when the **-f** flag is used, the `password` field in the **/etc/passwd** file is set to ! (exclamation point) and an * (asterisk) appears in the `password=` field to indicate that no password has been set.

The **-q** flag permits the root user or members of the security group to query password information. Only the status of the **lastupdate** attribute and the **flags** attribute appear. The encrypted password remains hidden.

The **-c** flag clears all password flags for the user.

Flags

Item	Description
-c	Clears all password flags for the user.
-f *Flags*	Specifies the **flags** attribute of a password. The *Flags* variable must be from the following list of comma-separated attributes: **NOCHECK** Signifies that new passwords need not follow the guidelines established in the **/etc/security/user** file for password composition. **ADMIN** Specifies that password information may be changed only by the root user. Only the root user can enable or disable this attribute. **ADMCHG** Resets the **ADMCHG** attribute without changing the user's password. This forces the user to change passwords the next time a **login** command or an **su** command is given for the user. The attribute is cleared when the user specified by the *User* parameter resets the password.
-q	Queries the status of the password. The values of the **lastupdate** attribute and the **flags** attribute appear.
-R *load_module*	Specifies the loadable I&A module that is used to change the user's attributes.

Security

Access Control: Only the root user and members of the security group should have execute (x) access to this command. The command should have the **trusted computing base** attribute and be **setuid** to the root user to have write (w) access to the **/etc/passwd** file, the **/etc/security/passwd** file, and other user database files.

Files Accessed:

Mode	File
rw	**/etc/passwd**
rw	**/etc/security/passwd**
r	**/etc/security/user**

Auditing Events:

Event	Information
PASSWORD_Change	user
PASSWORD_Flags	user, flags

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To set a password for user susan, a member of the security group enters:

   ```
   pwdadm susan
   ```

   When prompted, the user who invoked the command is prompted for a password before Susan's password can be changed.
2. To query the password status for user susan, a member of the security group enters:

   ```
   pwdadm -q susan
   ```

   This command displays values for the **lastupdate** attribute and the **flags** attribute. The following example shows what appears when the **NOCHECK** and **ADMCHG flags** attributes are in effect:

   ```
   susan:
           lastupdate=
           flags= NOCHECK,ADMCHG
   ```

Files

Item	Description
/usr/bin/pwdadm	Contains the **pwdadm** command.
/etc/security/passwd	Contains password information.

html

pwdck Command

Purpose

Verifies the correctness of local authentication information.

Syntax

pwdck { **-p** | **-n** | **-t** | **-y** } [**-l**]{ **ALL** | *User ...* }

Description

The **pwdck** command verifies the correctness of the password information in the user database files by checking the definitions for **ALL** the users or for the users specified by the *User* parameter. If more than one user is specified, there must be a space between the names.

Note: This command writes its messages to **stderr**.

You must select a flag to indicate whether the system should try to fix erroneous attributes. The following attributes are checked for locally defined users in the **/etc/passwd** file:

Item	Description
entry	Ensures that each entry is readable and that it contains at least two : (colons). If you indicate that the system should fix errors, the entire entry is discarded.
passwd	Ensures that the password field is an ! (exclamation point). If you indicate that the system should fix errors, it transfers the information in the password field to the **/etc/security/passwd** file, updates the **lastupdate** attribute in the **/etc/security/passwd** file, and then replaces the password field in the **/etc/passwd** file with an !. In general, passwords are required if the **minalpha**, **minother**, or **minlen** password restriction is set to a nonzero value in the **/etc/security/user** file.
user	Ensures that the user name is a unique string of 8 bytes or less. It cannot begin with a + (plus sign), a : (colon), a - (minus sign), or a ~ (tilde). It cannot contain a : (colon) in the string and cannot be the **ALL**, **default**, or ***** keywords. If you indicate that the system should fix errors, it removes this user's entry line from the **/etc/passwd** file. If the user name starts with a + or a - symbol, the user is not locally defined, and checks are not performed.

Attributes checked in the **/etc/security/passwd** file are:

Item	Description
line	Ensures that each line is readable and is part of a stanza. Any invalid line is discarded.
password	Ensures that the **password** attribute exists and is not blank, if passwords are required on the system. If you indicate that the system should fix errors, the password is set to ***** (asterisk), and the **lastupdate** attribute is discarded. In general, passwords are required if either of the **minalpha** or **minother** password restrictions are set to nonzero values in the **/etc/security/user** file. If a user's **flags** attribute specifies the **NOCHECK** keyword, a password is not required for this user, and the check is ignored.
lastupdate	Ensures that the **lastupdate** attribute exists for a valid non-blank password, and that its time is prior to the current time. If you indicate that the system should fix errors, the **lastupdate** attribute is discarded or updated, depending on the **password** attribute. The **lastupdate** attribute is discarded if the **password** attribute doesn't exist, or equals a blank or an * (asterisk). Otherwise, the **lastupdate** time is set to the current time.
flags	Ensures that the **flags** attribute contains only the keywords **ADMIN**, **ADMCHG**, and **NOCHECK**. If you indicate that the system should fix errors, it deletes any undefined flags.

Attributes checked in the **/etc/security/user** file are:

Item	Description
auth1	Ensures that each SYSTEM;*username* entry defined for a local user has an *username* entry in the **/etc/security/passwd** file. If you indicate that the system should fix errors, a stanza is added to the **/etc/security/passwd** file for each missing entry, in the following format: `username:` `password = *` If a user's entry and a default entry both are missing from the **/etc/security/user** file, the system assumes the following values and the check on auth1 is performed: `auth1 = SYSTEM;user` **Note:** The **auth1** attribute is deprecated and should not be used.

Item	Description
auth2	Ensures that each authname;*username* entry defined for a local user has an *username* entry in the **/etc/security/passwd** file. If you indicate that the system should fix errors, an entry is added for each missing entry. If a user's entry and a default entry both are missing from the **/etc/security/user** file, the system assumes the following values and the check on **auth2** is performed: `auth2 = NONE` When ALL is specified, the **pwdck** command ensures that each stanza in the **/etc/security/passwd** file corresponds to an authentication name of a local user as a SYSTEM;*username* entry in the **/etc/security/user** file. If you indicate that the system should fix errors, a stanza which does not correspond to an username entry in the **/etc/security/user** file is discarded from the **/etc/security/passwd** file. The **pwdck** command locks the **/etc/passwd** file and the **/etc/security/passwd** file when it updates them. If either of these files are locked by another process, the **pwdck** command waits a few minutes for the files to be unlocked, and terminates if this does not happen. The **pwdck** command checks to see if the **/etc/passwd** file and the **/etc/security/passwd** file are modified by another process while the current **pwdck** process is running. If you indicate that the system should fix errors, the **pwdck** command updates the **/etc/passwd** file and the **/etc/security/passwd** file, and may overwrite any changes made by the other process. **Note:** The **pwdck** command disables any Extended Access Control Lists (ACLs) on the files when it fixes errors and reports them. The **pwdck** command also checks to see if the database management security files (**/etc/passwd.nm.idx**, **/etc/passwd.id.idx**, **/etc/security/passwd.idx**, and **/etc/security/lastlog.idx**) files are up-to-date or newer than the corresponding system security files. Please note, it is alright for the **/etc/security/lastlog.idx** to be not newer than **/etc/security/lastlog**. If the database management security files are out-of-date, a warning message appears indicating that the root user should run the **mkpasswd** command. Generally, the **sysck** command calls the **pwdck** command as part of the verification of a trusted-system installation. In addition, the root user or a member of the security group can enter the command. **Note:** The **auth2** attribute is deprecated and should not be used.

Flags

Item	Description
-l	Locks file during entire run.
-n	Reports errors but does not fix them.
-p	Fixes errors but does not report them.
-t	Reports errors and asks if they should be fixed.
-y	Fixes errors and reports them.

Security

Access Control: This command should grant execute (x) access to the root user and members of the security group. The command should be **setuid** to the root user, to read and write the authentication information, and have the **trusted computing base** attribute.

Files Accessed:

Mode	File
rw	/etc/passwd
r	/etc/security/user
rw	/etc/security/passwd
r	/etc/security/login.cfg

Auditing Events:

Event	Information
PASSWORD_Check	user, error/fix, status
PASSWORD_Ckerr	file/user, error, status

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To verify that all local users have valid passwords, enter:

   ```
   pwdck -y ALL
   ```

   This reports errors, and fixes them.
2. To ensure that user `ariel` has a valid stanza in the **/etc/security/passwd** file, enter:

   ```
   pwdck -y ariel
   ```

Files

Item	Description
/usr/bin/pwdck	Contains the **pwdck** command.
/etc/security/passwd	Contains actual passwords and security information.
/etc/security/login.cfg	Contains configuration information and password restrictions.

pwtokey Command

Purpose

Converts passwords into localized and non localized authentication and privacy keys.

Syntax

pwtokey [**-e**] [**-d** *DebugLevel*] [**-p** *Protocol*] [**-u** *KeyUsage*] [**-s**] *Password* [*EngineID* | *HostName* | *IPAddress*]

Description

AIX provides a facility called **pwtokey** that allows conversion of passwords into localized and nonlocalized authentication and privacy keys. The **pwtokey** procedure takes as input a password and an identifier of the agent and generates authentication and privacy keys. Since the procedure used by the pwtokey facility is the same algorithm used by the **clsnmp** command, the person configuring the SNMP agent can

generate appropriate authentication and privacy keys to put in the **snmpd.conf** file for a user, given a particular password and the IP address at which the agent will run.

If the IP address or the hostname is specified, the SNMP agent must be an AIX agent. The engineID will be created using a vendor-specific formula that incorporates the IP address of the agent and an enterprise ID representing AIX.

Flags

Item	Description
-d *DebugLevel*	This flag indicates what level of debug information is desired. Debug tracing is either on or off, so a value of 1 causes debug tracing to be generated to the screen of the command issuer (sysout), and a value of 0 specifies that no debug tracing be generated. Debug tracing is off (0) by default.
-e	This flag indicates that the agent for which the key is being defined is identified by engineID rather than by IP address or host name.
-p *Protocol*	This flag indicates the protocols for which the keys should be generated. Valid values are: **HMAC-MD5** Generates keys for use with the HMAC-MD5 authentication protocol. **HMAC-SHA** Generates keys for use with the HMAC-SHA authentication protocol **all** Generates both HMAC-MD5 and HMAC-SHA keys. The default is that keys for the HMAC-MD5 protocol are generated.
-s	This flag indicates that output data should be displayed with additional spaces to improve readability. By default, data is displayed in a condensed format to facilitate cut-and-paste operations on the keys into configuration files or command lines.
-u *KeyUsage*	This flag indicates the usage intended for the key. Valid values are: **auth** An authentication key. **priv** A privacy key. **all** Both authentication and privacy keys. **Note:** There is no difference between a key generated for authentication and a key generated for privacy. However, the length of privacy keys depends on whether the key is localized or not.

Parameters

Item	Description
EngineID	Specifies the engineID of the SNMP agent at which the key will be used. The engineID is determined at SNMP agent initialization from the snmpd.boots file. The engineID must be a string of 1-32 octets (2-64 hex digits). The default is that the agent identification is not an engineID.
HostName	Specifies the SNMP agent at which the key will be used on an SNMP request.
IPAddress	Specifies an IPv4 or an IPv6 address of the SNMP agent at which the key will be used on an SNMP request.
Password	Specifies the text string to be used in generating the keys. The password must be in the range of 8-255 characters long. In general, while any printable characters can be used in the passwords, the AIX shell may interpret some characters rather than passing them to the pwtokey command. Include passwords in single quotes to avoid interpretation of the characters by the AIX shell. **Note:** This password is not related to the community name (or "password") used with community-based security (SNMPv1 and SNMPv2c). This password is used only to generate keys for user-based security, an entirely different security scheme.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. This example shows a simple invocation of the **pwtokey** command:

   ```
   pwtokey testpassword 9.67.113.79
   ```

   The output from this command looks similar to the following:

   ```
   Display of 16 byte HMAC-MD5 authKey:
    775b109f79a6b71f94cca5d22451cc0e

   Display of 16 byte HMAC-MD5 localized authKey:
    de25243d5c2765f0ce273e4bcf941701
   ```

   As this example shows, **pwtokey** generates two keys—one that is localized (has been tailored to be usable only at the agent identified) and one that has not been localized. Typically, the localized key is used in the configuration for the SNMP agent. The nonlocalized key is used in the configuration for the clsnmp command.

2. The **pwtokey** can be invoked requesting HMAC-SHA keys for both authentication and privacy, as in the following example:

```
pwtokey -p HMAC-SHA -u all testpassword 9.67.113.79
```

The output of this command looks similar to the following:

```
Display of 20 byte HMAC-SHA authKey:
 b267809aee4b8ef450a7872d6e348713f04b9c50

Display of 20 byte HMAC-SHA localized authKey:
 e5438092d1098a43e27e507e50d32c0edaa39b7c

Display of 20 byte HMAC-SHA privKey:
 b267809aee4b8ef450a7872d6e348713f04b9c50

Display of 16 byte HMAC-SHA localized privKey:
 e5438092d1098a43e27e507e50d32c0e
```

The output for the privacy keys is the same as the output for the authentication keys, except that the localized privacy key has been truncated to 16 bytes, as is required for DES.

Note: If encryption is used, it is more secure to use different passwords for authentication and privacy.

3. The following example shows that the **pwtokey** command is using an IPv6 address:

```
pwtokey testpassword 2000:1:1:1:209:6bff:feae:6d67
```

The output from this command looks similar to the following:

```
Display of 16 byte HMAC-MD5 authKey:
 775b109f79a6b71f94cca5d22451cc0e

Display of 16 byte HMAC-MD5 localized authKey:
 2a30fe53690fa6b62dba3f9ea30e11fb
```

As this example shows, the **pwtokey** command generates two keys: one that is localized (has been tailored to be usable only at the agent identified) and one that has not been localized. Typically, the localized key is used in the configuration for the SNMP agent. The non-localized key is used in the configuration for the **clsnmp** command. SNMP agent at which the key will be used on an SNMP request is an IPv6 address.

pxed Command

Purpose

Implements a Preboot Execution Environment (PXE) Proxy Dynamic Host Configuration Protocol (DHCP) server.

Syntax

To start the **pxed** daemon using the system resource controller:

startsrc -s **pxed** [**-a**]

To start the **pxed** daemon without using the system resource controller:

pxed [**-f** *ConfigurationFile*]

Description

The Preboot Execution Environment defines a protocol and mechanism through which network-connected client systems can automatically download boot images from a network server to start their operating system. As an extension to the BOOTP and DHCP protocols, it provides the configuration ability for administrators that are not necessarily DHCP or network administrators to manage the operating systems installed on the PXE-capable client systems.

Like a DHCP server, the PXE Proxy DHCP server provides information needed by a PXE client to locate and download its appropriate boot files from a network server. However, the PXE Proxy DHCP server does not administer client IP addresses or other DHCP client options.

The PXE Proxy DHCP server is intended to be used when the management of the system boot images must be separated from the management of the DHCP addresses and DHCP client network configurations. The **pxed** daemon can be configured to run on a system that is the DHCP server or is not the DHCP server.

Flags

Item	Description
-a	The argument to be supplied.
-f *ConfigurationFile*	Specifies the path and name of the configuration file that is to be used by the server. If unspecified, the default is **/etc/pxed.cnf**.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

The PXE protocol allows for a nonprivileged user to be the administrator of the PXE client boot images because the **pxed** daemon listens to client messages on ports other than the well-known, protected DHCP server port. However, to configure such an environment, the DHCP server must be running on the same server system as the **pxed** daemon, and the file permissions on the **pxed** daemon must be changed for non-root execution.

Files

Item	Description
/usr/sbin/pxed	Contains the PXE Proxy DHCP server daemon.
/usr/sbin/db_file.dhcpo	Implements a database to be used by the PXE Proxy DHCP server and the DHCP server to store, retrieve, and manage configuration information.
/etc/pxed.cnf	The default configuration file for the **pxed** daemon.

q

The following AIX commands begin with the letter *q*.

qadm Command

Purpose

Performs system administration functions for the printer spooling system.

Syntax

qadm { **-G** } | { [**-D** *Printer*] [**-K** *Printer*] [**-U** *Printer*] [**-X** *Printer*] }

Description

The **qadm** command is a front-end command to the **enq** command. This command brings printers, queues, and the spooling system up or down and also cancels jobs. The **qadm** command translates the requested flags into a format that can be run by the **enq** command.

The **qadm** command works only on local print jobs. Remote print is not supported.

> **Note:** You must either have root user authority or belong to the printq group to run this command.

You could also use the System Management Interface Tool (SMIT) **smit qadm** fast path to run this command.

Flags

Item	Description
-D *Printer*	Brings down the printer you name in the *Printer* variable. The **qdaemon** process stops sending jobs to the device. Entering the **qchk -P** *Printer* command, where *Printer* matches the *Printer* variable in the **-D** flag, reports the device is *down*. The **qadm** command allows current jobs to finish before stopping the printer.
-G	Gracefully brings down the queuing system. This flag temporarily interrupts the **qdaemon** process after all currently running jobs on all queues are finished. Use of this flag is the only way to bring the system down without causing such problems as jobs hanging up in the queue.
-K *Printer*	Brings down the printer that you name in the *Printer* variable, ending all current jobs immediately. Jobs remain in the queue and run again when the printer is brought back
-U *Printer*	Brings up the printer that you name in the *Printer* variable. The **qdaemon** process sends jobs to the printer again. Entering the **qchk -P** *Printer* command, where *Printer* matches the *Printer* variable in the **-U** flag, reports the device is *ready*.
-X *Printer*	Cancels all the jobs of the user that executed the command. If you have root user privileges or are a member of the printq group, then all jobs on the queue system will be canceled.

> **Note:** When **-U** and **-D** flags are used together, the **-U** flag has higher priority.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To bring the queuing system down gracefully, enter:

   ```
   qadm  -G
   ```

2. To cancel all of a particular user's jobs on printer `lp0`, or all jobs on printer `lp0` if you are have root user authority, enter:

   ```
   qadm  -X lp0
   ```

3. To bring up the printer `lpd0` attached to queue `lp0`, enter:

   ```
   qadm  -U lp0:lpd0
   ```

Files

Item	Description
/usr/sbin/qdaemon	Contains the **qdaemon** daemon.
/var/spool/lpd/qdir/*	Contains the job description files.
/var/spool/lpd/stat/*	Contains information on the status of the devices.
/var/spool/qdaemon/*	Contains the temporary copies of enqueued files.
/etc/qconfig	Contains the configuration file.
/etc/qconfig.bin	Contains the digested, binary version of the **/etc/qconfig** file.

qcan Command

Purpose

Cancels a print job.

Syntax

qcan [**-X**] [**-x** *JobNumber*] [**-P** *Printer*]

Description

The **qcan** command cancels either a particular job number or all jobs in a print queue.

You could also use the System Management Interface Tool (SMIT) **smit qcan** fast path to run this command.

The **qstatus** was enhanced to improve the administration of local queues showing duplicate 3-digit job numbers. You can use the **-W** flag with the **enq**, **qchk**, **lpstat**, and **lpq** status commands to display more job number digits.

If your queue display shows duplicate 3-digit job numbers, use **qchk -W** to list job numbers with greater precision. You can then cancel a specific job.

For example, `qchk` might display job number 123 twice while, `qchk -W` would display job number 1123 and 2123. If you want to cancel job number 2123, specifying `qcan -x 123`, causes the **qdaemon**

to cancel the first matching job number it finds in its internal list, which may be 1123. By having the additional information that the **-W** flag provides, you can cancel a specific job number.

Flags

Item	Description
-P *Printer*	Specifies the *Printer* where either all jobs or the selected job number will be canceled.
-x *JobNumber*	Specifies that only the job number specified by the *JobNumber* variable be canceled.
-X	Cancels all jobs or all jobs for the specified printer. If you have root user authority, all jobs on that queue are deleted. If you do not have root user authority, only jobs you submitted will be canceled. This flag is only valid for local print jobs.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To cancel all jobs queued on printer `lp0`, enter:

   ```
   qcan  -X  -P lp0
   ```

2. To cancel job number 123 on whatever printer the job is on, enter:

   ```
   qcan  -x 123
   ```

Files

Item	Description
/usr/sbin/qdaemon	Contains the **qdaemon** daemon.
/var/spool/lpd/qdir/*	Contains the job description files.
/var/spool/lpd/stat/*	Contains information on the status of the devices.
/var/spool/qdaemon/*	Contains the temporary copies of enqueued files.
/etc/qconfig	Contains the configuration file.
/etc/qconfig.bin	Contains the digested, binary version of the **/etc/qconfig** file.

qchk Command

Purpose

Displays the status of a print queue.

Syntax

qchk [**-A**] [**-L**] [**-W**] [**-P** *Printer*] [**-#** *JobNumber*] [**-q**] [**-u** *UserName*] [**-w** *Delay*]

Description

The **qchk** command displays the current status information regarding specified print jobs, print queues, or users. Use the appropriate flag followed by the requested name or number to indicate specific status information. If you run the **qchk** command with no flags, the status of the default queue is returned.

You could also use the System Management Interface Tool (SMIT) **smit qchk** fast path to run this command.

Flags

Item	Description
-# *JobNumber*	Requests the status of the job number specified by the *JobNumber* variable. The **qchk** command looks for *JobNumber* on the default queue when the **-#***JobNumber* flag is used alone. To search for *JobNumber* on all queues **-#** flag must be used with the **-A** flag. The **-#** flag may also be used in conjunction with the **-P** *Queue* flag. **Notes:** 1. Specify the **-P** *Queue* to override the default destination printer. 2. If jobs 1, 2, and 3 are in the printer queue, and you specify that you want the status of job 3 while job 1 is running, the status information will show job 1 and job 3, not only job 3. 3. If you specify a job number that does not exist, the system displays the current job number on the queue instead of an error message.
-A	Requests the status of all queues.
-L	Displays information in a long-form mode. If the **-L** and **-W** flags are used simultaneously, the**-L** flag displays status of the print job in a semicolon-separated format.
-P *Printer*	Requests the status of the printer specified by the *Printer* variable.
-q	Requests the status of the default print queue.
-u *UserName*	Requests the status of all print jobs sent by the user specified by the *UserName* variable.
-W	Displays information in a wide-form mode with longer queue names, device names, and job numbers. Larger job number information is supported. If the **-W** and **-L** flags are used simultaneously, the **-W** flag displays the status of the print job in a semicolon-separated format.
-w *Delay*	Updates requested status information at intervals, in seconds, as specified by the *Delay* variable until all print jobs are finished.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the default print queue, enter:

   ```
   qchk -q
   ```

2. To display the long status of all queues until empty, while updating the screen every 5 seconds, enter:

```
qchk -A -L -w 5
```

3. To display the status for printer `lp0`, enter:

```
qchk -P lp0
```

4. To display the status for job number 123, enter:

```
qchk -# 123
```

5. To display the status of all print jobs while restricting the queue status to only printer lp0, enter:

```
qchk -A -P lp0
```

6. To display the wide status of the default print queue, enter:

```
qchk -W -q
```

Files

Item	Description
/usr/sbin/qdaemon	Contains the **qdaemon** daemon.
/var/spool/lpd/qdir/*	Contains the job description files.
/var/spool/lpd/stat/*	Contains information on the status of the devices.
/var/spool/qdaemon/*	Contains the temporary copies of enqueued files.
/etc/qconfig	Contains the configuration file.
/etc/qconfig.bin	Contains the digested, binary version of the **/etc/qconfig** file.

qdaemon Command

Purpose

Schedules jobs enqueued by the **enq** command.

Syntax

qdaemon

Description

The **qdaemon** command is a background process (usually started by the **startsrc** command) that schedules printing jobs enqueued by the **enq** command.

Recommendation: To edit the **/etc/qconfig** file, use the **chque**, **mkque**, **rmque**, **chquedev**, **mkquedev**, and **rmquedev** commands or SMIT. Further, it is recommended to run these commands during slow or off-peak time.

If manual editing of the **/etc/qconfig** file is necessary, you can first issue the **enq -G** command to bring the queuing system and the **qdaemon** to a halt after all jobs are processed. Then you can edit the **/etc/qconfig** file and restart the **qdaemon** with the new configuration.

Security

Privilege Control: Only the root user and members of the printq group should have execute (x) access to this command.

Auditing Event	Information
ENQUE_exec	Queue name, job name, host name, file name, user name

Files

Item	Description
/usr/sbin/qdaemon	Contains the **qdaemon** daemon.
/var/spool/lpd/qdir/*	Contains the job description files.
/var/spool/lpd/pio/@local/fullmsg	Contains a flag file whose existence activates **qdaemon** messages to contain complete information.
/var/spool/lpd/stat/*	Contains information on the status of the devices.
/var/spool/qdaemon/*	Contains the temporary copies of enqueued files.
/etc/qconfig	Contains the configuration file.
/etc/qconfig.bin	Contains the digested, binary version of the **/etc/qconfig** file.

qhld Command

Purpose

Holds and releases a spooled print job.

Syntax

qhld [**-r**] { **-#***JobNumber* [**-P***Queue*] | **-P***Queue* | **-u***User* [**-P***Queue*] }

Description

The **qhld** command holds print jobs in a spooled state. The job to be held is designated by job number, queue, or user name. The **-r** flag releases the hold on the print job.

The **qstatus** was enhanced to improve the administration of local queues showing duplicate 3-digit job numbers. You can use the **-W** flag with the **enq**, **qchk**, **lpstat**, and **lpq** status commands to display more job number digits.

If your queue display shows duplicate 3-digit job numbers, use **qchk -W** to list job numbers with greater precision. You can then hold a specific job.

For example, `qchk` might display job number 123 twice while, `qchk -W` would display job number 1123 and 2123. If you want to hold job number 2123, specifying `qhld -#` 123, causes the **qdaemon** to hold the first matching job number it finds in its internal list, which may be 1123. By having the additional information that the **qstatus -W** provides, you can hold a specific job number.

Flags

Item	Description
-#*JobNumber*	Specifies the print job number to be held.
-P*Queue*	Specifies the print queue to be held.
-r	Releases the print job by number, queue, or user name.
-u*User*	Specifies the name of user whose print jobs are to be held.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To hold the print job number `300`, enter:

```
qhld -#300
```

2. To hold all print jobs on queue `lp0`, enter:

```
qhld -P lp0
```

3. To hold all jobs that belong to user `fred`, enter:

```
qhld -u fred
```

4. To release job number `300`, enter:

```
qhld -#300 -r
```

5. To release all the jobs on queue `lp0`, enter:

```
qhld -Plp0 -r
```

6. To release all jobs that belong to user `fred`, enter:

```
qhld -u fred -r
```

Files

Item	Description
/usr/sbin/qdaemon	Contains the **qdaemon** daemon.
/var/spool/lpd/qdir/*	Contains the job description files.
/var/spool/lpd/stat/*	Contains information on the status of the devices.
/var/spool/qdaemon/*	Contains the temporary copies of enqueued files.
/etc/qconfig	Contains the configuration file.
/etc/qconfig.bin	Contains the digested, binary version of the **/etc/qconfig** file.

qmov Command

Purpose

Moves spooled print jobs to another queue.

Syntax

qmov -m*NewQueue* { **-#***JobNumber* [**-P***Queue*] | **-P***Queue* | **-u***User* [**-P***Queue*] }

Description

The **qmov** command moves spooled print jobs to another print queue. The print job to be moved is identified by job number, queue, or user name. The format of the command requires the queue where the job is to be moved to as the first argument and the name of the job to move as the second argument.

The **qstatus** was enhanced to improve the administration of local queues showing duplicate 3-digit job numbers. You can use the **-W** flag with the **enq**, **qchk**, **lpstat**, and **lpq** status commands to display more job number digits.

If your queue display shows duplicate 3-digit job numbers, use **qchk -W** to list job numbers with greater precision. You can then move a specific job.

For example, `qchk` might display job number 123 twice while, `qchk -W` would display job number 1123 and 2123. If you want to move job number 2123, specifying `qmov -#` 123, causes the **qdaemon** to move the first matching job number it finds in its internal list, which may be 1123. By having the additional information that the **qstatus -W** provides, you can move a specific job number.

Flags

Item	Description
-#*JobNumber*	Specifies the job number of the print job to be moved.
-m*NewQueue*	Specifies the name of the destination print queue.
-P*Queue*	Specifies the present print queue of the job to be moved.
-u*User*	Specifies the name of the user whose print jobs are to be moved.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To move job number 280 to queue `lp0`, enter:

   ```
   qmov -mlp0 -#280
   ```

2. To move all print jobs on queue `lp1` to queue `lp0`, enter:

   ```
   qmov -mlp0 -Plp1
   ```

3. To move all of Mary's print jobs to queue `lp0`, enter:

   ```
   qmov -mlp0 -u mary
   ```

Files

Item	Description
/usr/sbin/qdaemon	Contains the **qdaemon** daemon.
/var/spool/lpd/qdir/*	Contains the job description files.
/var/spool/lpd/stat/*	Contains information on the status of the devices.
/var/spool/qdaemon/*	Contains the temporary copies of enqueued files.
/etc/qconfig	Contains the configuration file.
/etc/qconfig.bin	Contains the digested, binary version of the **/etc/qconfig** file.

qosadd Command

Purpose

Adds a QoS (Quality of Service) Service Category or Policy Rule.

Syntax

To add a Service Category:

qosadd [**-s** *ServiceCategory*] [**-t** *OutgoingTOS*] [**-b** *MaxTokenBucket*] [**-f** *FlowServiceType*] [**-m** *MaxRate*] **service**

To add a Policy Rule:

qosadd [**-s** *ServiceCategory*] [**-r** *ServicePolicyRules*] [**-l** *PolicyRulePriority*] [**-n** *ProtocolNumber*] [**-A** *SrcAddrRange*] [**-a** *DestAddrRange*] [**-P** *SrcPortRange*] [**-p** *DestPortRange*] **policy**

Description

The **qosadd** command adds the specified Service Category or Policy Rule entry in the **policyd.conf** file and installs the changes in the QoS Manager.

Flags

Flags with service add:

Item	Description
-s	The name of the **ServiceCategory** attribute, which is mandatory.
-t	The **OutgoingTOS** attribute, specified as an 8 bit binary number.
-b	The **MaxTokenBucket** attribute, specified in Kb (Kilobits).
-f	The **FlowServiceType** attribute, which is ControlledLoad or Guaranteed.
-m	The **MaxRate** attribute, which is specified in Kbps (Kilobits per second).

Flags with policy add:

Item	Description
-s	The name of the **ServiceCategory** attribute, which is mandatory.
-r	The name of the **ServicePolicyRules** attribute, which is mandatory.
-l	The **PolicyRulePriority** attribute, which is a positive integer.
-n	The **ProtocolNumber** attribute, which is defined in the **/etc/protocols** file.
-A	The **SrcAddrRange** attribute, which is the Source IP address range from a1 to a2 where a2 >= a1.
-a	The **DestAddrRange** attribute, which is the Destination IP address range from i1 to i2 where i2 >= i1.
-P	The **SrcPortRange** attribute, which is the Source Port range from a1 to a2 where a2 >= a1.
-p	The **DestPortRange** attribute, which is the Destination Port range from i1 to i2 where i2 >= i1.

Exit Status

Item	Description
0	Successful completion
Positive Integer	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To add the sc01 service, type:

```
qosadd -s sc01 -t 10000001 -b 81 -f ControlledLoad -m 41 service
```

2. To add the pr01 policy, type:

```
qosadd -s sc01 -r pr01 -l 2 -n 17 -A 9.3.25.1-9.3.25.10 -a  9.3.25.33-9.3.25.33
-p 9001-9010 -P 9000-9000 policy
```

3. To add the sc02 service, type:

```
qosadd -s sc02 -t 10000001 -b 81 service
```

4. To add the pr02 policy, type:

```
qosadd -s sc02 -r pr02 -l 2 -n 17 policy
```

qoslist Command

Purpose

Lists a specific QoS (Quality of Service) Service Category or Policy Rule or lists all of them.

Syntax

To list a Service Category:

qoslist [*ServiceCategory*] **service**

To list a Policy Rule:

qoslist [*ServicePolicyRule*] **policy**

Description

The **qoslist** command lists the specified Service Category or Policy Rule. The **qoslist** command lists all Service Categories or Policy Rules if no specific name is given.

Exit Status

Item	Description
0	Successful completion

Item	Description
Positive Integer	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list the sc01 service, type:

```
qoslist sc01 service
```

2. To list the the pr01 policy, type:

```
qoslist pr01 policy
```

3. To list all of the QoS ServiceCategories, type:

```
qoslist service
```

4. To list all of the QoS PolicyRules, type:

```
qoslist policy
```

qosmod Command

Purpose

Modifies an existing QoS (Quality of Service) Service Category or Policy Rule.

Syntax

To modify an existing Service Category:

qosmod [**-s** *ServiceCategory*] [**-t** *OutgoingTOS*] [**-b** *MaxTokenBucket*] [**-f** *FlowServiceType*] [**-m** *MaxRate*] **service**

To modify an existing Policy Rule:

qosmod [**-s** *ServiceCategory*] [**-r** *ServicePolicyRules*] [**-l** *PolicyRulePriority*] [**-n** *ProtocolNumber*] [**-A** *SrcAddrRange*] [**-a** *DestAddrRange*] [**-P** *SrcPortRange*] [**-p** *DestPortRange*] **policy**

Description

The **qosmod** command modifies the specified Service Category or Policy Rule entry in the **policyd.conf** file and installs the changes in the QoS Manager.

The **qosmod** command clears out all the statistics of the old policy. When a **qosstat** command is executed immediately after **qosmod**, the user may not see all the data connections that were using the older rule shifted to the modified rule. This is because the reclassification of the data connection is delayed until a data packet arrives on that connection.

Note: Modifying the priority or filter spec of the rule only results in reclassification of the data connections which use that particular rule. Connections using other rules maintain their existing classification.

Flags

Flags with service modify:

Item	Description
-s	The name of the **ServiceCategory** attribute, which is mandatory.
-t	The **OutgoingTOS** attribute, specified as an 8-bit binary number.
-b	The **MaxTokenBucket** attribute, specified in Kb (Kilobits).
-f	The **FlowServiceType** attribute, which is ControlledLoad or Guaranteed.
-m	The **MaxRate** attribute, which is specified in Kbps (Kilobits per second).

Flags with policy modify:

Item	Description
-s	The name of the **ServiceCategory** attribute, which is mandatory.
-r	The name of the **ServicePolicyRules** attribute, which is mandatory.
-l	The **PolicyRulePriority** attribute, which is a positive integer.
-n	The **ProtocolNumber** attribute, which is defined in the **/etc/protocols** file.
-A	The **SrcAddrRange** attribute, which is the Source IP address range from a1 to a2, where a2 >= a1.
-a	The **DestAddrRange** attribute, which is the Destination IP address range from i1 to i2, where i2 >= i1.
-P	The **SrcPortRange** attribute, which is the Source Port range from a1 to a2, where a2 >= a1.
-p	The **DestPortRange** attribute, which is the Destination Port range from i1 to i2, where i2 >= i1.

Exit Status

Item	Description
0	Successful completion
Positive Integer	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To modify the sc01 service, type:

```
qosmod -s sc01 -t 10001100 -b 84 -f Guaranteed  service
```

2. To modify the pr01 policy, type:

```
qos -s sc01 -r pr01 -l 10 -n 6 -A 9.3.25.15-9.3.25.20 -a 9.3.25.39-9.3.25.39 -p 9015-9020
policy
```

3. To modify the sc02 service, type:

```
qosmod -s sc02 -t 10001111 service
```

4. To modify the pr02 policy, type:

```
qosmod -s sc02 -r pr02 -l 13 -n 6 policy
```

qosremove Command

Purpose

Removes a QoS (Quality of Service) Service Category or Policy Rule.

Syntax

To remove a Service Category:

qosremove [*ServiceCategory*] **service**

To remove a Policy Rule:

qosremove [*ServicePolicyRule*] **policy**

To remove all the Policies and Service categories installed in the kernel:

qosremove all

Description

The **qosremove** command removes the specified Service Category or Policy Rule entry in the **policyd.conf** file and the associated policy or service in the QoS Manager.

Exit Status

Item	Description
0	Successful completion
Positive Integer	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To remove the sc01 service, type:

```
qosremove sc01 service
```

2. To remove the pr01 policy, type:

```
qosremove pr01 policy
```

qosstat Command

Purpose

Show Quality of Service (QoS) status.

Syntax

qosstat [**-A**] [**-F**] [**-S**]

Description

The **qosstat** command displays information about the installed Qos policies. Using **qosstat** without any flags returns filter/flow specification and statistical information for each installed policy.

Flags

Item	Description
-A	Returns the policy rule handle for each installed policy. A unique handle is assigned by the qoS manager for each policy installed.
-F	Returns the flow and filter specification for each policy installed.
-S	Returns the statistical information for each policy installed.

Examples

1. **qosstat**

```
Policy Rule handle 1:

Filter specification for rule index 1:
        PolicyRulePriority:                     0
        protocol:                   TCP
        source IP addr:             INADDR_ANY
        destination IP addr:        INADDR_ANY
        source port:                80
        destination port:           ANY_PORT
Flow Class for rule index 1:
        service class:     Diff-Serv
        peak rate:         100000000 bytes/sec
        average rate:      128 bytes/sec
        bucket depth:      4096 bytes
        TOS (in profile):  0
        TOS (out profile): 0
Statistics for rule index 1:
        total number of connections:          0
        total bytes transmitted:              0
        total packets transmitted:            0
        total in-profile bytes transmitted:   0
        total in-profile packets transmitted: 0
Policy Rule Handle 2:

Filter specification for rule index 2:
        PolicyRulePriority:                      0
        protocol:                   TCP
        source IP addr:             INADDR_ANY
        destination IP addr:        INADDR_ANY
        source port:                100
        destination port:           ANY_PORT
Flow Class for rule index 2:
        service class:     Diff-Serv
        peak rate:         100000000 bytes/sec
        average rate:      128 bytes/sec
        bucket depth:      4096 bytes
        TOS (in profile):  0
        TOS (out profile): 0
Statistics for rule index 2:
        total number of connections:          0
```

```
        total bytes transmitted:              0
        total packets transmitted:            0
        total in-profile bytes transmitted:   0
        total in-profile packets transmitted: 0
```

2. **qosstat -A**

```
Policy Rule Handle 1:
        rule index:    1

Policy Rule Handle 2:
        rule index:    2
```

3. **qosstat -F**

```
Policy Rule Handle 1:
Filter specification for rule index 1:
        PolicyRulePriority:                    0
        protocol:                   TCP
        source IP addr:             INADDR_ANY
        destination IP addr:        INADDR_ANY
        source port:                80
        destination port:           ANY_PORT
Flow Class for rule index 1:
        service class:     Diff-Serv
        peak rate:         100000000 bytes/sec
        average rate:      128 bytes/sec
        bucket depth:      4096 bytes
        TOS (in profile):  0
        TOS (out profile): 0

Policy Rule Handle 2:
Filter specification for rule index 2:
        PolicyRulePriority:                    0
        protocol:                   TCP
        source IP addr:             INADDR_ANY
        destination IP addr:        INADDR_ANY
        source port:                100
        destination port:           ANY_PORT
Flow Class for rule index 2:
        service class:     Diff-Serv
        peak rate:         100000000 bytes/sec
        average rate:      128 bytes/sec
        bucket depth:      4096 bytes
        TOS (in profile):  0
        TOS (out profile): 0
```

4. **qosstat -S**

```
Statistics for rule index 1:
        total number of connections:          0
        total bytes transmitted:              0
        total packets transmitted:            0
        total in-profile bytes transmitted:   0
        total in-profile packets transmitted: 0

Policy Rule Handle 2:
Statistics for rule index 2:
        total number of connections:          0
        total bytes transmitted:              0
        total packets transmitted:            0
        total in-profile bytes transmitted:   0
        total in-profile packets transmitted: 0
```

qpri Command

Purpose

Prioritizes a job in the print queue.

Syntax

qpri -# *JobNumber* **-a** *PriorityNumber*

Description

The **qpri** command prioritizes a job in a print queue by specifying the job number and giving it a priority number.

The **qpri** command works only on local print jobs and the local side of remote queues. Remote print jobs are not supported. Also, you must have root user authority or belong to the printq group to run this command.

You could also use the System Management Interface Tool (SMIT) **smit qpri** fast path to run this command.

The **qstatus** was enhanced to improve the administration of local queues showing duplicate 3-digit job numbers. You can use the **-W** flag with the **enq**, **qchk**, **lpstat**, and **lpq** status commands to display more job number digits.

If your queue display shows duplicate 3-digit job numbers, use **qchk -W** to list job numbers with greater precision. You can then alter the priority of a specific job.

For example, `qchk` might display job number 123 twice while, `qchk -W` would display job number 1123 and 2123. If you want to alter the priority of job number 2123, specifying `qpri -#` 123, causes the **qdaemon** to alter the priority of the first matching job number it finds in its internal list, which may be 1123. By having the additional information that the **qstatus -W** provides, you can alter the priority of a specific job number.

Flags

Item	Description
-# *JobNumber*	Specifies the job number on which to change priority.
-a *PriorityNumber*	Specifies the new priority number for the print job specified by the *JobNumber* variable. The range of priority numbers is 1 through 20, except for the root user or a member of the printq group, who can select priority numbers from 1 through 30.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

To change job number 123 to priority number 18, enter:

```
qpri  -# 123  -a 18
```

Files

Item	Description
/usr/sbin/qdaemon	Contains the **qdaemon** daemon.
/var/spool/lpd/qdir	Contains the job description files.
/var/spool/lpd/stat	Contains information on the status of the devices.
/var/spool/qdaemon	Contains the temporary copies of enqueued files.
/etc/qconfig	Contains the configuration file.
/etc/qconfig.bin	Contains the digested, binary version of the **/etc/qconfig** file.

qprt Command

Purpose

Starts a print job.

Syntax

qprt [**-a** *PreviewOption*] [**-A** *Level*] [**-b** *BottomMargin*] [**-B** *Value*] [**-c**] [**-C**] [**-d** *InputDataType*] [**-D** "*User*"] [**-e** *EmphasizedOpt*] [**-E** *DblHigh*] [**-f** *Filter*] [**-F** *Name*] [**-g** *Begin*] [**-G** *Coord*] [**-h "***Header***"**] [**-H "***HostName***"**] [**-i** *Indent*] [**-I** *FontPath*] [**-j** *Init*] [**-J** *Restore*] [**-k** *Color*] [**-K** *Condense*] [**-l** *Length*] [**-L** *LineWrap*] [**-m** *Message*] [**-M** *MessageFile*] [**-n**] [**-N** *NumberCopies*] [**-O** *PaperHand*] [**-p** *Pitch*] [**-P** *Queue* [**:***QueueDevice*]] [**-Q** *Value*] [**-q** *Quality*] [**-r**] [**-R** *Priority*] [**-s** *NameType*] [**-S** *Speed*] [**-t** *TopMargin*] [**-T "***Title***"**] [**-u** *PaperSrc*] [**-U** *Directional*] [**-v** *LinesPerIn*] [**-V** *Vertical*] [**-w** *PageWidth*] [**-W** *DblWide*] [**-x** *LineFeed*] [**-X** *CodePage*] [**-y** *DblStrike*] [**-Y** *Duplex*] [**-z** *Rotate*] [**-Z** *FormFeed*] [**-#** { **j** | **h** | **v** }] [**-=** *OutputBin*]{ *File* | **-** } ...

Description

The **qprt** command creates and queues a print job to print the file specified by the *File* parameter. To print a file from standard input, specify a **-** (dash) instead of a file name. If you specify multiple files, then they all together make up one print job. The **qprt** command prints the files in the order you specify them.

To print a file, you must have read access to it. Using the **-r** flag you can remove a file after printing it. To remove a file, you must have write access to the directory that contains it. If you want the **qprt** command to notify you when a print job completes, specify the **-n** flag.

You can use the **-B** flag in conjunction with the **-D**, **-H**, and **-T** flags to customize burst pages. Burst pages mark the beginning, end, or both of a print job. To mark the beginning and end of a print job with burst pages, use the **-B aa** flag.

All flags are optional and you can specify them in any order. The **qprt** command ignores spaces between a flag and its argument. You can group flags without arguments after a single **-** (dash). All flags and their arguments must precede the *File* parameter.

You could also use the System Management Interface Tool (SMIT) **smit qprt** fast path to run this command.

Some of the flags and arguments listed in this command are invalid for particular printer types. If you experience problems using an option, you can use SMIT to preview a **qprt** command statement. See the System management interface tool (SMIT) in *General Programming Concepts: Writing and Debugging Programs*. Alternatively, consult your printer manual to find out what options your printer supports.

Flags

Item	Description
-#{j|h|v}	Specifies a special functionality. The possible values for the *Value* variable are: **j** Displays a job number for the specified print job. **h** Queues the print job but holds it in a HELD state. **v** Validates the specified printer backend flag values. As part of the validation process, the command performs legality checking for illegal flag values, type checking, range checking, list checking, and other types of validation. Typically, the validation of backend flag values is useful because illegal flags are identified when the print job is submitted rather than at a later stage when the print job is processed.
-= *OutputBin*	Specifies the output bin destination for a print job. If you do not specify this flag, it uses the default value from the printer driver. The possible values for *OutputBin* are: **0** Top printer bin. **1 - 49** High Capacity Output (HCO) bins 1 - 49. **>49** Printer specific output bins. **Note:** Valid output bins are printer dependent.
-a *PreviewOption*	Previews parameter values for a print job without actually printing any files. You can specify a **0** or a **1** for the *PreviewOption* variable. If you specify **0**, the **qprt** command preview displays normal print processing. If you specify a **1**, the command returns a list of the flag values and the filter pipeline that would be used to convert the input data type to the data type expected by the printer. These flag values are the default command line flag values from the configuration database, overridden by any flag parameters specified on the command line. Only flags that are valid for the *InputDataType* variable specified (or defaulted) for the **-d** flag are shown. Flag values related only to the spooling of your print job, instead of the actual printing, are not shown. The default values for the spooling flags are included with the flag descriptions. The flag values are not checked to verify that they are supported by the printer. The pipeline of filters shows the filter commands (and the flag values passed to the filter commands) that would process the data from your print files before it is sent to the printer. You can review the description for each of the filter commands to determine the type of filtering that is performed.

Item	Description
-A *Level*	Sets the level of diagnostic output. Diagnostic output is useful for diagnosing errors encountered by a filter pipeline that is processing a print file, a header page, or a trailer page. Diagnostic output is mailed to the user who submitted the print job. You can specify one of the following levels: **0** Discards any standard error output produced. **1** Returns flag values, the standard error output, and the complete pipeline that produced any standard error output. **2** Returns the flag values, standard error output (if any), and complete pipelines, regardless of whether an error is detected. If an error is detected, the print job is terminated. **3** Similar to a value of **2**, except that the file is not printed. A value of **1** is recommended. A value of **0** is useful if a filter in a pipeline produces output to standard error, even if no error is encountered (for example, status information). A value of **2** or **3** is useful for diagnosing a problem, even if the problem does not cause any output to standard error.
-b *BottomMargin*	Specifies the bottom margin, the number of blank lines to be left at the bottom of each page.
-B *Value*	Prints burst pages. The *Value* variable consists of a two-character string. The first character applies to header pages. The second character applies to trailer pages. The following values are valid: **a** Always print the (header or trailer) page for each file in each print job. **n** Never print the (header or trailer) page. **g** Print the (header or trailer) page once for each print job (group of files). For example, the **-B ga** flag prints a header page at the beginning of each print job and a trailer page after each file in each print job. **Note:** In a remote print environment, the default is determined by the remote queue on the server.
-c	Copies each print file and prints from the copy. Specify this flag if you plan to modify the print file or files after the **qprt** command is issued, but before the print job completes. If this flag is not specified and the print job is printed on the same node where it was submitted, copies of the print file or files are not made. Printing occurs directly from the file or files you specified with the *File* parameter.

Item	Description
-C	Mails messages generated by your print job to you, even if you are logged in. By default, the **qprt** command displays messages on the console. The **-C** flag only applies to local print jobs. If you want to be notified when a job sent to a remote printer is completed, use the **-n** flag to receive a mail message. **Note:** You cannot redirect certain messages from the **qdaemon** and the printer backend in any way. They are sent directly to the **/dev/console** file.
-d *InputDataType*	Identifies the input data type of the file or files to print. Based on the input data type and the data type expected by the printer, the print files are passed through filters (if necessary) before being sent to the printer. You can specify any of the following input data types: **a** Extended ASCII **c** PCL **d** Diablo 630 **g** Hewlett-Packard GL **p** Pass-through (sent to printer unmodified) **s** PostScript If the printer you select does not support the specified input data type, and if filters are not available to convert the data type of your print file or files to a data type supported by the printer, the print job terminates with an error message.
-D *"User"*	Labels the output for delivery to *User*. Normally the output is labeled for delivery to the user name of the person issuing the **qprt** command request. The value of *User* must be a single word meeting the same requirements of a regular user ID.
-e *EmphasizedOpt*	Sets emphasized print to one of the following: **+** Use emphasized print. **!** Do not use emphasized print.
-E *DblHigh*	Sets double-high print to one of the following: **+** Use double-high print. **!** Do not use double-high print.

Item	Description
-f *Filter*	Identifies the filter to pass your print files through before sending them to the printer. The identifiers are similar to the filter flags available with the **lpr** command. The available filter identifiers are **p**, which invokes the **pr** filter, **n**, which processes output from the **troff** command, and **l**, which allows control characters to be printed.
-F *Name*	Specifies the list of X font files containing the image of characters to be used for printing. Items in the list must be separated by commas. The *Name* parameter value can be full path names, font alias names, or XLFD names. The **-F** Name flag is effective only for MBCS printer queues.
-g *Begin*	Sets the page number to begin printing. This flag is recognized only if the print files are to be formatted (for example, with the **-d a** flag). It is not recognized for pass-through (the **-d p** flag), PostScript (the **-d s** flag), and other types of data that are already formatted.
-G *Coord*	Indicates how to print pages on laser printers that cannot print to the edge of the paper. Use one of the following for the *Coordinate* variable: **+** Whole page coordinate system **!** Print page coordinate system
-h "*Header***"**	Specifies the header text for use by the **pr** command when the **-f p** flag is also specified. If this flag is not specified, the **pr** command uses the print file name as the header. This flag is useful if you also specified the **-c** flag. With the **-c** flag, the print file name used by the **pr** command as the default header is the name of a temporary file generated by the spooler, instead of the file name you specified with the **qprt** command.
-H "*HostName***"**	Sets the host name on the header page.
-i *Indent*	Indents each line the specified number of spaces. You must include the *Indent* variable in the page width specified by the **-w** flag.
-I *FontID*	(uppercase i) Specifies a font identifier. Specifying a font identifier overrides the pitch (the **-p** flag) and type style (the **-s** flag). The **-I***FontID* command is effective for single byte code set print queues only.
-I*FontPath*	(uppercase i) Specifies the comma-separated list of font paths required for the **-F** flag when the font files are designated with a font alias name or an XLFD name. The *FontPath* flag is effective only for MBCS printer queues.
-j *Init*	Initializes the printer before each file is printed. You can specify any of the following: **0** No initialization **1** Full initialization **2** Emulator selection only

Item	Description
-J *Restore*	Restores the printer at the end of the print job. You can specify one of the following: **+** Restore at the end of the print job. **!** Do not restore at the end of the print job.
-k *Color*	Specifies the print color. Typical values are black, red, blue, green, and so on. Consult your printer manual for colors supported and the ribbon position assigned to a particular color.
-K *Condense*	Sets condensed print to one of the following: **+** Use condensed print. **!** Do not use condensed print.
-l *Length*	(lowercase L) Sets the page length. If the *Length* variable is 0, page length is ignored, and the output treated as one continuous page. The page length includes the top and bottom margins and indicates the printable length of the paper.
-L *LineWrap*	Sets line wrap for lines wider than the page width to one of the following: **+** Wrap long lines to the next line. **!** Truncate long lines at the right margin.

Item	Description
-m "*Message***"**	Displays the specified message on the console when the print job is assigned a printer and is ready to begin printing. The print job does not proceed until the message is acknowledged at the console.
-M *MessageFile*	Identifies a file containing message text. This text is displayed on the console when the print job is assigned a printer and is ready to begin printing. The print job does not proceed until the message is acknowledged at the console.
-n	Notifies you when the print job completes. If the **-D "***User***"** flag is also specified, the specified user is notified as well. By default, you are not notified when the print job completes.
-N *NumberCopies*	Specifies the number of copies to print. If this flag is not specified, one copy is printed.
-O *PaperHand*	Sets the type of input paper handling to one of the following: **1** Manual (insert one sheet at a time) **2** Continuous forms **3** Sheet feed

Item	Description
-p *Pitch*	Sets the number of characters per inch. Typical values for *Pitch* are 10 and 12. The actual pitch of the characters printed is also affected by the values for the **-K** (condensed) flag and the **-W** (double-wide) flag. If you are printing an ASCII file on a PostScript printer, this flag determines the character point size. You can specify positive numbers greater than or equal to 1.
-P *Queue*[:*QueueDevice*]	Specifies the print queue name and the optional queue device name. If this flag is not specified, the following conditions occur: • If the **LPDEST** environment variable is set, the **qprt** command uses the queue name specified by the **LPDEST** variable. If set, this value is always used, even if the **PRINTER** variable is also set. • If the **PRINTER** variable is set and no **LPDEST** variable is set, the **qprt** command uses the queue name specified by the **PRINTER** environment variable. Any destination command-line options override both the **LPDEST** and **PRINTER** environment variables. • If neither the **LPDEST** nor the **PRINTER** variable is set, the **qprt** command uses the system default queue name. (The system default queue name is the name of the first queue defined in the **/etc/qconfig** file.) If the *QueueDevice* variable is not specified, the first available printer configured for the queue is used. **Note:** If multiple printers are configured for the same print queue and one or more of the printers is not suitable for printing your files, you should use the *QueueDevice* variable. Otherwise, the spooler assigns the first available printer.
-q *Quality*	Sets the print quality to one of the following: **0** Fast font **1** Draft quality **2** Near letter quality **3** Enhanced quality **300** 300 dots per inch (dpi) **600** 600 dpi
-Q *Value*	Sets the paper size. The *Value* for paper size is printer-dependent. Typical values are: **1** for letter-size paper, **2** for legal, and so on. Consult your printer manual for the values assigned to specific paper sizes.
-r	Removes the print files after the print job completes. If this flag is not specified, the print files are not removed.

Item	Description
-R *Priority*	Sets the priority for the print job. Higher values for the *Priority* variable indicate a higher priority for the print job. The default priority value is **15**. The maximum priority value is **20** for most users and **30** for users with root user privilege and members of the system group (group 0). **Note:** You cannot use this flag when requesting remote print jobs.
-s *NameType*	Specifies a type style with the *NameType* variable. Examples are courier and prestige. The particular type style choices differ depending on the printer type.
-S *Speed*	Sets high-speed printing to one of the following: **+** Use high-speed printing. **!** Do not use high-speed printing.
-t *TopMargin*	Sets the top margin, the number of blank lines left at the top of each page.
-T "*Title***"**	Specifies a print job title with the *Text* variable. If this flag is not specified, the first file name on the **qprt** command line is used as the print job title. The print job title is displayed on the header page and on responses to inquiries about queue status.
-u *PaperSrc*	Sets the paper source to one of the following: **1** Primary **2** Alternate **3** Envelopes
-U *Directional*	Sets unidirectional printing to one of the following: **+** Use unidirectional printing. **!** Do not use unidirectional printing.
-v *LinesPerIn*	Sets the line density to a number of lines per inch. Typical values for the *LinesPerIn* variable are **6** and **8**.
-V *Vertical*	Sets vertical printing to one of the following: **+** Use vertical printing. **!** Do not use vertical printing.
-w *PageWidth*	Sets the page width in number of characters. The page width must include the number of indention spaces specified with the **-i** flag.

Item	Description
-W *DblWide*	Sets double-wide print to one of the following: **+** Use double-wide print. **!** Do not use double-wide print.
-x *LineFeed*	Specifies automatic line feed or automatic carriage return: **0** Do not change line feeds, vertical tabs, and carriage returns. **1** Add a line feed for each carriage return. **2** Add a carriage return for each line feed and each vertical tab.
-X *CodePage*	Provides the code page name. Valid values for the *CodePage* variable are ISO8859-1 through ISO8859-9, IBM-943, IBM-eucJP, IBM-eucKR, IBM-eucTW, and UTF-8. The code page in the user's locale definition is the default.
-y *DblStrike*	Sets double-strike print to one of the following: **+** Use double-strike print. **!** Do not use double-strike print.
-Y *Duplex*	Sets duplexed output. Duplexed output uses both the front and back of each sheet of paper for printing. You can set one of the following: **0** Simplex **1** Duplex, long edge binding **2** Duplex, short edge binding
-z *Rotate*	Rotates page printer output the number of quarter-turns clockwise as specified by the *Value* variable. The length (**-l**) and width (**-w**) values are automatically adjusted accordingly. **0** Portrait **1** Landscape right **2** Portrait upside-down **3** Landscape left

Item	Description
-Z *FormFeed*	Sends a form feed to the printer after each print file. You can specify either of the following: **+** Send a form feed command. **!** Do not send a form feed command to the printer. Use this option carefully since it can result in the next print job beginning on the last output page generated by this print job. Printers printing on continuous forms cannot determine the top of the form for subsequent pages.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To queue the `myfile` file to print on the first available printer configured for the default print queue using the default values, enter:

   ```
   qprt myfile
   ```

2. To queue a file on a specific queue, to print the file without using nondefault flag values, and to validate the flag values at the time of the print job submission, enter:

   ```
   qprt  -f p  -e +  -P fastest  -r  -n  -C  -#v somefile
   ```

   This command line passes the `somefile` file through the **pr** command (the **-f p** flag) and prints it using emphasized mode (the **-e +** flag) on the first available printer configured for the queue named **fastest** (the **-P fastest** flag). The **-#v** flag verifies that all flags associated with this command are valid before passing the print job to the printer backend. After the file is printed, it is removed (the **-r** flag), and the user who submitted the print job is notified (the **-n** flag) by mail (the **-C** flag) that the print job completed.

3. To print `myfile` on legal size paper, enter:

   ```
   qprt  -Q2 myfile
   ```

4. To enqueue the `myfile` file and return the job number, enter:

   ```
   qprt -#j myfile
   ```

5. To queue `MyFile` and hold it, enter:

   ```
   qprt -#h MyFile
   ```

Files

Item	Description
/etc/qconfig	Contains the queue and queue device configuration file.
/usr/bin/qprt	Contains the **qprt** command.

qstatus Command

Purpose

Provides printer status for the print spooling system.

Syntax

qstatus [**-#** *JobNumber*] [**-A**] [**-L**] [**-W**] [**-P** *Printer*] [**-e**] [**-q**] [**-u** *UserName*] [**-w** *DelaySeconds*]

Description

The **qstatus** command performs the actual status function for the print-spooling system. This command is never entered on the command line; it is called by the **enq** command. The **qstatus** command generates status information on specified jobs, printers, queues, or users.

The display generated by the **qstatus** command contains two entries for remote queues. The first entry contains the client's local queue and local device name and its status information. The second entry follows immediately; it contains the client's local queue name (again), followed by the remote queue name. Any jobs submitted to a remote queue are displayed first on the local side and are moved to the remote device as the job is processed on the remote machine.

Since the status commands communicate with remote machines, the status display may occasionally appear to hang while waiting for a response from the remote machine. The command will eventually time-out if a connection cannot be established between the two machines.

Flags

All flags are optional. If flags are not specified, the **qstatus** command returns the status of the following:

- The printer specified by the **LPDEST** variable, if the **LPDEST** environment variable is set. If set, this value is always used, even if the **PRINTER** variable is also set.
- The printer specified by the **PRINTER** environment variable, if the **PRINTER** variable is set and no **LPDEST** variable is set.
- The default printer, if neither the **LPDEST** nor the **PRINTER** variable is set.

  Note: Any destination command line options override both the **LPDEST** and the **PRINTER** environment variables.

Item	Description
-# *JobNumber*	Displays current status information for the job specified by the *JobNumber* variable. Normally, the status of all queued jobs is displayed. 1. Specify the **-P** *Queue* to override the default destination printer. 2. If jobs 1, 2, and 3 are in the printer queue, and you specify that you want the status of job 3 while job 1 is running, the status information will show job 1 and job 3, not only job 3. 3. If you specify a job number that does not exist, the system displays the current job number on the queue instead of an error message.
-A	Displays status information on all queues defined in the **/etc/qconfig** file.
-e	Excludes status information from queues that are not under the control of the **qdaemon** command. The status from such queues may appear in different formats. The **-e** flag can be used with any combination of flags.
-L	Displays status information in a long and detailed version. If the **-L** flag and the **-W** flag are used simultaneously, the **-L** flag displays the long status of the print job in a semicolon-separated format.

Item	Description
-P *Printer*	Displays current status information for the printer specified by the *Printer* variable. Normally, the default printer is used, or the value of either the **LPDEST** or **PRINTER** environment variable is used. The **LPDEST** variable always takes precedence over the **PRINTER** variable.
-q	Displays the current status of the default queue. The default queue is specified by the **LPDEST** variable, or if a **LPDEST** value does not exist, by the **PRINTER** environment variable. If neither variable exists, the **qstatus** command uses the first queue listed in the **/etc/qconfig** file.
-u *UserName*	Displays current status information for all jobs submitted by the user specified by the *UserName* variable. Normally, the status of all queued jobs is displayed.
-W	Displays a wide version of the status information with longer queue names, device names, and job numbers. Longer job number information is supported. If the **-L** flag and the **-W** flag are used simultaneously, the **-W** flag displays the long status of the print job in a semicolon-separated format.
-w *DelaySeconds*	Displays requested queue information at intervals specified by the *DelaySeconds* variable. When the queue is empty, the display ends.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the default print queue, enter:

   ```
   qstatus -q
   ```

2. To display the long status of all queues until empty, while updating the screen every 5 seconds, enter:

   ```
   qstatus -A -L -w 5
   ```

3. To display the status for printer `lp0`, enter:

   ```
   qstatus -P lp0
   ```

4. To display the status for job number 123, enter:

   ```
   qstatus -# 123 -P lp0
   ```

5. To display the status of all queues in wide format, enter:

   ```
   qstatus -A -W
   ```

Files

Item	Description
/var/spool/lpd/qdir/*	Contains the job description files.
/etc/qconfig	Contains the configuration file.
/etc/qconfig.bin	Contains the digested, binary version of the **/etc/qconfig** file.
/usr/lib/lpd/rembak	Contains the remote back end.
/usr/lib/lpd/qstatus	Contains the command file.

Item	Description
/var/spool/lpd/stat/*	Contains the status files for the **qstatus** command.

quiz Command

Purpose

Tests your knowledge.

Syntax

quiz { **-i** *File* | **-t** | *Category1 Category2* }

Description

The **quiz** command gives associative knowledge tests on various selectable subjects. It asks about items chosen from *Category1* and expects answers from *Category2*. If you do not specify the categories, the **quiz** command lists the available categories, provides instructions, and returns to the shell prompt.

The game provides the correct answer whenever you press the Enter key. When questions run out or when you press the Interrupt (Ctrl-C) or End Of File (Ctrl-D) key sequences, the game reports a score and ends.

Flags

Item	Description
-i *File*	Substitutes the named *File* for the standard index file. **Note:** In the following syntax description, brackets are normally used to indicate that an item is optional. However, a bold-faced bracket or brace should be entered as a literal part of the syntax. A vertical list of items indicates that one must be chosen. The lines in *File* must have the following syntax: **line = category [:category] . . .** **category = alternate [|alternate] . . .** **alternate = [primary]** **primary = character** **[category]** **option** **option = {category}** In an index file, the first category of each line must specify the name of an information file. The information file contains the names of files with quiz material. The remaining categories specify the order and contents of the data in each line of the information file. The quiz data in an information file follows the same syntax. A \ (backslash) is an escape character that allows you to quote syntactically significant characters or to insert a new-line character (\n) into a line. When either a question or its answer is blank, the **quiz** command does not ask the question. The construct **a|ab** does not work in an information file. Use **a{b}**.
-t	Provides a tutorial. Repeats missed questions and introduces new material gradually.

Examples

1. To start a Latin-to-English quiz, enter:

```
/usr/games/quiz latin english
```

The game displays Latin words and waits for you to enter what they mean in English.

2. To start an English-to-Latin quiz, enter:

```
/usr/games/quiz english latin
```

3. To set up a Latin-English quiz, add the following line to the index file:

```
/usr/games/lib/quiz/latin:latin:english
```

This line specifies that the **/usr/games/lib/quiz/latin** file contains information about the categories Latin and English.

You can add new categories to the standard index file, **/usr/games/lib/quiz/index**, or to an index file of your own. If you create your own index file, run the **quiz** command with the **-i***File* flag and enter your list of quiz topics.

4. The following is a sample information file:

```
cor:heart
sacerdos:priest{ess}
quando:when|since|because
optat:{{s}he |it }[desires|wishes]\|
desire|wish
alb[us|a|um]:white
```

This information file contains Latin and English words. The : (colon) separates each Latin word from its English equivalent. Items enclosed in { } (braces) are optional. A | (vertical bar) separates two items when entering either is correct. The [ ] (brackets) group items separated by vertical bars.

The first line accepts only the answer `heart` in response to the Latin word `cor`. The second accepts either `priest` or `priestess` in response to `sacerdos`. The third line accepts `when`, `since`, or `because` for `quando`.

The \ (backslash) at the end of the fourth line indicates that this entry continues on the next line. In other words, the fourth and fifth lines together form one entry. This entry accepts any of the following in response to `optat`:

```
she desires it desires desire
she wishes it wishes wish
he desires desires
he wishes wishes
```

If you start a Latin-to-English quiz, the last line of the sample information file instructs the **quiz** command to ask you the meaning of the Latin word `albus`. If you start an English-to-Latin quiz, the **quiz** command displays `white` and accepts `albus`, `alba`, or `album` for the answer.

If any of the characters { (left brace), } (right brace),[(left bracket) ,], (right bracket) or | (vertical bar) appear in a question item, the **quiz** command gives the first alternative of every | group and displays every optional group. Thus, the English-to-Latin question for the fourth definition in this sample is `she desires`.

Files

Item	Description
/usr/games/lib/quiz/index	Default index file for quiz categories.
/usr/games/lib/quiz/*	Used to specify the contents of a given file.
/usr/games	Location of the system's games.

quot Command

Purpose

Summarizes file system ownership.

Syntax

quot [**-c**] [**-f**] [**-h**] [**-n**][**-v**] [*FileSystem ...*]

quot -a [**-c**] [**-f**] [**-h**] [**-n**] [**-v**]

Description

The **quot** command summarizes file system ownership for JFS file systems by displaying the number of 512-byte blocks currently owned by each user in the specified file system (*FileSystem*). If no file system is specified, the **quot** command displays the same information for each of the JFS file systems in the `/etc/filesystems` file.

Flags

Item	Description
-a	Generate a report on all mounted systems.
-c	Displays a three-column report. The first column specifies the file size in 512-byte blocks. The second column specifies the number of files of that size. Finally, the third column specifies the cumulative total of 512-byte blocks in all files of that size or smaller. **Note:** Files greater than or equal to 500 blocks are grouped under a block size of 499. However, their exact block count contributes to the cumulative total of blocks.
-f	Displays the total number of blocks, the total number of files, and the user name associated with these totals.
-h	Estimates the number of blocks used by the file. This estimation is based on the file size and may return greater than actual block usage when used on files with holes.
-n	Produces a list of all files and their owners by running the following pipeline: `ncheck filesystem | sort +On | quot -n filesystem`
-v	Displays output in three columns containing the number of blocks not accessed in the last 30, 60, and 90 days.

Security

Access Control: This command is owned by the bin user and bin group.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the number of files and bytes owned by each user in the **/usr** file system, enter:

```
quot -f /usr
```

The system displays the following information:

```
/usr:
63056    3217    bin
20832     390    root
 1184      42    uucp
   56       5    adm
    8       1    guest
    8       1    sys
```

2. To display file size statistics, the number of files of each size, and a cumulative total, enter:

```
quot -c /usr
```

The system displays the following information:

```
/usr:
8       103      824
16      2        856
499     0        856
```

3. To generate a report of all mounted file systems, type:

```
quot -a
```

4. To generate a report of the **/var** file system, type:

```
#quot -v /var
/var:
45695   root            12852   11878   11774
 2569   guest            2567    1280     960
 2121   adm                92      91      91
 1343   bin               465     233     193
   14   uucp                0       0       0
    5   daemon              0       0       0
    1   invscout            1       1       1
    1   nuucp               1       1       1
    1   sys                 0       0       0
```

Files

Item	Description
/etc/passwd	Contains user names.
/etc/filesystems	Contains file system names and locations.

quota Command

Purpose

Displays disk usage and quotas.

Syntax

quota [**-u** [*User*]] [**-g** [*Group*]] [**-v** | **-q**]

Description

The `quota` command displays disk usage and quotas. By default, or with the `-u` flag, only user quotas are displayed. The `quota` command reports the quotas of all file systems listed in the `/etc/filesystems` file. If the `quota` command exits with a non-zero status, one or more file systems are over quota.

A root user may use the `-u` flag with the optional User parameter to view the limits of other users. Users without root user authority can view the limits of groups of which they are members by using the `-g` flag with the optional Group parameter.

Note:

1. In a JFS file system, if a particular user has no files in a file system on which that user has a quota, this command displays `quota: none` for that user. The user's actual quota is displayed when the user has files in the file system, or when the `-v` flag is specified. For JFS2, a user's actual quota is displayed in all cases.
2. In JFS2 systems, because the root user is not limited by quotas, limits for the root user are always displayed as zero (unlimited).
3. The rpc.rquotad protocol does not support the group quota for NFS. Thus, it does not return group quota information for NFS.

Flags

Item	Description
-g	Displays the quotas of the user's group.
-u	Displays user quotas. This flag is the default option.
-v	Displays quotas on file systems with no allocated storage.
-q	Prints a terse message, containing only information about file systems with usage over quota. **Note:** The **-q** flag takes precedence over the **-v** flag.

Security

Access Control: This command is owned by the root user and the bin group.

Privilege Control: This program is **setuid** in order to allow non-privileged users to view personal quotas.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display your quotas as user `keith`, type:

   ```
   quota
   ```

   The system displays the following information:

   ```
   User quotas for user keith (uid 502):
   Filesystem  blocks  quota  limit  grace  Files  quota limit grace
           /u      20     55     60            20     60    65
   ```

2. To display quotas as the root user for user `davec`, type:

   ```
   quota -u davec
   ```

   The system displays the following information:

   ```
   User quotas for user davec (uid 2702):
   Filesystem  blocks  quota  limit  grace  files  quota limit grace
           /u      48     50     60             7     60    60
   ```

Files

Item	Description
quota.user	Specifies user quotas.
quota.group	Specifies group quotas.
/etc/filesystems	Contains file system names and locations.

quotacheck Command

Purpose

Checks file system quota consistency.

Syntax

quotacheck [-d] [**-g**] [**-u**] [**-v**] { **-a** | *Filesystem ...* }

Description

The `quotacheck` command examines a file system specified by the *FileSystem* parameter, builds a table of current disk usage, and compares the information in the table to that recorded in the file system's disk quota files. If any inconsistencies are detected, the quota files are updated. By default, both user and group quotas are checked.

The optional `-g` flag specifies that only group quotas are checked. The optional `-u` flag specifies that only user quotas are checked. Specifying both `-g` and `-u` flags is equivalent to the default behavior which checks both user and group quotas. The `-a` flag specifies that all file systems in the `/etc/filesystem` file with disk quotas enabled are checked.

For both JFS and JFS2 file systems, the optional `-d` flag deletes Usage statistics for any user or group ID that does not exist in `/etc/passwd` or `/etc/group`, and which has no allocation in the file system. The affected users or groups will no longer have statistics displayed by the `repquota` command.

The `quotacheck` command normally operates silently. If the `-v` flag is specified, the `quotacheck` command reports discrepancies between the calculated and recorded disk quotas.

For JFS, the `quotacheck` command determines the quota file names from the `/etc/filesystems` file (by default, the files are named `quota.user` and `quota.group` and are located at the root of the file system); for JFS2, the names and location of these files are predetermined and cannot be changed. If these files do not exist, the `quotacheck` command creates them.

Note: Do not run the `quotacheck` command against an active file system. If the file system has any current activity, running `quotacheck` may result in incorrect disk usage information.

Flags

Item	Description
-a	Checks all file systems with disk quotas enabled in **/etc/filesystems**.
-d	Deletes Usage statistics for undefined IDs with no allocation (both JFS and JFS2).
-g	Checks group quotas only.
-u	Checks user quotas only.
-v	Reports discrepancies between the calculated and recorded disk quotas.

Security

Access Control: Only a user with root user authority can execute this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To check the user and group quotas in the `/usr` file system, type:

```
quotacheck /usr
```

2. To check only the group quotas in the `/usr` file system, type:

```
quotacheck -g /usr
```

Files

Item	Description
quota.usr	Specifies user quotas.
quota.group	Specifies group quotas.
/etc/filesystems	Contains file system names and locations.
/etc/group	Contains basic group attributes.
/etc/passwd	Contains user names.

quotaon or quotaoff Command

Purpose

Turns on and off file system quotas.

Syntax

quotaon [**-g**] [**-u**] [**-v**] { **-a** | *FileSystem ...* }

quotaoff [**-g**] [**-u**] [**-v**] { **-a** | *FileSystem ...* }

Description

The `quotaon` command enables disk quotas for one or more file systems specified by the *FileSystem* parameter. The specified file system must be defined with quotas in the `/etc/filesystems` file, and must be mounted. The `quotaon` command looks for the `quota.user` and `quota.group` files in the root directory of the associated file system, and will return an error if not found.

Note: For JFS only, the default quota file names (`quota.user` and `quota.group`) may be overridden in the `/etc/filesystems` file. The quota files can be external to the quota enabled file system by specifying full paths in the `/etc/filesystems` file. For JFS2 file systems, the file names may not be overridden and must reside in the root directory of the file system.

By default, both user and group quotas are enabled. The `-u` flag enables only user quotas; the `-g` flag enables only group quotas. Specifying both `-g` and `-u` flags is equivalent to the default (no option specified). The `-a` flag specifies that all file systems with disk quotas, as indicated by the `/etc/filesystems` file, are enabled.

The `quotaoff` command disables disk quotas for one or more file systems. By default, both user and group quotas are disabled. The `-a`, `-g`, and `-u` flags operate as with the `quotaon` command. The `-v` flag prints a message for each quota type (user or group) in every file system in which quotas are turned on or off with the `quotaon` and `quotaoff` commands, respectively.

An error (**EPERM**) will be returned if the `quota.user` and `quota.group` files are not owned by user **root** and group **system**. Ownership changes on these files are not permitted while quotas are active.

Flags

Item	Description
-a	Enables or disables all file systems that are read-write and have disk quotas, as indicated by the **/etc/filesystems** file. When used with the **-g** flag, only group quotas in the **/etc/filesystems** file are enabled or disabled; when used with the **-u** flag, only user quotas in the **/etc/filesystems** file are enabled or disabled.
-g	Specifies that only group quotas are enabled or disabled.
-u	Specifies that only user quotas are enabled or disabled.
-v	Prints a message for each file system in which quotas are turned on or off.

Security

Access Control: Only the root user can execute this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To enable user quotas for the `/usr` file system, enter:

```
quotaon -u /usr
```

2. To disable user and group quotas for all file systems in the **/etc/filesystems** file and print a message, enter:

```
quotaoff -v -a
```

Files

Item	Description
quota.user	Specifies user quotas.
quota.group	Specifies group quotas.
/etc/filesystems	Contains file system names and locations.

r

The following AIX commands begin with the letter *r*.

raddbm Command

Purpose

Modifies entries in the local database of RADIUS user-authentication information.

Syntax

`raddbm` [`-a` *Command*] [`-d` *Database_filename*] [`-e` *EAP_type*] [`-i` *Config_filename*] [`-l` *Load_filename*] [`-n`] [`-p`] [`-t` *pwd_expire_wks*] [`-u` *User_ID*] [`-w`]

Description

The `raddbm` command is used to create and modify a local database of user-authentication information. The RADIUS server can be configured to use this database as the source of information it uses to authenticate users.

The local database is stored in a file. Data in the file is in a binary tree format to make searches faster. The database file name is specified in the RADIUS `/etc/radius/radiusd.conf` configuration file and has the default value of `dbdata.bin`. You can modify the file name by editing `radiusd.conf` through SMIT.

Each entry has the following fields:

Item	Description
USERID	Specifies the user's ID.
PASSWORD	Specifies the user's password.
PASSWORD_EXPIRATION	Specifies the password expiration time in number of weeks.
EAP_TYPE	Specifies the EAP type allowed for authentication.

Passwords in the database file are not stored in clear text in order to prevent simple password compromise, but the algorithm used to hide the passwords is not considered to be cryptographically secure. The file, `dbdata.bin`, is protected by `root:` security as the owner and group.

Several operations on the local database are supported by the `raddbm` command, including the following:

- Add a user to the database.

  To add a user, the command form is:

  ```
  raddbm -a ADD -u User_ID -e EAP_type -t pwd_expire_wks
  ```

  The user's password is prompted from standard input.

  The `-e` and `-t` flags are optional. If no value for the `-e` flag is entered, the default value of none is used for EAP_TYPE, meaning EAP packets are ignored for this user. If no value for the `-t` flag is entered, the default value of `0` is used for PASSWORD_EXPIRATION, meaning that password expiration is never checked. The `-p` flag is optional since the `raddbm` command always prompts for a new password when adding a new user.
- Change a user in the database.

  To change the user's information in the local database, type the following:

```
raddbm -a CHANGE -u User_ID -p -e EAP_type -t pwd_expire_wks
```

The `-e`, `-p`, and `-t` flags are optional, but at least one must be specified. If the `-p` flag is used, the `raddbm` command will prompt for the password.

- Delete a user from the database.

  To delete a user's entry from the database, type the following:

  ```
  raddbm -a DELETE -u User_ID
  ```

- List users in the database.

  To list a user's entries in the database, type the following:

  ```
  raddbm -a LIST
  raddbm -a LIST -u User_ID
  raddbm -a LIST -u User_ID -w
  ```

  The `-w` and `-u` flags are optional. If the `-w` flag is specified, all fields in the user's entry are displayed (except the password, which for security reasons is never displayed).

  If the `-u` flag is specified, the user's information is displayed in colon-separated format. If the `-u` flag is not specified, all entries in the database are displayed in column format.

- Create a new database.

  The RADIUS server ships an empty database in `/etc/radius/dbdata.bin`. If a user wants to create a new database, at least one user must be added at the time of creation. The form of the command is the following:

  ```
  raddbm -a ADD -u User_ID -e EAP_type -t pwd_expire_wks -n
  ```

  The user's password is prompted from standard input.

  The `-e` and `-t` flags are optional. They default to `EAP_type=NONE` and no password expiration checking.

- Load a list of users into the database.

  A list of users can be loaded directly into the database using the `-l` flag. A file must be created for each user that has records in it of the form:

  ```
  "userid" "password"
  ```

  The double quotes must be present.

  The file can then be used with the `-l` flag in the following way:

  ```
  raddbm -l filename
  ```

  Placing user passwords in plain text format in a file is strongly discouraged. This option is provided mainly for testing purposes.

Flags

Item	Description
?	Displays the help screen.
-a *Command*	Specifies the action to perform. Values are `ADD`, `LIST`, `DELETE`, or `CHANGE`.
-d *Database_filename*	Specifies the database file name. Used to override the default database file specified in the `radiusd.conf` RADIUS configuration file.
-e *EAP_type*	Specifies the EAP type the user is allowed to use for authentication. Currently, only `EAP-TLS`, `MD5-challenge`, or `none` is supported. The default is **none**.

Item	Description
`-i` *Config_filename*	Specifies the RADIUS configuration file name. Used to override the default `/etc/radius/radiusd.conf` configuration file .
`-l` *Load_filename*	Specifies the file name of the user name and password file to load.
`-n`	Creates a new database file. Valid only with the ADD command option. If this option is used, all previous information in the database is lost.
`-p`	Indicates that the user's password is to be changed. For security reasons, the password is prompted from standard input instead of read from the command line.
`-t` *pwd_expire_wks*	Specifies the number of weeks the user's password is valid. This flag is valid with the ADD and CHANGE commands. The default is 0, indicating no password expiration. Valid values are from 0 to 52.
`-u` *User_ID*	Specifies the user's ID. A valid user ID must be less than 253 characters in length, and can contain letters, numbers, and some special characters. It cannot contain blanks. Duplicate user IDs are not allowed.
`-w`	Generates a long listing of user information.

Exit Status

This command has the following exit values:

Item	Description
0	The command completed successfully.
>0	An error occurred.

Security

Only the root user or a member of the security group can execute this command.

Examples

1. To create a new local RADIUS database, you must add at least one user. To create the database, type the following:

   ```
   raddbm -a ADD -u user01 -n
   ```

   Note: The `-n` option will overwrite the existing database, destroying the previous contents. The database file created will be named the default name as specified in the `/etc/radius/radiusd.conf` RADIUS configuration file.
2. To add a user to the database, type the following:

   ```
   raddbm -a ADD -u user01
   ```

   The default values of EAP_TYPE = "none" and PASSWORD_EXPIRATION = "0" are used.
3. To delete a user from the database, type the following:

   ```
   raddbm -a DELETE -u user01
   ```
4. To change a user's password, type the following:

   ```
   raddbm -a CHANGE -u user01 -p
   ```

   The command prompts for the new password.
5. To display a long listing of all entries in the default database, type the following:

```
raddbm -a LIST -w
```

Passwords are not displayed.

6. To display a particular user's database entry, type the following:

```
raddbm -a LIST -u user01 -w
```

7. To add a list of users from a file, first create the file of users and passwords that has one entry per line and has the form:

```
"userid"  "password"
```

Then type the following:

```
raddbm -l Load_filename
```

Restrictions

The RADIUS daemon must be stopped before the `raddbm` command is run. Use the `radiusctl stop` command to stop the daemon. After you have modified the database, restart the daemon with the `radiusctl start` command.

Implementation Specifics

This command is part of the `radius.base` fileset.

Location

/usr/radius/bin/raddbm

Standard Input

For security reasons, when a user is added to the database, the user's password is read from standard input instead of from the command line.

Standard Error

If the call to the `raddbm` command fails, an information message is written to standard error.

Files

Item	Description
`/usr/radius/bin/raddbm`	Location of the `raddbm` command.
`/etc/radius/raddbm.bin`	The default database file as specified in the `radiusd.conf` file.
`/etc/radius/radiusd.conf`	Specifies the RADIUS configuration values, including the default database file name.

radiusctl Command

Purpose

Starts, stops, or restarts the RADIUS authentication, authorization, and accounting daemons.

Syntax

radiusctl start

radiusctl stop

radiusctl restart

Description

The **radiusctl** command starts, stops, or restarts the RADIUS server daemons used for controlling network authentication, authorization, and accounting.

This command enables full EAP-TLS support in the AIX RADIUS server in conjunction with the OpenSSL package shipped on the AIX Expansion Pack media.

The local user database of the AIX RADIUS server can be updated while the server is running, however, new changes take effect only after you restart the system. The **radiusctl** command also makes this possible.

Note: This command deprecates the old method of starting and stopping the AIX RADIUS server (for example, **startsrc -s radiusd**, **stopsrc -s radiusd**, and so on).

Flags

Item	Description
start	Starts running the RADIUS server. **Note:** If EAP-TLS is enabled through OpenSSL, you are prompted to enter the private key password when you attempt to start or restart the server.
stop	Stops the RADIUS server.
restart	Restarts the RADIUS server whether or not it is currently running. If the server is not running, this flag behaves the same as the **start** flag.

Examples

1. To start running the AIX RADIUS server, enter the following command:

   ```
   radiusctl start
   ```

2. To restart an already running AIX RADIUS server, enter the following command:

   ```
   radiusctl restart
   ```

3. To stop the AIX RADIUS server from running, enter the following command:

   ```
   radiusctl stop
   ```

ranlib Command

Purpose

Converts archive libraries to random libraries.

Syntax

ranlib [**-t**] [**-X** {**32**|**64**|**32_64**}] *Archive ...*

Description

The **ranlib** command converts each *Archive* library to a random library. A random library is an archive library that contains a symbol table.

If given the **-t** option, the **ranlib** command only touches the archives and does not modify them. This is useful after copying an archive or using the **-t** option of the **make** command in order to avoid having the **ld** command display an error message about an out-of-date symbol table.

Flags

Item	Description
-t	Touches the named archives without modifying them.
-X *mode*	Specifies the type of object file **ranlib** should examine. The *mode* must be one of the following: **32** Processes only 32-bit object files **64** Processes only 64-bit object files **32_64** Processes both 32-bit and 64-bit object files The default is to process 32-bit object files (ignore 64-bit objects). The *mode* can also be set with the **OBJECT_MODE** environment variable. For example, **OBJECT_MODE=64** causes **ranlib** to process any 64-bit objects and ignore 32-bit objects. The **-X** flag overrides the **OBJECT_MODE** variable.

Examples

To randomize the archive file `genlib.a`, enter:

```
ranlib genlib.a
```

Files

Item	Description
/usr/ccs/bin/ranlib	Contains the **ranlib** command.

raso Command

Purpose

Manages Reliability, Availability, Serviceability parameters.

Syntax

`raso` [`-p` | `-r`] [**-y**] [`-o` *Tunable* [= *Newvalue*]]

`raso` [`-p` | `-r`] [**-y**] [`-d` *Tunable*]

`raso` [`-p`] [`-r`] [**-y**] `-D`

`raso` [`-p`] [`-r`] [**-F**] `-a`

`raso` `-h` [*Tunable*]

`raso` [**-F**] `-L` [*Tunable*]

`raso` [**-F**] `-x` [*Tunable*]

Note: Multiple `-o`, `-d`, `-x`, and `-L` flags can be specified.

Description

Note: The `raso` command requires root authority.

The `raso` command is used to configure Reliability, Availability, Serviceability tuning parameters. The `raso` command sets or displays the current or next-boot values for all RAS tuning parameters. The `raso` command can also be used to make permanent changes or to defer changes until the next reboot. The specified flag determines whether the `raso` command sets or displays a parameter. The `-o` flag can be used to display the current value of a parameter or to set a new value for a parameter.

Understanding the Effect of Changing Tunable Parameters

Misuse of the `raso` command can cause performance degradation or operating system failure. Before modifying any tunable parameter, you should first carefully read about all of the parameter's characteristics in the Tunable Parameters section in order to fully understand the parameter's purpose. You should then ensure that the Diagnosis and Tuning sections for this parameter actually apply to your situation and that changing the value of this parameter could help improve the performance of your system. If the Diagnosis and Tuning sections both contain only "N/A", it is recommended that you do not change the parameter unless you are specifically directed to do so by AIX development.

Flags

Item	Description
-a	Displays the current, reboot (when used in conjunction with the `-r` flag), or permanent (when used in conjunction with the `-p` flag) values for all tunable parameters, with one tunable parameter per line displayed in pairs as *Tunable* = *Value*. For the permanent option, a value is only displayed for a parameter if its reboot and current values are equal. Otherwise `NONE` is displayed as the value.
-d *Tunable*	Resets *Tunable* to the default value. If *Tunable* needs to be changed (that is, it is currently not set to its default value) and is of type Bosboot or Reboot, or if it is of type Incremental and has been changed from its default value, and the `-r` flag is not used in combination, *Tunable* is not changed and a warning displays.
-D	Resets all tunables to their default values. If any tunables that need to be changed are of type Bosboot or Reboot, or if any tunables that need to be changed are of type Incremental and have been changed from their default value, and `-r` is not used in combination, these tunables are not changed and a warning displays.
-F	Forces restricted tunable parameters to be displayed when the options **-a**, **-L** or **-x** are specified alone on the command line. If you do not specify the **-F** flag, restricted tunables are not included, unless they are specifically named in association with a display option.
-h *Tunable*	Displays help about the `raso` command if no *Tunable* parameter is specified. Displays help about the *Tunable* parameter if a *Tunable* parameter is specified.
-L *Tunable*	Lists the characteristics of one or all tunables, with one tunable displayed per line using the following format: `NAME CUR DEF BOOT MIN MAX UNIT TYPE DEPENDENCIES` `mtrc_commonbufsize 3974 3974 3974 1 5067 4KBpages D mtrc_enabled` `mtrc_enabled 1 1 1 0 1 boolean B` `mtrc_rarebufsize 2649 2649 2649 1 3378 4KB pages D` `...` `where:` `CUR = current value` `DEF = default value` `BOOT = boot value` `MIN = minimal value` `MAX = maximum value` `UNIT = tunable unit of measure` `TYPE = parameter type: D (for Dynamic), S (for Static), R (for Reboot),B (for Bosboot), M (for Mount), I (for Incremental), C (for Connect), and d (for Deprecated)` `DEPENDENCIES = list of dependent tunable parameters, one per line`
-o *Tunable* [=*Newvalue*]	Displays the value or sets *Tunable* to *Newvalue*. If *Tunable* needs to be changed (the specified value is different than current value) and is of type Bosboot or Reboot, or if *Tunable* if it is of type Incremental and its current value is larger than the specified value, and if the `-r` flag is not used in combination, *Tunable* is not changed and a warning displays. If the `-r` flag is used in combination without a new value, the nextboot value for *Tunable* is displayed. If the `-p` flag is used in combination without a new value, a value is displayed only if the current and next boot values for *Tunable* are the same. Otherwise, `NONE` is displayed as the value.
-p	When the `-p` flag is used in combination with the `-o`, `-d`, or `-D` flag, changes apply to both the current and reboot values (in addition to the current value being updated, the `/etc/tunables/nextboot` file is updated). These combinations cannot be used on Reboot and Bosboot type parameters because the current values for these parameters cannot be changed. When the `-p` flag is used with the `-a` or `-o` flag without specifying a new value, values are displayed only if the current and next boot values for a parameter are the same. Otherwise, `NONE` is displayed as the value.

Item	Description
-r	When the `-r` flag is used in combination with the `-o`, `-d`, or `-D` flag, changes apply to reboot values (the `/etc/tunables/nextboot` file is updated). If any parameter of type Bosboot is changed, you are prompted to run the `bosboot` command. When the `-r` flag is used with the `-a` or `-o` flag and a new value is not specified, the next boot values for tunables are displayed instead of the current values.
-x *Tunable*	Lists the characteristics of one or all tunables, with one tunable displayed per line using the following format (spreadsheet format): `Tunable Current Default Reboot Minimum Maximum Unit Type Dependencies` where *Tunable* is the tunable parameter, *Current* is the current value of the tunable parameter, *Default* is the default value of the tunable parameter, *Reboot* is the reboot value of the tunable parameter, *Minimum* is the minimum value of the tunable parameter, *Maximum* is the maximum value of the tunable parameter, *Unit* is the tunable unit of measure, *Type* is the parameter type, and *Dependencies* is the list of dependent tunable parameters. If you make any change (with **-o**, **-d**, or **-D**) to a parameter of type Mount, it results in a warning message that the change is only effective for future mountings. If you make any change (with **-o**, **-d** or **-D**) to a parameter of type Connect, it results in **inetd** being restarted, and a warning message that the change is only effective for future socket connections. If you make any change (with **-o**, **-d**, or **-D**) to a parameter of type Bosboot or Reboot without **-r**, it results in an error message. If you make any change (with **-o**, **-d**, or **-D** but without **-r**) to the current value of a parameter of type Incremental with a new value smaller than the current value, it results in an error message.
-y	Suppresses the confirmation prompt before running the **bosboot** command.

If you make any change (with **-o**, **-d** or **-D**) to a restricted tunable parameter, it results in a warning message that a tunable parameter of the restricted-use type has been modified. If you also specify the **-r** or **-p** options on the command line, you are prompted for confirmation of the change. In addition, at system reboot, the presence of restricted tunables in the **/etc/tunables/nextboot** file, which were modified to a value that is different from their default value (using a command line specifying the **-r** or **-p** options), results in an error log entry that identifies the list of these modified tunables.

You can specify a modified tunable value using the abbreviations K, M, G, T, P and E to indicate units. The following table shows the prefixes and values that are associated with the number abbreviations.

Item	Description	
Abbreviation	**Prefix**	**Power of 2**
K	kilo	2^{10}
M	mega	2^{20}
G	giga	2^{30}
T	tera	2^{40}
P	peta	2^{50}
E	exa	2^{60}

Thus, a tunable value of 1024 might be specified as 1K.

Tunable Parameters Type

All the tunable parameters manipulated by the tuning commands (**no**, **nfso**, **vmo**, **ioo**, **schedo**, and **raso**) have been classified into these categories:

Item	Description
Dynamic	If the parameter can be changed at any time
Static	If the parameter can never be changed
Reboot	If the parameter can only be changed during reboot
Bosboot	If the parameter can only be changed by running bosboot and rebooting the machine

Item	Description
Mount	If changes to the parameter are only effective for future file systems or directory mounts
Incremental	If the parameter can only be incremented, except at boot time
Connect	If changes to the parameter are only effective for future socket connections. The parameters must be of type Bosboot.

For parameters of type Bosboot, whenever a change is performed, the tuning commands automatically prompt the user to ask if they want to execute the **bosboot** command. For parameters of type Connect, the tuning commands automatically restart the **inetd** daemon.

Note that the current set of parameters managed by the **schedo** command only includes Dynamic and Reboot types.

Compatibility Mode

When running the **raso** command in the pre 5.2 compatibility mode that is controlled by the **pre520tune** attribute of **sys** 0, the reboot values for parameters, except for those of type Bosboot, are not considered because in this mode they are not applied at the boot time. See NFS tuning on the client in the *Performance management* guide for detailed information.

In pre 5.2 compatibility mode, setting reboot values to tuning parameters continues to be achieved by imbedding calls to tuning commands in scripts called during the boot sequence. Parameters of type **Reboot** can therefore be set without the **-r** flag, so that existing scripts continue to work.

This mode is automatically turned ON when a machine is MIGRATED to AIX 5.2. For complete installations, it is turned OFF and the reboot values for parameters are set by applying the content of the **/etc/tunables/nextboot** file during the reboot sequence. Only in that mode are the **-r** and **-p** flags fully functional. See Kernel Tuning in the *Performance Tools Guide and Reference* for more information.

Tunable Parameters

For default values and range of values for tunables, refer the **raso** command help (**-h** <*tunable_parameter_name*>).

Item	Description
kern_heap_noexec	**Purpose:** Specifies whether no-execute protection should be enabled for the kernel heap. **Tuning:** With protection enabled, any attempt to execute code in the protected heap will result in a kernel exception.
kernel_noexec	**Purpose:** Specifies whether no-execute protection should be enabled for kernel data regions. **Tuning:** With protection enabled, any attempt to execute code in the protected regions will result in a kernel exception.

Item	Description
mbuf_heap_noexec	**Purpose:** Specifies whether no-execute protection should be enabled for the mbuf heap. **Tuning:** With protection enabled, any attempt to execute code in the protected heap will result in a kernel exception.
mtrc_commonbufsize	**Purpose:** Specifies the memory trace buffer size for common events of Lightweight Memory Trace (LMT) which provides system trace information for First Failure Data Capture (FFDC). **Tuning:** The default value is based on data generation under a reference system-wide activity, hardware, and system characteristics. The range upper limit is based on the hardware and system characteristics and depends on the current value of **mtrc_rarebufsize** because they share the LMT resource. Recorded events are saved in system dump and/or reported through user commands.
mtrc_enabled	**Purpose:** Defines the Lightweight Memory Trace (LMT) state. **Tuning:** Value of 1 means LMT is enabled. To be effective, any change of state requires a subsequent bosboot and system reboot.
mtrc_rarebufsize	**Purpose:** Specifies the memory trace buffer size for rare events of Lightweight Memory Trace (LMT) which provides system trace information for First Failure Data Capture (FFDC). **Tuning:** The default value is based on data generation under a reference system-wide activity, hardware, and system characteristics. The range upper limit is based on the hardware and system characteristics and depends on the current value of **mtrace_commonbufsize** because they share the LMT resource. Recorded events are saved in system dump and/or reported through user commands.
tprof_cyc_mult	**Purpose:** Specifies the Performance Monitor PM_CYC and software event sampling frequency multiplier as a means to control the trace sampling frequency.
tprof_evt_mult	**Purpose:** Specifies the Performance Monitor PM_* event sampling frequency multiplier as a means to control the trace sampling frequency.

Item	Description
tprof_inst_threshold	**Purpose:** Specifies the minimum number of completed instructions between Performance Monitor event samples as a means to control the trace sampling frequency. **Values:** • Default: 1000 • Range: 1 to 2G-1 • Type: Dynamic **Diagnosis:** Not applicable **Tuning:** Not applicable
tprof_evt_system	**Purpose:** Allows or restricts the non-privileged users from using the system-wide Performance Monitor event-sampling. **Values:** • Default: 0 • Range: 0, 1 • Type: Dynamic • Unit: Boolean **Tuning:** With **tprof_evt_system** enabled (value 1), the non-privileged users can use tprof and pmctl commands to perform system-wide Performance Monitor event-sampling. When disabled (value 0), non-privileged users can perform event-sampling for processes started with -y option of tprof and pmctl commands. In the disabled mode, non-privileged users cannot perform event-sampling of kernel and kernel extensions.

Examples

1. To list the current and reboot value, range, unit, type, and dependencies of all tunable parameters managed by the `raso` command, type the following:

```
raso -L
```

2. To turn off the Lightweight Memory Trace, type the following:

```
raso -r -o mtrc_enabled=0
```

3. To display help for mtrc_commonbufsize, type the following:

```
raso -h mtrc_commonbufsize
```

4. To set tprof_inst_threshold to 10000 after the next reboot, type the following:

```
raso -r -o tprof_inst_threshold=10000
```

5. To permanently reset all `raso` tunable parameters to their default values, type the following:

```
raso -p -D
```

6. To list the reboot level for all Virtual Memory Manager tuning parameters, type the following:

```
raso -r -a
```

ras_logger Command

Purpose

Log an error using the errors template.

Syntax

/usr/lib/ras/ras_logger [**-y** *template-file*]

Description

The **ras_logger** command logs one error, provided in standard input, using the error's template to determine how to log the data. The format of the input is the following:

```
error_label
resource_name
64_bit_flag
detail_data_item1
detail_data_item2
...
```

The **error_label** field is the error's label defined in the template. The **resource_name** field is up to 16 characters in length. The **64_bit_flag** field's values are 0 for a 32-bit error and 1 for a 64-bit error. The **detail_data** fields correspond to the **Detail_Data** items in the template.

Flags

Item	Description
-y *template-file*	Specifies a template file other than the **/var/adm/ras/errtmplt** default file.

Examples

1. Log an error. The template is the following:

```
 + FOO:
        Catname = "foo.cat"
        Err_Type = TEMP
        Class = O
        Report = TRUE
        Log    = TRUE
        Alert  = FALSE
        Err_Desc = {1, 1, "Error FOO"}
        Prob_Causes = {1, 2, "Just a test"}
        User_Causes = {1, 2, "Just a test"}
        User_Actions = {1, 3, "Do nothing"}
        Detail_Data = 4, {2, 1, "decimal"} ,DEC
        Detail_Data = W, {2, 1, "hex data"} ,HEX
        Detail_Data = 100, {2, 1, "long string"} ,ALPHA
```

   The ras_logger input in the **tfile** file appears as follows:

```
FOO
resource
0
15
A0
hello world
```

Run the **/usr/lib/ras/ras_logger <tfile** command. This will log the FOO error with **resource** as the resource name. The detail data will consist of 4 bytes set to decimal 15, 4 bytes of hex data set to 0xa0, and the string "hello world". Note that if the value of the 64-bit flag was 1, the hexidecimal data would be 8 bytes set to 0xa0.

2. Multi-item decimal values. The template is the following:

```
+ FOO:
        Catname = "foo.cat"
        Err_Type = TEMP
        Class = O
        Report = TRUE
        Log    = TRUE
        Alert  = FALSE
        Err_Desc = {1, 1, "Error FOO"}
        Prob_Causes = {1, 2, "Just a test"}
        User_Causes = {1, 2, "Just a test"}
        User_Actions = {1, 3, "Do nothing"}
        Detail_Data = 8, {2, 1, "decimal"} ,DEC
        Detail_Data = W, {2, 1, "hex data"} ,HEX
        Detail_Data = 100, {2, 1, "long string"} ,ALPHA
```

The **ras_logger** command enters the following into the **tfile**file:

```
FOO
resource
0
15 -15
A0
hello world
```

Note: The decimal data is normally shown by the **errpt** command as two separate values using 4 bytes each. The input therefore contains 15 and -15. This is how it is shown by the **errpt** command.

rbacqry Command

Purpose

Reports a set of used privileges and authorizations for a process.

Syntax

`/usr/sbin/rbacqry` [`-T` |`-C`] `-n` *programname* [`-i` *auditfile*] `-u` *username* [`-t` *timeperiod*]

`/usr/sbin/rbacqry -c` [`-s`]`-u` *username* `-S`

Description

The **rbacqry** command is used as a monitor utility to enable role based access control (RBAC) for applications. The **rbacqry** command reports the privileges and authorizations used by a program after the program is run. It uses the audit subsystem to log the privileges and authorizations of all processes that are created by the program and its spawning process.

The **rbacqry** command operates when the system is operating in the enhanced RBAC mode. The privileges obtained from this report can be assigned to the `innateprivs` and `inheritprivs` attributes for the application by using the **setsecattr** command, which enables the command for RBAC. You can consolidate the privileges for the children of a process and provide it under `inheritprivs` attribute or have separate entries for the children in the `/etc/security/privcmds` file for RBAC enablement.

Notes:

- The **rbacqry** command depends on the audit report that is generated by the AIX auditing subsystem.
- The `rbac` audit class is added to the`/etc/security/audit/config` file when the `rbacqry –c` command is run. The audit class can be configured manually.

- When you are tracing privileges and authorizations by using this utility, assign the `rbac` audit class to a specific user in the `/etc/security/audit/config` file to avoid creating large audit logs.
- The **rbacqry** command does not suggest or provide any RBAC roles as part of the output. The command provides only the privileges and authorizations used by the specified program.
- When you are tracing shell scripts by using the `rbacqry` tool, the shell interpreter (for example: `#!/usr/bin/ksh`) must be mentioned in the first line of the script that is being traced.

Flags

Item	Description
`-c`	Configures the `/etc/security/audit/config` file with the `rbac` class for the specified user.
`-C`	Provides a set of used privileges and authorization for the process tree in a comma-separated list of the set. This option is mutually exclusive with the –T option.
`-i` *auditfile*	Specifies the audit trail file to be processed by the **rbacqry** command. If not specified, the flag uses the `/audit/trail` file by default.
`-n` *programname*	Specifies the target program name that must be traced for used privileges.
`-s`	Starts the auditing subsystem if it is turned off. Restarts the audit subsystem if it is already on.
`-S`	Prints the output in stanza format.
`-T`	Provides a set of used privileges and authorizations for the processes in a tree format.
`-t` *timeperiod*	Accepts a value that is equal to the number of days from when the used privilege report must be generated from the current system date.
`-u` *username*	Specifies the user name. This option is required to configure the audit events for the user, and to query the process run by the user.

Exit status

Error Value	Descriptor
= 0	Successful completion
> 0	An error

Security

On Trusted AIX systems, only authorized users can run the **restore** command.

Item	Descriptor
`aix.fs.manage.restore`	Required to run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To determine the privileges and authorizations that are used by a program, use one of the following methods:

   a. When a program or application is run by a non-root user for which the **rbacqry** command must be run, complete the following steps:

      i) Enable the program for RBAC temporarily under a root or an authorized user's shell, by running the **setsecattr** command:

      ```
      setsecattr -c accessauths=ALLOW_ALL innateprivs=PV_ROOT secflags=FSF_EPS progpath
      ```

      Note: The program path must be a full pathname of the program for which the **rbacqry** command is run.

      ii) Run `setkst -t cmd` (as root or authorized user) to make the above changes effective.

      iii) Run the **rbacqry** command under a root or authorized user's shell to configure the user for auditing:

      ```
      rbacqry -c -s -u username
      ```

      iv) Run the specified program or application as non-root user.

      v) When the program execution completes, run the **rbacqry** command under a root or authorized user's shell to collect used privileges and authorizations:

      ```
      rbacqry -n program -u username (additional options can also be used)
      ```

      vi) Remove the program entry from the `/etc/security/privcmds` file that was added from step (i) by running the following commands as a root or authorized user:

      ```
      rmsecattr -c progpath; setkst -t cmd
      ```

   b. When a program or application is executed by a root user (as root login or switching to a root by using the **su** command) and for which the **rbacqry** command must be run, complete these steps:

      i) Run the **rbacqry** command under a root or authorized user's shell to configure the user for auditing:

      ```
      rbacqry -c -s -u root
      ```

      ii) Run the specified program or application as a root user.

      iii) When the program execution completes, run the **rbacqry** command under a root or authorized user's shell to collect used privileges and authorizations:

      ```
      rbacqry -n program -u root (additional options can also be used)
      ```

      Note: When tracing a program or application that was executed by switching to a root user by using the **su** command after following steps i and ii, run the **rbacqry** command as follows:

      ```
      rbacqry -n program -u user_name (additional options can also be used)
      ```

2. To determine the privileges and authorizations that are used by the **chfs** command (which was executed by user Scooby with aix authorization) and its spawning processes in a tree-formatted output, run the following command:

```
# rbacqry -n chfs -u scooby -T
CMD                     AUTHORIZATIONS                  USED_PRIVS
------------------------------------------------------------------------------
chfs                    aix.fs.manage.change
|                                                       PV_FS_RESIZE
|
|\extendlv              aix.lvm.manage.extend
|  |                                                    PV_AU_ADMIN            PV_KER_ACCT
|  |
|  |\putlvcb            aix.lvm.manage
```

```
|   |   |                                                    PV_FS_MKNOD             PV_PROC_PRIV
|   |   |                                                    PV_KER_LVM              PV_DEV_QUERY
|   |   |
|   |\lextendlv          aix.lvm.manage.extend
|   |   |                                                    PV_AU_ADD               PV_AU_PROC
|   |   |                                                    PV_FS_MKNOD             PV_PROC_PRIV
|   |   |                                                    PV_KER_ACCT             PV_KER_LVM
|   |   |                                                    PV_DEV_QUERY            PV_SU_UID
|   |   |
|   |   |\savebase        aix.system.boot.create
|   |   |   |                                                PV_AU_PROC              PV_FS_MKNOD
|   |   |   |                                                PV_PROC_PRIV            PV_KER_ACCT
|   |   |   |                                                PV_KER_LVM              PV_DEV_QUERY
|   |   |   |                                                PV_SU_UID
|   |   |   |
|   |   |   |\compress     aix.fs.manage.backup
|   |   |   |  |                                             PV_KER_ACCT             PV_SU_UID
|   |   |   |  |
.....
```

3. To display the privileges and authorizations that are used by the **chfs** command (which was executed by user Scooby with aix authorization) from a different audit trail file, run the following command:

```
# rbacqry -u scooby -n chfs -i /audit/trail_example
CMD           AUTHORIZATIONS               USED_PRIVS
-------------------------------------------------------------------------------
chfs          Used_Auth:                   PV_DAC_O              PV_FS_CHOWN
              aix.fs.manage.change         PV_FS_RESIZE
              Checked_Auths:
```

4. To obtain a comma-separated list of privileges that are used by the **chfs** command (which was executed by user Scooby with aix authorization), run the following command:

```
# rbacqry -n chfs -u scooby -C
CMD                     AUTHORIZATIONS          USED_PRIVS
-------------------------------------------------------------------------------
chfs                    aix.fs.manage.change
                                                PV_FS_RESIZE
extendlv                aix.lvm.manage.extend
                                                PV_AU_ADMIN,PV_KER_ACCT
putlvcb                 aix.lvm.manage

PV_FS_MKNOD,PV_PROC_PRIV,PV_KER_LVM,PV_DEV_QUERY
lextendlv               aix.lvm.manage.extend

PV_AU_ADD,PV_AU_PROC,PV_FS_MKNOD,PV_PROC_PRIV,
                                                PV_KER_ACCT,PV_KER_LVM,PV_DEV_QUERY,PV_SU_UID
savebase                aix.system.boot.create

PV_AU_PROC,PV_FS_MKNOD,PV_PROC_PRIV,PV_KER_ACCT,
                                                PV_KER_LVM,PV_DEV_QUERY,PV_SU_UID
compress                aix.fs.manage.backup
                                                PV_KER_ACCT,PV_SU_UID
.......
```

This output format is useful when the USED PRIVS set is added to the privileged command in the `/etc/security/privcmds` database.

Note: The system authorization and custom authorizations can be traced. If the system authorizations must be displayed in the output, a higher authorization (example aix authorization) must be assigned to the user.

5. To configure the user scooby for auditing, run the following command:
   a. To configure the user and to start the auditing for that user, run the following command:

```
#/usr/sbin/rbacqry -c -s -u scooby
```

   Audit subsystem started.

   b. To configure the user for auditing without restarting the auditing, run the following command:

```
#/usr/sbin/rbacqry -c -u scooby
```

Note: The user scooby is not traced by the auditing subsystem because the auditing is not restarted. An entry for scooby is made in the `/etc/security/audit/config` file. You must restart the auditing subsystem manually to allow the auditing to trace the user, or you must run the **rbacqry** command as follows:

```
#/usr/sbin/rbacqry -c -s -u scooby
```

User scooby already configured for audit. Audit subsystem started

6. To show the following stanza for the **-S** format, run the following command:

```
# rbacqry -u scooby -n chfs -S chfs:
                Used_Auth=aix.fs.manage.change
        Checked_Auths=
        Used_Privs=PV_DAC_O,PV_FS_CHOWN,PV_FS_RESIZE
```

7. To execute the **rbacqry** command without any format options, run the following command:

```
# rbacqry -u scooby -n chfs
CMD           AUTHORIZATIONS                USED_PRIVS
------------------------------------------------------------------------------
chfs          Used_Auth:                    PV_DAC_O              PV_FS_CHOWN
              aix.fs.manage.change          PV_FS_RESIZE
              Checked_Auths:
```

Note: The *checked_Auths* parameter are blank when no *checked Auths* parameters are present. If not the **rbacqry** command displays the *checked_auths* parameters as below:

```
# rbacqry -u scooby -n lsuser
CMD             AUTHORIZATIONS                  USED_PRIVS
------------------------------------------------------------------------------
lsuser          Used_Auth:                      PV_AZ_CHECK          PV_DAC_R
                ALLOW_ALL                       PV_DAC_X
                Checked_Auths:
                aix.security.user.list
                aix.security.user.audit
                aix.security.efs
```

Files

File path	Description
`/audit/trail`	Specifies the audit file to capture the audit logs.

rbactoldif Command

Purpose

Prints certain role-based access control (RBAC) and Domain role-based access control tables that are defined locally to standard output (**stdout**) in the LDIF format.

Syntax

rbactoldif -d *baseDN* [**-s** *tables*]

Description

The **rbactoldif** command reads data from locally defined RBAC tables and prints the result to **stdout** in LDIF format. If redirected to a file, the result can be added to an LDAP server with the **ldapadd** command or the **ldif2db** command.

The **rbactoldif** command reads the **/etc/security/ldap/sectoldif.cfg** file to determine what to name the authorization, role, privileged command, privileged device, and privileged file sub-trees that the data will

be exported to. The **rbactoldif** command only exports data to the AUTHORIZATION, ROLE, PRIVCMD, PRIVDEV, and PRIVFILE types defined in the file. The names specified in the file will be used to create sub-trees under the base distinguished name (DN) specified with the **-d** flag. For more information, see the **/etc/security/ldap/sectoldif.cfg** file in *Files Reference* .

Flags

Item	Description
-d *baseDN*	Specifies the base DN under which the RBAC data is placed.
-s *tables*	Specifies a set of tables to be read. If you do not specify the **-s** flag, all of the RBAC and Domain RBAC tables are read. Specify at least one of the following letters, each representing a table name: **a** Specifies the authorization table. **c** Specifies the privileged command table. **d** Specifies the privileged device table. **e** Specifies the domain table. **f** Specifies the privileged file table. **o** Specifies the domain object table. **r** Specifies the role table. **t** Specifies the trvi table.

Security

The **rbactoldif** command is owned by root and security group, with mode bits 500.

File Accessed

File	Mode
/etc/security/authorizations	r
/etc/security/roles	r
/etc/security/privcmds	r
/etc/security/privdevs	r
/etc/security/privfiles	r
/etc/security/.rbac_ids	r
/etc/security/domains	r
/etc/security/domobjs	r

Examples

1. To export all of the RBAC and Domain RBAC tables to LDIF format with base DN of `cn=aixdata`, use the following command:

```
rbactoldif -d cn=aixdata
```

2. To export only the authorization and role tables with base DN of `cn=aixdata`, use the following command:

```
rbactoldif -d cn=aixdata -s ar
```

3. To export only the `domobjs` tables with base DN of `cn=aixdata`, use the following command:

```
rbactoldif -d cn=aixdata -s o
```

rc Command

Purpose

Performs normal startup initialization.

Syntax

rc

Description

The **rc** command has an entry in the **/etc/inittab** file. The **init** command creates a process for the **rc** command entry in the **/etc/inittab** file. The **rc** command performs normal startup initialization for the system. The contents of **/etc/rc** are installation specific. If all of the necessary operations complete successfully, the file exits with a zero return code that allows the **init** command to start loggers to complete normal initialization and startup.

Note:

1. Many bringup functions such as activating page spaces and mounting filesystems are done by the **rc** command.
2. The root file system is implicitly mounted.

rc.mobip6 Command

Purpose

Enables the system to function as a mobile IPv6 home agent or correspondent node.

Syntax

rc.mobip6 { **start** [**-H**] [**-S**] | **stop** [**-N**] [**-F**] }

Description

The **/etc/rc.mobip6** file is a shell script that, when executed, enables the system to function as a mobile IPv6 home agent or correspondent node. If mobile IPv6 has been configured using system management to start at each system restart, the script will be executed automatically at restart.

Flags

Item	Description
-F	Disables IPv6 forwarding.
-H	Enables the system as a Mobile IPv6 home agent and correspondent node. If this flag is not used, the system will be enabled as a correspondent node only.
-N	Stops the **ndpd-router** daemon.
-S	Enables checking of IP security authentication.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Security

You must have root authority or be a member of the system group to execute this command.

Examples

1. The following example enables the system as a mobile IPv6 home agent and correspondent node:

```
/etc/rc.mobip6 start -H
```

2. The following example enables the system as a mobile IPv6 correspondent node and enables IP security checking:

```
/etc/rc.mobip6 start -S
```

3. The following example disables all mobile IPv6 and IPv6 gateway functionality on the system:

```
/etc/rc.mobip6 stop -N -F
```

4. The following example disables all mobile IPv6 functionality but allows the system to continue functioning as an IPv6 gateway:

```
/etc/rc.mobip6 stop
```

Files

Item	Description
/etc/rc.mobip6	Contains the **rc.mobip6** command.

rc.powerfail Command

Purpose

Handles RPA (RS/6000 Platform Architecture) specific EPOW (Environmental and POwer Warning) events and shuts down the system if needed, as part of EPOW event handling.

Syntax

rc.powerfail [**-h**] | [[**-s**] [**-t** [*mm*]][-c [ss]]]

Description

The `rc.powerfail` command is started by the `/etc/inittab` file when `init` receives a SIGPWR signal from the kernel. The `rc.powerfail` command uses `ioctl()` to determine the state of the system. The `rc.powerfail` command should be called only when an EPOW event has occurred.

The various EPOW events handled by `rc.powerfail` and the corresponding event handling done by `rc.powerfail` are listed in the following table:

EPOW class	Event handling done by rc.powerfail	Example
1 These types of errors are considered non-critical cooling problems by the Operating System.	`rc.powerfail` warns the users currently logged onto the system through a `cron` entry which will be walled every 12 hours until the situation disappears.	Redundant Fan Faults. Internal Thermal Problems.
2 These types of errors are considered non-critical power problems by the Operating System.	`rc.powerfail` warns the users currently logged onto the system through a `cron` entry which will be walled every 12 hours until the situation disappears.	Redundant AC input fault.
3 These events are critical in nature and the system should be powered down as soon as possible.	`rc.powerfail` initiates the system shutdown in 10 minutes unless the user has specified some other wait time through the `-t` option.	Ambient temperature approaching specification limit.
4 These kinds of errors are extreme in nature and need an immediate halting of the system.	`rc.powerfail` is expected to process this event in 20 seconds. In these cases, `rc.powerfail` warns the users currently logged onto the system and then immediately halts the system.	Loss of AC input: All the power sources have lost power.
5, 7 These kinds of errors are extreme in nature and should be handled in terms of micro seconds.	Since they should be handled in micro seconds, `rc.powerfail` will not be handling these events. If `rc.powerfail` gets control in these conditions, it will continue to wait out the wait time period.	All the fan systems have failed, non redundant power fault.

As previously mentioned, in case of EPOW class 3 events, the `rc.powerfail` command is given approximately 10 minutes prior to shut down of the system. The user can alter this time by using the `-t` option on the `/etc/inittab` file's powerfail entry. Prior to the last 60 seconds, any users still logged-on are sent a message telling them how much time remains until shutdown. If, at any time in the last 60 seconds, the event clears, the system shutdown halts and the users are notified that all errors have cleared. If a shutdown is not desired, the user may add the `-s` option to the command in the `/etc/inittab` file.

Also in case of EPOW class 3 events, `rc.powerfail` will allow executing environment-specific scripts (if any) to be executed before system shutdown. These scripts will be located under `/usr/lib/scripts/epow`, and `rc.powerfail` will wait for 10 seconds, by default, for their completion. This wait time can be altered using the `-c` option. The value provided through the `-c` option will be taken as the wait time for these scripts, in seconds.

Flags

Item	Description
-h	Gives an information message containing the power status codes and the resulting action. The **rc.powerfail -h** command shuts down the system if needed, as part of EPOW event handling.
-s	Does not do a system shutdown if there is a power failure in systems with either a battery backup or fan fault. The logged-on users still receive all the appropriate messages, but the actual system shutdown is left up to the system administrator. This flag has no effect if a critical power failure is detected.
-t *mm*	Gives the number of whole minutes until system shutdown in the case of a primary power loss with battery backup or fan fault. This number should be equal to half the length of time guaranteed by the battery backup. This flag has no effect if a critical power failure is detected.
-c *ss*	Gives the number of seconds to wait for the completion of any environment specify third party scripts to be executed by rc.powerfail, at EPOW 3 situations.

Exit Status

If the system shuts down, no exit value is returned. Otherwise, the **rc.powerfail** command returns the following exit values:

Item	Description
0	Normal condition.
1	Syntax error.
2	**halt -q** failed
3	**shutdown -F** failed.
4	An error has occurred. Shut your system down immediately using **shutdown -F**.
5	An undefined state. Call your Service Representative.

Security

Access Control: root only.

Examples

1. To look at the cause of a power status equal to 3, enter:

   ```
   rc.powerfail -h
   ```

2. To block system shutdown when non-critical power failures or fan faults occur, enter:

   ```
   chitab "powerfail::powerfail:/etc/rc.powerfail -s >dev/console 2>&1"
   ```

   The next SIGPWR received by **init** will not cause a system shutdown if a non-critical power failure occurs.

3. To change the time until shutdown to 30 minutes, enter:

   ```
   chitab "powerfail::powerfail:/etc/rc.powerfail -t 30 >/dev/console 2>&1"
   ```

   Assuming the condition is not critical, the next SIGPWR received by `init` will have a 30 minute delay until system shutdown.

Files

Item	Description
html	

rc.wpars Command

Purpose

Automatically starts a workload partition.

Syntax

/etc/rc.wpars

Description

The **/etc/rc.wpars** command invokes the **startwpar** command on all workload partitions with the **autostart** option (**mkwpar/chwpar -A**) enabled. The **/etc/rc.wpars** command runs automatically each time the system starts.

rcp Command

Purpose

Transfers files between a local and a remote host or between two remote hosts.

Syntax

rcp [**-p**] [**-F**] [**-k** *realm*] [**-m**] { { *User***@***Host***:***File* | *Host***:***File* | *File* } { *User***@***Host***:***File* | *Host***:***File* | *File* | *User***@***Host***:***Directory* | *Host***:***Directory* | *Directory* } | [**-r**] { *User***@***Host***:***Directory* | *Host***:***Directory* |*Directory* } { *User***@***Host***:***Directory* | *Host***:***Directory* | *Directory* } }

Description

The **/usr/bin/rcp** command is used to copy one or more files between the local host and a remote host, between two remote hosts, or between files at the same remote host.

Remote destination files and directories require a specified *Host***:** parameter. If a remote host name is not specified for either the source or the destination, the **rcp** command is equivalent to the **cp** command. Local file and directory names do not require a *Host***:** parameter.

Note: The **rcp** command assumes that a : (colon) terminates a host name. When you want to use a : in a filename, use a / (slash) in front of the filename or use the full path name, including the /.

If a *Host* is not prefixed by a *User***@** parameter, the local user name is used at the remote host. If a *User***@** parameter is entered, that name is used.

If the path for a file or directory on a remote host is not specified or is not fully qualified, the path is interpreted as beginning at the home directory for the remote user account. Additionally, any metacharacters that must be interpreted at a remote host must be quoted using a \ (backslash), a " (double quotation mark), or a ' (single quotation mark).

File Permissions and Ownership

By default, the permissions mode and ownership of an existing destination file are preserved. Usually, if a destination file does not exist, the permissions mode of the destination file is equal to the permissions

mode of the source file as modified by the **umask** command (a special command in the Korn shell) at the destination host. If the **rcp** command **-p** flag is set, the modification time and mode of source files are preserved at the destination host.

The user name entered for the remote host determines the file access privileges the **rcp** command uses at that host. Additionally, the user name given to a destination host determines the ownership and access modes of the resulting destination file or files.

Using Standard Authentication

The remote host allows access if one of the following conditions is satisfied:

- The local host is included in the remote host **/etc/hosts.equiv** file and the remote user is not the root user.
- The local host and user name is included in a **$HOME/.rhosts** file on the remote user account.

Although you can set any permissions for the **$HOME/.rhosts** file, it is recommended that the permissions of the .rhosts file be set to 600 (read and write by owner only).

In addition to the preceding conditions, the **rcp** command also allows access to the remote host if the remote user account does not have a password defined. However, for security reasons, the use of a password on all user accounts is recommended.

For Kerberos 5 Authentication

The remote host allows access only if all of the following conditions are satisfied:

- The local user has current DCE credentials.
- The local and remote systems are configured for Kerberos 5 authentication (On some remote systems, this may not be necessary. It is necessary that a daemon is listening to the klogin port).
- The remote system accepts the DCE credentials as sufficient for access to the remote account. See the **kvalid_user** function for additional information.

rcp and Named Pipelines

Do not use the **rcp** command to copy named pipelines, or FIFOs, (special files created with the **mknod -p** command). The **rcp** command uses the **open** subroutine on the files that it copies, and this subroutine blocks on blocking devices like a FIFO pipe.

Restrictions

The SP Kerberos V4 rcp execution path does not support remote-to-remote copy as Kerberos does not support forwarding credentials. The message you would receive under these circumstances is the message indicating you do not have tickets and must use **kinit** to login. The message would be issued from the remote source machine. Please see the example below for using Kerberos to perform a remote-to-remote copy.

Flags

Item	Description
-p	Preserves the modification times and modes of the source files in the copies sent to the destination only if the user has root authority or is the owner of the destination. Without this flag, the **umask** command at the destination modifies the mode of the destination file, and the modification time of the destination file is set to the time the file is received. When this flag is not used, the umask being honored is the value stored in the appropriate database. It is not the value that is set by issuing the **umask** command. The permission and ownership values that result from the **umask** command do not affect those stored in the database.
-r	Recursively copies, for directories only, each file and subdirectory in the source directory into the destination directory.

Item	Description
-F	Causes the credentials to be forwarded. In addition, the credentials on the remote system will be marked forwardable (allowing them to be passed to another remote system). This flag will be ignored if Kerberos 5 is not the current authentication method. Authentication will fail if the current DCE credentials are not marked forwardable.
-k *realm*	Allows the user to specify the realm of the remote station if it is different from the local systems realm. For these purposes, a realm is synonymous with a DCE cell. This flag will be ignored if Kerberos 5 is not the current authentication method.
-m	Support for metacharacters in filenames.

Parameters

Item	Description
*Host***:***File*	Specifies the host name (*Host*) and file name (*File*) of the remote destination file, separated by a : (colon). **Note:** Because the **rcp** command assumes that a : (colon) terminates a host name, you must insert a \ (backslash) before any colons that are embedded in the local file and directory names.
*User***@***Host***:***File*	Specifies the user name (*User***@**) that the **rcp** command uses to set ownership of the transferred file, the host name (*Host*), and file name (*File*) of the remote destination file. The user name entered for the remote host determines the file access privileges the **rcp** command uses at that host.
File	Specifies the file name of the local destination file.
*Host***:***Directory*	Specifies the host name (*Host*) and directory name (*Directory*) of the remote destination directory. **Note:** Because the **rcp** command assumes that a : (colon) terminates a host name, you must insert a \ (backslash) before any colons that are embedded in the local file and directory names.
*User***@***Host***:***Directory*	Specifies the user name (*User***@**) the **rcp** command uses to set ownership of the transferred file, the host name (*Host*), and directory name (*Directory*) of the remote destination directory. The user name entered for the remote host determines the file access privileges the **rcp** command uses at that host.
Directory	The directory name of the local destination directory.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

The remote host allows access only if at least one of the following conditions is satisfied:

- The local user ID is listed as a principal in the authentication database and had performed a **kinit** to obtain an authentication ticket.

- If a **$HOME/.klogin** file exists, it must be located in the local user's **$HOME** directory on the target system. The local user must be listed as well as any users or services allowed to **rsh** into this account. This file performs a similar function to a local .**rhosts** file. Each line in this file should contain a principal in the form of "principal.instance@realm." If the originating user is authenticated as one of the principals named in .**klogin**, access is granted to the account. The owner of the account is granted access if there is no .**klogin** file.

For security reasons, any **$HOME/.klogin** file must be owned by the remote user and only the AIX owner ID should have read and write access (permissions = 600) to .**klogin**.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

In the following examples, the local host is listed in the **/etc/hosts.equiv** file at the remote host.

1. To copy a local file to a remote host, enter:

   ```
   rcp localfile host2:/home/eng/jane
   ```

   The file `localfile` from the local host is copied to the remote host `host2`.
2. To copy a remote file from one remote host to another remote host, enter:

   ```
   rcp host1:/home/eng/jane/newplan host2:/home/eng/mary
   ```

   The file `/home/eng/jane/newplan` is copied from remote host `host1` to remote host `host2`.
3. To send the directory subtree from the local host to a remote host and preserve the modification times and modes, enter:

   ```
   rcp  -p  -r report jane@host2:report
   ```

   The directory subtree `report` is copied from the local host to the home directory of user `jane` at remote host `host2` and all modes and modification times are preserved. The remote file **/home/jane/.rhosts** includes an entry specifying the local host and user name.
4. This example shows how the root user can issue an **rcp** on a remote host when the authentication is Kerberos 4 on both the target and server. The root user must be in the authentication database and must have already issued **kinit** on the local host. The command is issued at the local host to copy the file, stuff, from node r05n07 to node r05n05 on an SP.

   ```
   /usr/lpp/ssp/rcmd/bin/rsh r05n07 'export KRBTKTFILE=/tmp/rcmdtkt$$; \
   /usr/lpp/ssp/rcmd/bin/rcmdtgt; \
   /usr/lpp/ssp/rcmd/bin/rcp /tmp/stuff r05n05:/tmp/stuff;'
   ```

   The root user sets the KRBTKTFILE environment variable to the name of a temporary ticket-cache file and then obtains a service ticket by issuing the **rcmdtgt** command. The **rcp** uses the service ticket to authenticate from host r05n07 to host r05n05.

Files

Item	Description
$HOME/.klogin	Specifies remote users that can use a local user account.
/usr/lpp/ssp/rcmd/bin/rcp	Link to AIX Secure **/usr/bin/rsh** that calls the SP Kerberos 4 **rcp** routine if applicable.

Prerequisite Information

Refer to the chapter on security in IBM *Parallel System Support Programs for AIX*: Administration Guide for an overview. You can access this publication at the following Web site: *http://www.rs6000.ibm.com/resource/aix_resource*

Refer to the "RS/6000 SP Files and Other Technical Information" section of IBM Parallel System Support Programs for AIX: Command and Technical Reference for additional Kerberos information. You can access this publication at the following Web site: *http://www.rs6000.ibm.com/resource/aix_resource*

rcvdist Command

Purpose

Sends a copy of incoming messages to additional recipients.

Syntax

rcvdist [**-form** *File*] *User ...*

Description

The **rcvdist** command forwards copies of incoming messages to users in addition to the original recipient. The **rcvdist** command is not started by a user. The **rcvdist** command is placed in the **.maildelivery** file called by the **/usr/lib/mh/slocal** command.

The **rcvdist** command sends a copy of an incoming message to the user or users specified by the *User* parameter. The default string is located in the **rcvdistcomps** file. This file formats the output from the command and sends it through the **send** command to the ID or alias specified.

You can copy the **rcvdistcomps** file into your local mail directory and change the string to suit your needs. The Message Handler (MH) package uses the **rcvdistcomps** file in your local mail directory first. Otherwise, you can use the **-form** flag to specify a file name that contains the string you want.

Flags

Item	Description
-form *File*	Specifies the file that formats the command output. The default is the **rcvdistcomps** file.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.

Files

Item	Description
$HOME/.maildelivery	Provides the user with MH instructions for local mail delivery.
$HOME/.forward	Provides the user with the default message filter.

rcvpack Command

Purpose

Saves incoming messages in a packed file.

Syntax

rcvpack [*File*]

Description

The **rcvpack** command places incoming messages in the packed file specified by the *File* parameter. The **rcvpack** command is not started by the user. The **rcvpack** command is placed in the **$HOME/ .maildelivery** file runs the **rcvpack** command on all incoming messages.

Flags

Item	Description
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.

Files

Item	Description
$HOME/html	

rcvstore Command

Purpose

Incorporates new mail from standard input into a folder.

Syntax

rcvstore [**+***Folder*] [**-create** | **-nocreate**] [**-sequence** *Name*] [**-public** | **-nopublic**] [**-zero** | **-nozero**]

Description

The **rcvstore** command adds incoming messages to a specified message directory (a folder). The **rcvstore** command is not started by the user. The **rcvstore** command is placed in the **$HOME/**.maildelivery file.

You can specify **rcvstore** command flags in the **$HOME/** .mh_profile file.

Flags

Item	Description
-create	Creates the specified folder in your mail directory if the folder does not exist. This flag is the default.
+*Folder*	Places the incorporated messages in the specified folder. The default is +inbox.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-nocreate	Does not create the specified folder if the folder does not exist.
-nopublic	Restricts the specified sequence of messages to your usage. The **-nopublic** flag does not restrict the messages in the sequence, only the sequence. This flag is the default if the folder is write-protected against other users.

Item	Description
-nozero	Appends the messages incorporated by the **rcvstore** command to the specified sequence of messages. This flag is the default.
-public	Makes the specified sequence of messages available to other users. The **-public** flag does not make protected messages available, only the sequence. This flag is the default if the folder is not write-protected against other users.
-sequence *Name*	Adds the incorporated messages to the sequence of messages specified by the *Name* parameter.
-zero	Clears the specified sequence of messages before placing the incorporated messages into the sequence. This flag is the default.

Profile Entries

Item	Description
`Folder-Protect:`	Sets the protection level for your new folder directories.
`Msg-Protect:`	Sets the protection level for your new message files.
`Path:`	Specifies the *UserMHDirectory* (the user's MH directory) variable.
`Unseen-Sequence:`	Specifies the sequences of commands used to keep track of your unseen messages.
`Rcvstore:`	Specifies flags for the **rcvstore** program.

Files

Item	Description
$HOME/.maildelivery	Provides the user with MH instructions for local mail delivery.
$HOME/.forward	Provides the user with the default message filter.

rcvtty Command

Purpose

Notifies the user of incoming messages.

Syntax

rcvtty [*Command*]

Description

The **rcvtty** command sends the user a message that incoming mail has arrived. The **rcvtty** command is not started by the user. The **rcvtty** command is placed in the **.maildelivery** file.

Flags

Item	Description
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.

Files

Item	Description
$HOME/$HOME/.mh_profile	Contains the MH user profile.

rdist Command

This document describes the old AIX **rdist** command located in the **/usr/bin/rdist** file as well as the new **/usr/sbin/rdist** command which is used with the new **rdistd** daemon.

/usr/bin/rdist Command

Purpose

Remote file distribution client program.

Syntax

To Use a Distribution File

rdist [**-n**] [**-q**] [**-b**] [**-D**] [**-R**] [**-h**] [**-i**] [**-v**] [**-w**] [**-y**] [**-f** *FileName*] [**-d** *Argument*=*Value*] [**-m** *Host*] ... [*Name*] ...

To Interpret Arguments as a Small Distribution File

rdist [**-n**] [**-q**] [**-b**] [**-D**] [**-R**] [**-h**] [**-i**] [**-v**] [**-w**] [**-y**] **-c** *Name ...* [*Login***@**] *Host* [**:***Destination*]

Description

Attention: Do not attempt to use the **rdist** command to send a file larger than 2 Gigabytes in size to a non-AIX machine. Doing so results in undefined behaviors and, in rare cases, the loss of data.

The **rdist** command maintains identical copies of files on multiple hosts. The **rdist** command preserves the owner, group, mode, and modified time of files, if possible, and can update programs that are running. The **rdist** command can receive direction from the following sources:

- The default distribution file, **distfile** file in your **$HOME** directory.
- A different distribution file, specified by the **-f** flag.
- Command-line arguments that augment or override variable definitions in the distribution file.
- Command-line arguments that serve as a small distribution file.

If you do not use the **-f** flag, the **rdist** command looks for the **distfile** file in your **$HOME** directory. If it doesn't find a **distfile** file, it looks for **Distfile** file.

The value specified by the *Name* parameter is read as the name of a file to be updated or a subcommand to execute. If you do not specify a value for the *Name* parameter on the command line, the **rdist** command updates all the files and directories listed in the distribution file. If you specify **-** (minus sign) for the *Name* parameter, the **rdist** command uses standard input. If the name of a file specified by the *Name* parameter is the same as the name of a subcommand, the **rdist** command interprets the *Name* parameter as a subcommand.

The **rdist** command requires that a **.rhosts** file be configured on each host. See File Format for TCP/IP in *Files Reference* for details.

Note:

1. If the **rdist** command is not present in the **/usr/bin/rdist** directory on a remote machine, create a link from the **/usr/bin/rdist** directory to the actual location of the **rdist** command. This location is usually the **/usr/ucb/rdist** directory.
2. Currently, the **rdist** command can handle only 7-bit ASCII file names.

Flags

Item	Description
-b	Performs a binary comparison and updates files if they differ.
-c	Directs the **rdist** command to interpret the remaining arguments as a small distribution file. Available arguments are: ***Name*** Specifies single name or list of names separated by blanks. The value can be either a file or a subcommand. **[*Login*@]*Host*** Specifies the machine to be updated and, optionally, the login name to be notified of the update. ***Destination*** Specifies a file on the remote machine if a single name is specified in the *Name* argument; specifies a directory if more than one name is specified. **Note:** Do not use the **-c** flag with the **-f**, **-d**, or **-m** flag.
-d *Argument*=*Value*	Defines the *Argument* variable as having the value specified by the *Value* variable. The **-d** flag defines or overrides variable definitions in the **distfile** file. The *Value* variable can be specified as an empty string, one name, or a list of names surrounded by parentheses and separated by tabs or spaces.
-D	Turns on the debugging output.
-f *FileName*	Specifies the name of the distribution file. If you do not use the **-f** flag, the default value is the **distfile** or **Distfile** file in your **$HOME** directory.
-h	Copies the file that the link points to rather than the link itself.
-i	Ignores unresolved links. The **rdist** command maintains the link structure of files being transferred and warns users if it cannot find all the links.
-m *Host*	Limits which machines are to be updated. You can use the **-m** *Host* option multiple times to limit updates to a subset of the hosts listed in the **distfile** file.
-n	Prints the subcommands without executing them. Use the **-n** flag to debug the **distfile** file.
-q	Operates in quiet mode. The **-q** option suppresses printing of modified files on standard output.
-R	Removes extraneous files. If a directory is being updated, any files that exist on the remote host but not in the master directory are removed. Use the **-R** flag to maintain identical copies of directories.
-v	Verifies that the files are up-to-date on all hosts; files that are out-of-date are then displayed. However, the **rdist -v** command neither changes files nor sends mail. This flag overrides the **-b** flag when they are used together.
-y	Prevents recent copies of files from being replaced by files that are not as recent. Files are normally updated when their time stamp and size differ. The **-y** flag prevents the **rdist** command from updating files more recent than the master file.
-w	Appends the entire path name of the file to the destination directory name. Normally, the **rdist** command uses only the last component of a name for renaming files, preserving the directory structure of the copied files. When the **-w** flag is used with a file name that begins with a ~ (tilde), everything except the home directory is appended to the destination name. File names that do not begin with a / (slash) or a ~ (tilde) use the destination user's home directory as the root directory for the rest of the file name.

Distribution File (distfile File)

The distribution file specifies the files to copy, destination hosts for distribution, and operations to perform when updating files to be distributed with the **rdist** command. Normally, the **rdist** command uses the **distfile** file in your **$HOME** directory. You can specify a different file If you use the **-f** flag.

Entry Formats

Each entry in the distribution file has one of the following formats:

Item	Description
VariableName **=** *NameList*	Defines variables used in other entries of the distribution file (*SourceList*, *DestinationList*, or *SubcommandList*).
[*Label***:**] *SourceList* **->** *DestinationList SubcommandList*	Directs the **rdist** command to distribute files named in the *SourceList* variable to hosts named in the *DestinationList* variable. Distribution file commands perform additional functions.
[*Label***:**] *SourceList* **::** *TimeStampFile SubcommandList*	Directs the **rdist** command to update files that have changed since a given date. Distribution file subcommands perform additional functions. Each file specified with the *SourceList* variable is updated if the file is newer than the time-stamp file. This format is useful for restoring files.

Labels are optional and used to identify a subcommand for partial updates.

Entries

Item	Description
VariableName	Identifies the variable used in the distribution file.
NameList	Specifies a list of files and directories, hosts, or subcommands.

Item	Description
SourceList	Specifies files and directories on the local host for the **rdist** command to use as the master copy for distribution.
DestinationList	Indicates hosts to receive copies of the files.
SubcommandList	Lists distribution file subcommands to be executed.

The **rdist** command treats new-line characters, tabs, and blanks as separators. Distribution file variables for expansion begin with a $ (dollar sign) followed by a single character or a name enclosed in {} (braces). Comments begin with a # (pound sign) and end with a new-line character.

Source and Destination List Format

The distribution file source and destination lists comprise zero or more names separated by blanks, as shown in the following format:

[*Name1*] [*Name2*] [*Name3*] ...

The **rdist** command recognizes and expands the following shell metacharacters on the local host in the same way as for the **csh** command.

- [(left bracket)
-] (right bracket)
- { (left brace)
- } (right brace)
- ((left parenthesis)
-) (right parenthesis)
- * (asterisk)
- ? (question mark)

To prevent these characters from being expanded, precede them with a \ (backslash). The **rdist** command also expands the ~ (tilde) in the same way as for the **csh** command, but does so separately on the local and destination hosts.

Distribution File Subcommands

Multiple commands to the shell must be separated by a ; (semicolon). Commands are executed in the user's home directory on the host being updated. The **special** subcommand can be used to rebuild private databases after a program has been updated.

The distribution file subcommand list may contain zero or more of the following subcommands:

Item	Description
install *Options* [*OptionalDestName*]**;**	Copies out-of-date files and directories. The **rdist** command copies each source file or directory to each host in the destination list. The available options as specified by the *Options* variable are the **rdist** command flags **-b**, **-h**, **-i**, **-R**, **-v**, **-w**, and **-y**. These options only apply to the files specified by the *SourceList* variable. When you use the **-R** flag, nonempty directories are removed if the corresponding file name is absent on the master host. The *OptionalDestName* parameter renames files. If no **install** subcommand appears in the subcommand list or the destination name is not specified, the source file name is used. Directories in the path name are created if they do not exist on the remote host. The login name used on the destination host is the same as the local host unless the destination name is of the format *login***@***host*.
notify *NameList***;**	Mails the list of updated files and any errors that may have occurred to the listed names (the *NameList* parameter). If no **@** (at sign) appears in the name, the destination host is appended to the name (*name***@***host*).
except *NameList***;**	Causes the **rdist** command to update all the files specified by the *SourceList* entry except for those files specified by the *NameList* variable.
except_pat *NameList***;**	Prevents the **rdist** command from updating any files that contain a string that matches a member of the list specified by the *NameList* variable.
special *NameList* **"***String***";**	Specifies shell commands (the **"***String***"** variable) to be executed on the remote host after the file specified by the *NameList* variable is updated or installed. If the *NameList* variable is omitted, the shell commands are executed for every file updated or installed. The shell variable **FILE** is set to the current file name before the **rdist** command executes the **"***String***"** variable. The **"***String***"** value must be enclosed in **" "** (double quotation marks) and can cross multiple lines in the distribution file.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	Specifies that an error occurred.

Examples

Examples of the Format: VariableName = NameList

1. To indicate which hosts' files to update, enter a line similar to the following:

```
HOSTS =( matisse root@arpa )
```

   where the HOSTS variable is defined to be `matisse` and `root@arpa`. The **rdist** command updates files on the hosts `matisse` and `root@arpa`. You could use this variable as a destination list.

2. To indicate a name to use as a value for a *SourceList* entry, enter a line similar to the following:

```
FILES = ( /bin /lib/usr/bin /usr/games
    /usr/include/{*.h,{stand,sys,vax*,pascal,machine}/*.h}
       /usr/lib /usr/man/man? /usr/ucb /usr/local/rdist )
```

   where the FILES value is defined to be the files to be used for the *SourceList* entry.

3. To indicate which files to exclude from the updating process, enter a line similar to the following:

```
EXLIB = ( Mail.rc aliases aliases.dir aliases.pag crontab dshrc
    sendmail.cf sendmail.fc sendmail.hf sendmail.st uucp vfont)
```

   where the EXLIB value is defined as a list of files to exclude from the updating process.

4. To copy all files from **/usr/src/bin** to **arpa** expanding the *namelist* variable so that all files except those present in the *namelist* variable and having `.o` as an extension are copied:

```
/usr/src/bin ->arpa
except_pat(\e\e.o\e ${<namelist> /SCCS\e ${<namelist>}
```

   or

```
/usr/src/bin ->arpa
except_pat(\\.o\e ${<namelist> /SCCS\e ${<namelist>}
```

5. To copy all files from **/usr/src/bin** to **arpa** except those with an `.o` extension:

```
/usr/src/bin ->arpa
except_pat(\\.o\$ /SCCS\$
```

Examples of the Format: [label:] SourceList - DestinationList SubcommandList

1. To copy a source list of files to a destination list of hosts, enter a line similar to the following:

```
${FILES} ->${HOSTS}
    install -R
    except /usr/lib/${EXLIB}  ;
    except /usr/games/lib  ;
    special /usr/sbin/sendmail "/usr/sbin/sendmail.bz"  ;
```

   The [*Label***:**] entry of the line is optional and not shown here. The $ (dollar sign) and the {} (braces) cause the file names FILES, HOSTS, and EXLIB to be expanded into the lists designated for them in the previous examples. The rest of the example comprises the subcommand list.

2. To use the [*Label***:**] entry, enter the line as follows:

```
srcsL:
/usr/src/bin -> arpa
    except_pat (\e\e.o\e$ /SCCS\e$ ) ;
```

The label is `srcsL:` and can be used to identify this entry for updating. The `/usr/src/bin` file is the source to be copied and host `arpa` is the destination of the copy. The third line contains a subcommand from the subcommand list.

3. To use a time-stamp file, enter a line similar to the following:

```
${FILES} :: stamp.cory
    notify root@cory
```

The $ (dollar sign) and {} (braces) cause the name specified by `FILES` to be expanded into the list designated for it. The time-stamp file is `stamp.cory`. The last line is a subcommand from the subcommand list.

Files

Item	Description
/usr/bin/rdist	Contains the **rdist** command.
$HOME/distfile	Contains a list of subcommands to be read by the **rdist** command.
/tmp/rdist	Contains an update list. This is a temporary file.

/usr/sbin/rdist Command

This document describes the old AIX **rdist** command located in the **/usr/bin/rdist** file as well as the new **/usr/sbin/rdist** command which is used with the new **rdistd** daemon.

Purpose (/usr/sbin/rdist)

Client program for distributing files remotely.

Syntax (/usr/sbin/rdist)

To Use a Distribution File

/usr/sbin/rdist [**-F n**] [**-A** *num*] [**-a** *num*] [**-d** *var=value*] [**-l** < *local logopts*>] [**-L** <*remote logopts*>] [**-f** *distfile*] [**-M** *maxproc* **-m** *host*] [**-o** *distops*] [**-t** *timeout*] [**-p** <*rdist-path*>] [**-P** <*transport-path*>] [*name* ...]

To Interpret Arguments as a Small Distribution File

/usr/sbin/rdist -Fn -c *name* ... [*login@*] *host* [*:dest*]

To Invoke the Old rdist as a Server

/usr/sbin/rdist -Server

For Version Information

/usr/sbin/rdist -V

Description (/usr/sbin/rdist)

rdist is a program to maintain identical copies of files over multiple hosts. It preserves the owner, group, mode, and modification time of files if possible and can update programs that are running. The **rdist** command can receive direction from the following sources:

- The distribution file **distfile** in the current directory.
- The standard input if **distfile** is specified as -.
- If the **-f** flag is not used, **rdist** looks for the file named *distfile* and *Distfile*.

- If the **-c** flag is used, the trailing arguments are interpreted as a small **distfile**. The equivalent **distfile** is as follows.

```
( filename ... ) -> [user@]host
     install       [dest name] ;
```

If no **name** arguments are specified, **rdist** will update all of the files and directories listed in **distfile**. Otherwise, the argument is taken to be the name of a file to be updated or the label of a command to execute. If the label and file names conflict, it is assumed to be a label. These may be used together to update specific files using specific commands.

The **-Server** option provides backward compatibility for older versions of **rdist** which used this option to put **rdist** into server mode. If **rdist** is started with the **-Server** command line option, it will attempt to run the old version of **rdist**. This option will only work if the old **rdist** is located at **/usr/bin/rdist**.

rdist uses an arbitrary transport program to access each target host. The transport program can be specified on the command line with the **-P** flag. If the **-P** flag is not used, **rsh** is taken as the transport program. If the **rsh** method is used and the target host is the string **localhost** and the remote user name is the same as the local user name, **rdist** will attempt to run the following command:

```
/bin/sh -c rdistd -S
```

Otherwise **rdist** will run the following command:

```
rsh host -l remuser rdistd -S
```

In the example above, the *host* parameter is the name of the target host, *remuser* is the name of the user to make the connection as and, **rdistd** is the **rdist** server command on the target host.

The transport program must be compatible with the above syntax for **rsh**. If not, the transport program should be wrapped in a shell script which understands this command line syntax.

On each target host **rdist** will run the following command:

```
rdistd -S
```

or

```
<rdistd path> -S
```

In the example above, the **-p** flag was specified. If **-p** flag is not included, or the <rdistd path> is a simple filename, **rdistd** or <rdistd path> must be somewhere in the **PATH** of the user running **rdist** on the remote (target) host.

The **rdist** command uses the following environment variables:

Item	Description
TMPDIR	Name of temporary directory to use. Default is **/tmp**.

Flags (/usr/sbin/rdist)

Item	Description
-A *num*	Update or install files only if a minimum number of free files (inodes) exists on a filesystem.
-a *num*	Update or install files only if a minimum amount of free space exists on a filesystem.
-d *var = value*	Assign *value* to variable *var*. This option is used to define or override variable definitions in the **distfile**. *Value* can be the empty string, one name, or a list of names surrounded by parentheses and separated by tabs and/or spaces.
-F	Update all clients sequentially without forking child processes.
-f *distfile*	Use **distfile** as the distribution file. If **distfile** is specified as **-**, read from standard input.
-l *logopts*	Sets local logging options. See the **Message Logging** section for more information on the syntax for *logopts*.
-L *logopts*	Sets remote logging options. *logopts* is the same as for local logging except the values are passed to the remote server (**rdistd**). See the **Message Logging** section for more information on the syntax of *logopts*.
-M *num*	Limit the maximum number of simultaneously running child **rdist** processes to *num*. The default is 4.

Item	Description
-m *machine*	Limits the updating of files to the given machine. Multiple **-m** arguments can be given to limit updates to a subset of the hosts listed in the **distfile**.
-n	Display but do not execute commands. Use the **-n** flag to debug **distfile**.
-o *distopts*	Specifies the dist options to enable. *distopts* is a comma separated list of options listed below. The valid values for *distopts* are: **chknfs** If the target filesystem is NFS, do not check or update files. **chkreadonly** If a file on the target host resides on a read only filesystem, no checking or updating of the file is attempted. **chksym** If the target on the remote host is a symbolic link, but is not on the master host, the remote target will be left a symbolic link. **compare** Perform a binary comparison and update files if they differ. **follow** Copy the file that the symbolic link points to rather than the link itself. **ignlnks** Ignore links which do not resolve. The normal behavior of **rdist** is to warn the user about unresolved links. **nochkowner** If the file already exists, do not check user ownership. The file ownership is only set when the file is updated. **nochkgroup** If the file already exists, do not check group ownership. The file ownership is only set when the file is updated. **nochkmode** Avoid checking file and directory permission modes. The permission mode is only set when the file is updated. **nodescend** Do not descend recursively into a directory. Only the existence, ownership, and mode of the directory are checked. **noexec** Do not check or update executable files that are in **a.out** format. **numchkgroup** Use the numeric group id (gid) to check group ownership instead of the group name. **numchkowner** Use the numeric user id (uid) to check user ownership instead of the user name. **quiet** Supress printing files that are being modified on the standard output. **remove** Remove any files in directories that exist on the remote host that do not exist in the master directory on the local host. **savetargets** Save files that are updated instead of removing them. Target files that are updated are first renamed from **filename** to **filename.OLD**. **sparse** Enable checking for sparse files. This option adds some additional processing overhead so it should only be enabled for targets likely to contain sparse files.
-o *distopts*	*(dist options, continued)*: **verify** Any file on any host that is out of date will be displayed but no file will be changed nor any mail sent. **whole** The whole file name is appended to the destination directory name. Normally, only the last component of a name is used when renaming files. This will preserve the directory structure of the files being copied instead of flattening the directory structure. For example, rdisting a list of files such as **/path/dir1/f1** and **/path/dir2/f2** to **/tmp/dir** would create files **/tmp/dir/path/dir1/f1** and **/tmp/dir/path/dir2/f2** instead of **/tmp/dir/dir1/f1** and **/tmp/dir/dir2/f2**. **younger** Files are normally updated if their *mtime* and *size* disagree. This option causes **rdist** not to update files that are younger than the master copy. This can be used to prevent newer copies on other hosts from being replaced. A warning message is printed for files which are newer than the master copy.
-p <*rdist-path*>	Search for the **rdistd** server in the given path on the target host.
-P <*rdist-path*>	Use the transport program as given in *transport-path*. The *transport-path* may be a colon seperated list of possible pathnames. In this case, the first component of the path to exist is used.
-t *timeout*	Sets the *timeout* period (in seconds) for waiting for responses from the remote **rdist** server. The default is 900 seconds.
-V	Prints the version information and exits.

Message Logging

The **rdist** command provides a set of message facilities, each of which contains a list of message types specifying which types of messages to send to that facility. The local client (**rdist**) and the remote server (**rdistd**) each maintain separate copies of what types of messages to log to what facilities.

The **-l** *logopts* flag specifies what logging options to use locally on the client. The **-L** *logopts* flag specifies what logging options to pass to the remote **rdistd** server.

The form of *logopts* should be the following:

```
facility=types:facility= types...
```

The valid facility names are as follows:

stdout
Messages to standard output.

file
Messages are sent to a file. The file name can be specified by the format **file** = *filename* = *types*.

syslog
Messages are sent to the **syslogd** facility.

notify
Messages are sent to the internal **rdistnotify** facility. This facility is used in conjunction with the **notify** ph in a **distfile** to specify what messages are mailed to the **notify** address.

types should be a comma separated list of message types. Each message type specified enables that message level. This is unlike the **syslog** system facility which uses an ascending order scheme. The following are the valid types:

change
Log messages for things that change.

info
Log general information.

notice
Log messages for general info about things that change. This includes things like making directories which are needed in order to install a specific target, but which are not explicitly specified in the **distfile**.

nerror
Log messages for normal errors that are not fatal.

ferror
Log messages for fatal errors.

warning
Log warnings about errors which are not as serious as **nerror** type messages.

verbose
Log messages for more information than normal, but less than debugging level.

debug
Log debugging information.

all
Log all but debug messages.

Distribution File (/usr/sbin/rdist)

The distribution file specifies the files to copy, destination hosts for distribution, and operations to perform when updating files to be distributed with the **rdist** command.

Entry Formats

Each entry in the distribution file has one of the following formats:

VariableName = NameList
Defines variables used in other entries of the distribution file (*SourceList*, *DestinationList*, or *SubcommandList*).

[Label:] SourceList -> DestinationList SubcommandList
Directs the **rdist** command to distribute files named in the *SourceList* variable to hosts named in the *DestinationList* variable.

Distribution file commands perform additional functions.

[Label:] SourceList :: TimeStampFile SubcommandList

Directs the **rdist** command to update files that have changed since a given date. Distribution file subcommands perform additional functions.

Each file specified with the *SourceList* variable is updated if the file is newer than the time-stamp file.

Labels are optional. They are used to identify a command for partial updates.

Entries

Item	Description
VariableName	Identifies the variable used in the distribution file.
NameList	Specifies a list of files and directories, hosts, or subcommands.
SourceList	Specifies files and directories on the local host for the **rdist** command to use as the master copy for distribution.
DestinationList	Indicates hosts to receive copies of the files.
SubcommandList	Lists distribution file subcommands to be executed.

The **rdist** command treats newline characters, tabs, and blanks as separators. Distribution file variables for expansion begin with a dollar sign followed by a single character or a name enclosed in braces.

Comments begin with a pound sign and end with a newline character.

Source and Destination List Format

The distribution file source and destination lists comprise zero or more names separated by blanks, as shown in the following format:

```
[Name1] [Name2] [Name3] ...
```

The **rdist** command recognizes and expands the following shell metacharacters on the local host in the same way as for the **csh** command.

- [left bracket
-] right bracket
- { left brace
- } right brace
- (left parenthesis
-) right parenthesis
- * asterisk
- ? question mark

To prevent these characters from being expanded, precede them with a backslash. The **rdist** command also expands the tilde in the same way as for the **csh** command, but does so separately on the local and destination hosts. When the **-o** *whole* option is used with a file name that begins with a tilde, everything except the home directory is appended to the destination name. File names which do not begin with a forward slash or a tilde use the destination user's home directory as the root directory for the rest of the file name.

Distribution File Subcommands

Multiple commands to the shell must be separated by a semicolon. Commands are executed in the user's home directory on the host being updated. The special subcommand can be used to rebuild private databases after a program has been updated.

The distribution file subcommand list may contain zero or more of the following subcommands:

install Options[OptionalDestName];

Copies out-of-date files and directories. The **rdist** command copies each source file or directory to each host in the destination list.

The available options as specified by the *Options* variable are the **rdist** command flags **-b**, **-h**, **-i**, **-R**, **-v**, **-w**, and **-y**.

These options only apply to the files specified by the *SourceList* variable.

When you use the **-R** flag, nonempty directories are removed if the corresponding file name is absent on the master host. The *OptionalDestName* parameter renames files.

If no install subcommand appears in the subcommand list or the destination name is not specified, the source file name is used. Directories in the path name are created if they do not exist on the remote host.

The login name used on the destination host is the same as the local host unless the destination name is of the format login@host.

notify NameList;
Mails the list of updated files and any errors that may have occurred to the listed names (the *NameList* parameter).

If no @ (at sign) appears in the name, the destination host is appended to the name (name@host).

except NameList;
Causes the **rdist** command to update all the files specified by the *SourceList* entry except for those files specified by the *NameList* variable.

except_pat NameList;
Prevents the **rdist** command from updating any files that contain a string that matches a member of the list specified by the *NameList* variable.

special NameList "String";
Specifies shell commands (the "String" variable) to be executed on the remote host after the file specified by the *NameList* variable is updated or installed.

If the *NameList* variable is omitted, the shell commands are executed for every file updated or installed.

The shell variable FILE is set to the current file name before the **rdist** command executes the "String" variable.

The variable REMFILE will contain the full pathname of the remote file that was just updated and the variable BASEFILE will contain the basename of the remote file that was just updated.

The "String" value must be enclosed in double quotation marks and can cross multiple lines in the distribution file.

cmdspecial NameList "String";
The **cmdspecial** command is similar to the **special** command, except it is executed only when the entire command is completed instead of after each file is updated.

The shell variable FILES will contain the list of files. Each file name in the FILES shell variable is separated by a colon.

NFS checks are disabled if a hostname ends in a plus sign. This is equivalent to disabling the **-o** *chknfs* option just for this one host.

Exit Status (/usr/sbin/rdist)

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples (/usr/sbin/rdist)

1. To indicate which hosts' files to update, enter a line similar to the following:

```
HOSTS =( matisse root@arpa )
```

In the above example, the HOSTS variable is defined to be matisse and root@arpa. The **rdist** command updates files on the hosts matisse and root@arpa.

You could use this variable as a destination list.

2. To indicate a name to use as a value for a SourceList entry, enter a line similar to the following:

```
FILES = ( /bin /lib/usr/bin /usr/games
    /usr/include/{*.h,{stand,sys,vax*,pascal,machine}/*.h}
       /usr/lib /usr/man/man? /usr/ucb /usr/local/rdist )
```

In the above example, the FILES value is defined to be the files to be used for the *SourceList* entry.

3. To indicate which files to exclude from the updating process, enter a line similar to the following:

```
EXLIB = ( Mail.rc aliases aliases.dir aliases.pag crontab dshrc
    sendmail.cf sendmail.fc sendmail.hf sendmail.st uucp vfont)
```

In the above example, the EXLIB value is defined as a list of files to exclude from the updating process.

4. To copy all files from /usr/src/bin to arpa expanding the namelist variable so that all files except those present in the namelist variable and having .o as an extension are copied:

```
/usr/src/bin ->arpa
except_pat(\e\e.o\e ${<namelist> /SCCS\e ${<namelist>}
```

or

```
/usr/src/bin ->arpa
except_pat(\\.o\e ${<namelist> /SCCS\e ${<namelist>}
```

5. To copy all files from /usr/src/bin to arpa except those with an .o extension:

```
/usr/src/bin ->arpa
except_pat(\\.o\$ /SCCS\$
```

Examples of the Format: [label:] SourceList - DestinationList SubcommandList

1. To copy a source list of files to a destination list of hosts, enter a line similar to the following:

```
${FILES} ->${HOSTS}
    install -R
    except /usr/lib/${EXLIB}  ;
    except /usr/games/lib  ;
    special /usr/sbin/sendmail "/usr/sbin/sendmail.bz"  ;
```

The [Label:] entry of the line is optional and not shown here. Thedollar sign and the braces cause the file names FILES, HOSTS, and EXLIB to be expanded into the lists designated for them in the previous examples.

The rest of the example comprises the subcommand list.

2. To use the [Label:] entry, enter the line as follows:

```
srcsL:
/usr/src/bin -> arpa
    except_pat (\e\e.o\e$ /SCCS\e$ ) ;
```

The label is srcsL: and can be used to identify this entry for updating. The **/usr/src/bin** file is the source to be copied and host arpa is the destination of the copy.

The third line contains a subcommand from the subcommand list.

3. To use a time-stamp file, enter a line similar to the following:

```
${FILES} :: stamp.cory
    notify root@cory
```

The dollar sign and braces cause the name specified by FILES to be expanded into the list designated for it. The time-stamp file is **stamp.cory**.

The last line is a subcommand from the subcommand list.

Files (/usr/sbin/rdist)

Item	Description
/usr/sbin/rdist	Contains the **rdist** command at version 6.1.5.
distfile	Contains the input commands.
$ TMPDIR/rdist*	The temporary file for update lists.

rdistd Command

Purpose

Server program for distributing files remotely.

Syntax

```
rdistd -S
rdistd -V
```

Description

rdistd is the server program for the **rdist** command. It is normally run by **rdist** through **rsh**.

The **-S** flag ensures that **rdistd** is not accidentally started since it normally resides in a normal user's PATH environment variable.

Flags

Item	Description
-V	Print version information and exit.

Exit Status

This command returns the following exit values:

0
Successful completion.

>0
An error occurred.

Files

Item	Description
/usr/sbin/rdistd	Containsthe **rdistd** server
/usr/bin/rdistd	Symbolic link to **/usr/sbin/rdistd**

rdump Command

Purpose

Backs up files onto a remote machine's device.

Note: User must have root authority to run this command.

Syntax

rdump [**-b** *Blocks*] [**-B**] [**-c**] [**-d** *Density*] [**-L** *Length*] [**-s** *Size*] [**-u**] [**-w**] [**-W**] [**-***Level*] **-f** *Machine***:***Device* [*FileSystem* | *DeviceName*]

Description

The **rdump** command copies file systems by i-node from your local machine to a remote machine. The files are copied, using the **backup** command format, to a device on the remote machine. The device is accessed by using a remote server on the remote machine. You must have root authority to execute the **rdump** command. You must also define a local machine running the **rdump** command in the **/.rhosts** file of the target remote machine.

To back up a file system, specify the **-***Level* and *FileSystem* parameters to indicate the files you want to back up. You can use the **-***Level* parameter to back up either all files on the system (a full backup) or only the files that have been modified since a specific full backup (an incremental backup). The possible levels are 0 to 9. If you do not supply a level, the default level is 9. A level 0 backup includes all files on the file system. A level *n* backup includes all files modified since the last level *n* - 1 (*n* minus 1) backup. The levels, in conjunction with the **-u** flag, provide a method of maintaining a hierarchy of incremental backups for each file system.

Note:

1. Use the **-u** flag when you perform an incremental backup (the **-***Level* parameter) to ensure that information regarding the last date, time, and level of each incremental backup is written to the **/etc/dumpdates** file.
2. If the **rmt** command on the remote machine is not in **/usr/sbin/rmt**, then a link will need to be created on the remote machine from **/usr/sbin/rmt** to its actual location (usually **/etc/rmt**).

Flags

Item	Description
-b *Blocks*	Specifies the number of blocks to write in a single output operation. If you do not specify the *Blocks* variable, the **rdump** command uses a default value appropriate for the physical device selected. Larger values of the *Blocks* variable result in larger physical transfers to tape devices.
-B	Terminates the command without querying the user when an error occurs. If you specify the **-B** flag, the **rdump** command returns a nonzero value.
-c	Specifies that the tape is a cartridge format, not a 9-track format.
-d *Density*	Specifies the density of the tape in bits-per-inch (bpi). This value is used in calculating the amount of tape used per volume. If you do not specify a value for the *Density* variable, the default density is 1600 bpi. When using the **-c** flag without specifying a tape density, the default density is 8000 bpi.
-f *Machine***:***Device*	Specifies the *Machine* variable as the hostname of the remote machine. To send output to the named device, specify the *Device* variable as a file name (such as the **/dev/rmt0** file). The *Device* variable should specify only tape devices.

Item	Description
-L *Length*	Specifies the length of the tape in bytes. This flag overrides the **-c**, **-d**, and **-s** flags. You can specify the size with a suffix of b, k, m, or g to represent Blocks (512 bytes), Kilo (1024 bytes), Mega (1024 Kilobytes), or Giga (1024 Megabytes), respectively. To represent a tape length of 2 Gigabytes, type the following: `-L 2g`.
-s *Size*	Specifies the size of the tape in feet using the *Size* variable. If you do not specify a tape size, the default size is 2300 feet. When using the **-c** flag without specifying a tape size, the default size is 1700 feet. When the tape drive reaches the specified size, the **rdump** command waits for the tape to be changed.
-u	Updates the time, date, and level of the remote backup in the **/etc/dumpdates** file. This file provides the information needed for maintaining incremental backups.
-w	Currently disabled.
-W	Displays the file systems found in the **/etc/dumpdates** files.
-*Level*	Specifies the remote backup level (0 to 9). The default value of the *Level* variable is 9.
-?	Displays the usage message.

Parameters

Item	Description
DeviceName	Specifies the physical device name (the block or raw name).
FileSystem	Specifies the name of the directory on which the file system is usually mounted. The **rdump** command reads the **/etc/filesystems** file for the physical device name. If you do not specify a file system, the default is the root (**/**) file system.

Exit Status

This command returns the following exit values:

Item	Description
0	Indicates that the command completed successfully.
>0	Indicates that an error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To back up files in the **/usr** directory on your local machine to a remote machine, type:

```
rdump -u -0 -fcanine:/dev/rmt0 /usr
```

The **-u** flag tells the system to update the current backup level records in the **/etc/dumpdates** file. The **-***Level* flag set to backup level 0 specifies that all the files in the **/usr** directory are to be backed up. The ID of the remote machine is `canine` and the device is the **/dev/rmt0** device.

2. To back up files in the **/usr** directory on your local machine to a remote machine using an 8mm, 2.3GB tape, type:

```
rdump -fcanine:/dev/rmt0 -L 2200m /usr
```

Note: 2.2GB is used here instead of 2.3GB to avoid hitting the actual end of the tape.

3. To back up files in the **/usr** directory on your local machine to a remote machine using 0.25-inch tape, type:

```
rdump -fcanine:/dev/rmt0 -c /usr
```

When using the **-c** flag, the **rdump** command defaults to the correct size and density values for 0.25-inch tape.

Files

Item	Description
/etc/dumpdates	Contains logs of the most recent remote dump dates.
/etc/filesystems	Contains information on file systems.
/dev/rhd4	Contains the device where the default file system (root) is located.
/usr/sbin/rdump	Contains the **rdump** command.

read Command

Purpose

Reads one line from standard input.

Syntax

read [**-p**][**-r**][**-s**][**-u**[*n*]] [*VariableName?Prompt*]

[*VariableName ...*]

Description

The **read** command reads one line from standard input and assigns the values of each field in the input line to a shell variable using the characters in the IFS (Internal Field Separator) variable as separators. The *VariableName* parameter specifies the name of a shell variable that takes the value of one field from the line of input. The first shell variable specified by the *VariableName* parameter is assigned the value of the first field, the second shell variable specified by the *VariableName* parameter is assigned the value of the second field, and so on, until the last field is reached. If the line of standard input has more fields than there are corresponding shell variables specified by the *VariableName* parameter, the last shell variable specified is given the value of all the remaining fields. If there are fewer fields than shell variables, the remaining shell variables are set to empty strings.

> **Note:** If you omit the *VariableName* parameter, the variable REPLY is used as the default variable name.

The setting of shell variables by the **read** command affects the current shell execution environment.

Flags

Item	Description
-p	Reads input from the output of a process run by the Korn Shell using |& (pipe, ampersand). **Note:** An end-of-file character with the **-p** flag causes cleanup for this process so that another can be spawned.
-r	Specifies that the read command treat a \ (backslash) character as part of the input line, not as a control character.
-s	Saves the input as a command in the Korn Shell history file.
-u [*n*]	Reads input from the one-digit file descriptor number, *n*. The file descriptor can be opened with the ksh exec built-in command. The default value of the *n* is 0, which refers to the keyboard. A value of 2 refers to standard error.

Parameters

Item	Description
VariableName?Prompt	specifies the name of one variable, and a prompt to be used. When the Korn Shell is interactive, it will write the prompt to standard error, and then perform the input. If *Prompt* contains more than one word, you must enclose it in single or double quotes.
VariableName...	specfies one or more variable names separated by white space.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	Detected end-of-file character or an error occurred.

Examples

1. The following script prints a file with the first field of each line moved to the end of the line:

```
while read -r xx yy
do
         print printf "%s %s/n" $yy $xx
done < InputFile
```

2. To read a line and split it into fields, and use "Please enter: " as a prompt, type:

```
read word1?"Please enter:  " word2
```

   The system displays:

```
Please enter:
You enter:
hello world
```

   The value of the *word1* variable should have "hello" and *word2* should have "world."

3. To create a co-process, then use print **-p** to write to the co-process, and use read **-p** to read the input from the co-process, type:

```
(read; print "hello $REPLY")
print -p "world"
read -p line
```

The value of the *line* variable should have "hello world."

4. To save a copy of the input line as a command in the history file, type:

```
read -s line < input_file
```

If input_file contains "echo hello world," then "echo hello world" will be saved as a command in the history file.

readlvcopy Command

Purpose

Reads a specific mirror copy of a logical volume.

Syntax

readlvcopy **-d** *device* [**-c** *copy* | **-C** *copy* | **-b**] [**-n** *number_of_blocks*] [**-o** *outfile*] [**-s** *skip*] [**-S** *seek*]

Description

Flags

Item	Description
-d *device*	logical volume special device file to be read from
-c *copy*	Requested mirror copy to read from. Valid values are 1, 2, or 3 for the first, second, or third copy of the data. Data is read even if the logical partition has been marked stale. The default is the first copy of the data.
-C *copy*	Requested mirror copy to read from. Valid values are 1, 2, or 3 for the first, second, or third copy of the data. Stale logical partitions are not read.
-b	Read mirror copy marked as online backup.
-n *number_of_blocks*	Number of 128K blocks to read
-o *outfile*	Destination file. The default is *stdout*
-s *skip*	Number of 128K blocks to skip into *device*.
-S *seek*	Number of 128K blocks to seek into *outfile*

reboot or fastboot Command

Purpose

Restarts the system.

Syntax

{ **reboot** | **fastboot** } [**-l**] [**-n**] [**-q**] [**-t** *mmddHHMM* [*yy*]]

Description

The **reboot** command can be used to perform a reboot operation if no other users are logged into the system. The **lsattr** command and enter `lsattr -D -l sys0`. The default value is **true**. To reset the autorestart attribute value to **false**, use the **/var/adm/wtmp**, the login accounting file. These actions are inhibited if the **-l**, **-n**, or **-q** flags are present.

The **fastboot** command restarts the system by calling the **reboot** command. The **fsck** command runs during system startup to check file systems. This command provides BSD compatibility.

Flags

Item	Description
-l	Does not log the reboot or place a shutdown record in the accounting file. The **-l** flag does not suppress accounting file update. The **-n** and **-q** flags imply **-l**.
-n	Does not perform the **sync** command. Use of this flag can cause file system damage.
-q	Restarts without first shutting down running processes. **Note:** A file system synchronization will not occur if the **-q** flag is used. If you want the file system to be synchronized, manually run the **sync** command or use the **shutdown -r** command.
-t	Shuts down the system immediately and then restarts the system on the specified date. A valid date has the following format: *mmddHHMM* [*yy*] where: ***mm*** Specifies the month. ***dd*** Specifies the day. ***HH*** Specifies the hour. ***MM*** Specifies the minute. ***yy*** Specifies the year (optional). The two digit value represents the value of the year in the current century (based on the system time). For example, if the current year based on the systems time is 1985, 99 means 1999 and if the current year is 2005 then 99 means 2099 and 04 means 2004.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To shut down the system without logging the reboot, enter:

```
reboot  -l
```

Files

Item	Description
/etc/rc	Specifies the system startup script.
/var/adm/wtmp	Specifies login accounting file.

rebootwpar Command

Purpose

Stops and restarts a system workload partition.

Restriction: You cannot run the **rebootwpar** command on an application workload partition.

Syntax

rebootwpar [**-F** | **-h**] [**-N** | **-t** *seconds*] [**-v**] *WparName*

Description

The **rebootwpar** command stops and restarts the workload partition.

Flags

Item	Description
-F	Specifies a forced stop.
-h	Specifies a hard stop.
-N	Specifies there is no timeout for halt.
-t *seconds*	Specifies the halt timeout in seconds.
-v	Verbose mode.
WparName	Specifies the workload partition name.

Security

Access Control: Only the root user can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To reboot the workload partition called "MyWpar", enter:

```
rebootwpar MyWpar
```

recfgct Command

Purpose

Reconfigures the Reliable Scalable Cluster Technology (RSCT) subsystems.

Syntax

```
/opt/rsct/install/bin/recfgct [ -i Node_ID | -n | -s | -h ]
```

Description

Attention: Use this command with extreme caution.

The `recfgct` command is used to remove all RSCT data under the `/var/ct` directory, generate a new node ID, and make it appear as if the RSCT components are just installed. Because of the destructive nature of this command, it is not normally started by the system administrator. You must use this command *only* if you need to remove a duplicate node ID or if an IBM service representative instructs you to use it.

When RSCT is first installed, a node ID is automatically generated. The node ID is a true random 64-bit number. Each system where RSCT is installed must have a unique node ID. If a copy of an operating system image (OSI) that has RSCT installed on it is installed on another system, the other system has the same node ID as the system from which the copy is made. This is referred to as *cloning*. For AIX platform, cloning is typically performed using such AIX-supported commands and procedures as `mksysb`. These commands and procedures call `recfgct` automatically. For other platforms, the `recfgct` command must be run immediately after a cloned OSI is installed.

If the `-s` flag is specified, after all data under the `/var/ct` directory is removed, the node ID contained in the `/etc/ct_node_id` file is used to re-create the `/var/ct/cfg/ct_node_id` file.

Flags

-i *Node_ID*
Specifies the node ID that must be used. The node ID must contain 9 - 16 hexadecimal characters.

-n
Generates a new node ID. It is the default behavior if no option is specified.

-s
Saves the node ID.

-h
Writes the command usage statement to standard output and then exits.

Restrictions

The `-h` flag is supported on the following RSCT levels:

- RSCT 2.4.9.1 (or later) for AIX 5.3
- RSCT 2.5.1.1 (or later) for AIX 6.1 and all Linux platforms
- RSCT 3.1.0.0 (or later) for AIX 7.1 and later

If you try to run the `recfgct -h` command on a prior version of RSCT, the `-h` flag is ignored and all RSCT data is removed.

Files

/etc/ct_node_id
Contains a copy of the RSCT node ID

/var/ct/cfg/ct_node_id
Contains the RSCT node ID

Standard output

When the `-h` flag is specified, this command usage statement is written to standard output and then the command exits.

Exit status

0
The command ran successfully.

1
The command did not run successfully.

Security

Privilege control: only the `root` user must have execute (x) access to this command.

Implementation specifics

This command is part of the `rsct.core` fileset for the AIX operating system and `rsct.core-3.1.0.0-0.`*platform*`.rpm` package for Linux, Solaris, and Windows operating system, where *platform* is `i386`, `ppc`, `ppc64`, `s390`, or `x86_64`.

Location

`/opt/rsct/install/bin/recfgct`

Examples

1. After installing a cloned operating system image, enter:

```
/opt/rsct/install/bin/recfgct
```

recreatevg Command

Purpose

Re-creates a volume group that exists on a specified set of disks. Imports and varies on the volume group.

Syntax

recreatevg [-y *VGname*] [-p] [-f] [-Y*Lv_Prefix* | -l *LvNameFile*] [-L *Label_Prefix*] [-n] [-V *MajorNumber*] [-d] [-O] *PVname*...

Description

The **recreatevg** command re-creates a volume group on a set of disks that are duplicated from another set of disks that belong to a specific volume group. This command overcomes the problem of duplicated Logical Volume Manager (LVM) data structures and identifiers that are caused by a disk duplication process. This command allocates new physical volume identifiers (PVID) for the member disks, as the PVIDs are also duplicated by the disk duplication. Similarly, duplicated logical volume members are given new names with the user-specified prefixes.

1. The **recreatevg** command removes all logical volumes that fully or partially exist on the physical volumes that are not specified on the command line. Mirrored logical volumes can be an exception (see the -f flag).
2. The **recreatevg** command warns, if the log for the logical volume of a file system does not exist on the disks that are specified on the command line.
3. The **recreatevg** command fails, if the input list does not match the list that is compiled from the Volume Group Descriptor Area (VGDA).
4. The set of disks in the list must have consistent VGDA data. The **recreatevg** command does not fix VGDA problems.

5. When re-creating a concurrent-capable volume group, the volume group is not varied on when the **recreatevg** command completes. The new volume group must be varied on manually.

Flags

Item	Description
-d	Instead of completely re-creating the VG, the d flag causes the **recreatevg** command to create only new PVIDs for the specified disks and update the LVM metadata with the new PVIDs. Logical volumes (LVs) names and labels is not changed and the VG is not imported. This flag is incompatible with other flags except the -0 flag.
-f	Re-creates a volume group (VG) from a subset of disks. Only those disks and the logical volumes (LVs) that is present entirely on this subset of disks is present in the re-created VG. All other disks and LVs from the original VG is deleted in the re-created VG. For mirrored LVs, only LV mirror copies with physical partitions allocated on the deleted disks are removed. Therefore, a mirrored LV can be re-created with fewer mirror copies when one of copies is present on the subset of disks.
-l*LvNameFile*	Changes logical volume names to the name specified by *LvNameFile*. Entries must be in the format LV:NEWLV1. All logical volumes that are not included in *LvNameFile* are re-created with default system generated names. NEWLV1 name can be the same as LV name in the *LvNameFile* stanza (LV:NEWLV1) to leave the logical volume with the same name.
-L*Label_Prefix*	Changes the labels of logical volumes on the VG being re-created to this prefix. You must modify the /etc/filesystems filepath manually if a simple modification of the mount point is not enough to define the stanza uniquely. Specifying / (slash) as the *Label_Prefix*, leaves the label in the logical volume unchanged.
-n	Specifies that after **recreatevg** the volume group is imported but varied off. Default is imported and varies on.
-p	Disables the automatic generation of the new PVIDs. If the -p flag is used, you must ensure that there are no duplicated PVIDs on the system. All the disks that are hardware that is mirrored must have their PVIDs changed to a unique value.
-0	Forces the volume group to be re-created and varied on even if the metadata on the disk indicates that this volume group is varied on in another node. See the varyonvg command for detailed information.
-V*MajorNumber*	Allows the major number of the volume group to be specified rather than having the major number generated automatically.
-y *VGname*	Allows the volume group name to be specified rather than having the name generated automatically. Volume group names must be unique system wide and can range from 1 to 15 characters. The name cannot begin with a prefix already defined in the PdDv class in the Device Configuration database for other devices. The new volume group name is sent to standard output.

Item	Description
-Y *Lv_Prefix*	Causes the logical volumes on the volume group that is being re-created to be renamed with this prefix. The total length of the prefix and the logical volume name must be less than or equal to 15 characters. If the length exceeds 15 characters, the logical volume is renamed with a default name. The default name must comply to the following conditions: • Cannot begin with a prefix that is already defined in the PdDv class of the Device Configuration database. • Cannot use a name that is already used by another system. Specifying NA as the *Lv_Prefix*, leaves all the logical volume names unchanged.

Security

Access Control: You must have root authority to run this command.

Examples

1. To re-create a volume group that contains three physical volumes, enter the command:

   ```
   recreatevg hdisk1 hdisk2 hdisk3
   ```

   The volume group on hdisk1, hdisk2, and hdisk3 is re-created with an automatically generated name, which is displayed.
2. To re-create a volume group on hdisk1 with the new name testvg, enter the command:

   ```
   recreatevg -y testvg hdisk1
   ```
3. To re-create a volume group on hdisk14, re-create all logical volumes in that volume group, and rename them with the prefix newlv, enter the command:

   ```
   recreatevg -Y newlv hdisk14
   ```

Files

Item	Description
`/usr/sbin`	Directory where the **recreatevg** command is present.

recsh Command

Purpose

Invokes the recovery shell.

Syntax

recsh

Description

When the **libc.a** library is moved or renamed, an error message `Killed` will be displayed from the shell as there is no **libc.a** library available for the system to load and run the utilities. The **recsh** command

invokes recovery shell, which provides the ability to rename **libc.a** library if it is accidently moved. It uses an alternative **libc.a** library that is shipped with the system.

Note: This is a recovery shell and users should not use **recsh** as default shell.

Examples

1. If libc.a is renamed accidentally then the system will be in an unstable state where in execution of any utility will not be possible. To recover at this point, type:

```
recsh; cp -p  libc.a.new /usr/lib/libc.a; exit
```

Location

/usr/bin/recsh

Files

Item	Description
/usr/bin/recsh	Specifies the path name to the recovery shell.

redefinevg Command

Purpose

Redefines the set of physical volumes of the given volume group in the device configuration database.

Syntax

redefinevg { **-d** *Device* | **-i** *Vgid* } *VolumeGroup*

Description

During normal operations the device configuration database remains consistent with the Logical Volume Manager (LVM) information in the reserved area on the physical volumes. If inconsistencies occur between the device configuration database and the LVM, the **redefinevg** command determines which physical volumes belong to the specified volume group and re-enters this information in the device configuration database. The **redefinevg** command checks for inconsistencies by reading the reserved areas of all the configured physical volumes attached to the system.

> **Note:** To use this command, you must either have root user authority or be a member of the **system** group.

Flags

Item	Description
-d *Device*	The volume group ID, *Vgid*, is read from the specified physical volume device. You can specify the *Vgid* of any physical volume belonging to the volume group that you are redefining.
-i *Vgid*	The volume group identification number of the volume group to be redefined.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

To redefine rootvg physical volumes in the Device Configuration Database, enter a command similar to the following:

```
redefinevg -d hdisk0 rootvg
```

Files

Item	Description
/usr/sbin/synclvodm	Contains the **synclvodm** command.

reducevg Command

Purpose

Removes physical volumes from a volume group. When all physical volumes are removed from the volume group, the volume group is deleted.

Syntax

reducevg [**-d**] [**-f**] *VolumeGroup PhysicalVolume ...*

Description

> **Attention:** You can use the **reducevg** command while the volume group is in concurrent mode. However, if you run this command while the volume group is in concurrent mode and the end result is the deletion of the volume group, then the **reducevg** command will fail.

The **reducevg** command removes one or more physical volumes represented by the *PhysicalVolume* parameter from the *VolumeGroup*. When you remove all physical volumes in a volume group, the volume group is also removed. The volume group must be varied on before it can be reduced.

All logical volumes residing on the physical volumes represented by the *PhysicalVolume* parameter must be removed with the **rmlv** command or the **-d** flag before starting the **reducevg** command.

Note:

1. To use this command, you must either have root user authority or be a member of the **system** group.
2. Sometimes a disk is removed from the system without first running **reducevg** *VolumeGroup PhysicalVolume*. The VGDA still has this removed disk in it's memory, but the *PhysicalVolume* name no longer exists or has been reassigned. To remove references to this missing disk you can still use **reducevg**, but with the Physical Volume ID (PVID) instead of the disk name: **reducevg** *VolumeGroup PVID*.
3. You cannot use the **reducevg** command on a snapshot volume group.
4. You cannot use the **reducevg** command on a volume group that has an active firmware assisted dump logical volume.
5. The `reducevg` command discards any background space reclamation process that is running for the physical volumes that are removed from the volume group. To identify whether a space reclamation is running, you can use the `lvmstat` command with **`-r`** option.

For volume groups created on AIX 5.3 and varied on without the **varyonvg -M** flag, reducevg will dynamically raise the logical track group size for the volume group if necessary to match the common max transfer size of the remaining physical volumes.

You could also use the System Management Interface Tool (SMIT) **smit reducevg** fast path to run this command.

Flags

Item	Description
-d	Deallocates the existing logical volume partitions and then deletes resultant empty logical volumes from the specified physical volumes. User confirmation is required unless the **-f** flag is added. **Attention:** The **reducevg** command with the **-d** flag automatically deletes all logical volume data on the physical volume before removing the physical volume from the volume group. If a logical volume spans multiple physical volumes, the removal of any of those physical volumes may jeopardize the integrity of the entire logical volume.
-f	Removes the requirement for user confirmation when the **-d** flag is used.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To remove physical volume `hdisk1` from volume group `vg01`, enter:

   ```
   reducevg vg01 hdisk1
   ```

2. To remove physical volume `hdisk1` and all residing logical volumes from volume group `vg01` without user confirmation, enter the following command. **Attention:** The **reducevg** command with the **-d** flag automatically deletes all logical volume data before removing the physical volume.

   ```
   reducevg  -d  -f vg01 hdisk1
   ```

   The physical volume `hdisk1` and all residing logical volumes are removed.

Files

Item	Description
/usr/sbin/reducevg	Directory where the **reducevg** command resides.
/tmp	Directory where the temporary files are stored and while the command is running.

refer Command

Purpose

Finds and inserts literature references in documents.

Syntax

refer [**-b**] [**-e**] [**-P**] [**-S**] [**-a** [*Number*]] [**-B** *Label***.***Macro*] [**-c** *Keys*] [**-f** *Number* | **-k** *Label* | **-l** *Letter***,** *Digit*] [**-n**] [**-p** *Reference*] [**-s** *Keys*] [*File ...*]

Description

The **refer** command is a preprocessor for the **nroff** or the **troff** command. The **refer** command finds and formats references for footnotes or endnotes. It is also the basis for a series of programs designed to index, search, sort, and print standalone bibliographies or other data entered in the appropriate form.

Given an incomplete citation with sufficiently precise phs, the **refer** command searches a bibliographic database for references containing these phs anywhere in the title, author, journal, and so on. The input file (or else standard input) is copied to standard output, except for lines enclosed by the .[(period, left bracket) and .] (period, right bracket) delimiters. Lines enclosed by the delimiters are assumed to contain phs and are replaced by information from the bibliographic database. The user can search different databases, override particular fields, or add new fields. The reference data, from whatever source, is assigned to a set of **troff** command strings. Macro packages, such as the **ms** macro package, print the finished reference text from these strings. By default, references are flagged by footnote numbers.

To use your own references, put them in the format described in the Example section. These references can be accessed either by using the **-p** flag or by setting the **REFER** environment variable to those reference files. The references can be searched more rapidly by running the **indxbib** command on them before using the **refer** command. If you do not index, a linear search is made. When the **refer** command is used with any of the preprocessor commands (**eqn**, **neqn**, or **tbl** command), the **refer** command should be issued first, to minimize the volume of data passed through pipes.

> **Note:** Anytime you edit a reference file, you must reissue the **indxbib** command on that file. If you do not use the **indxbib** command, remove any **.ia**, **.ib**, **.ic**, and **.ig** files associated with that reference file; otherwise, you will get a `too many hits` error message from the **refer** command.

The **refer** command and associated programs expect input from a file of references composed of records separated by blank lines. A record is a set of fields (lines), each containing one kind of information. Fields start on a line beginning with the **%** (percent sign), followed by a key letter, a space character, and finally the contents of the field, and continue until the next line, starting with a **%** (percent sign). The output ordering and formatting of fields is controlled by the macros specified for the **nroff** and **troff** commands (for footnotes and endnotes), or the **roffbib** command (for standalone bibliographies). For a list of the most common key letters and their corresponding fields, see the **addbib** command.

Flags

Item	Description
-b	Bare mode: do not put any flags in text (either numbers or labels).
-e	Instead of leaving the references where encountered, accumulates them until a sequence of the following form is encountered: `.[` `$LIST$` `.]` then writes out all references collected so far.
-P	Places punctuation marks after the reference signal, rather than before. The punctuation marks are locale-specific and are defined in the **refer message catalog**.
-S	Produces references in the natural or social science format.
-a *Number*	Reverses the first specified number of author names (Jones, J. A. instead of J. A. Jones). If the *Number* variable is omitted, all author names are reversed.

Item	Description
-B *Label***.***Macro*	Specifies bibliography mode. Takes a file composed of records separated by blank lines and turns that file into **troff** command input. The specified label is turned into the specified macro, with the *Label* variable value defaulting to **%X** and the *.Macro* variable value defaulting to **.AP** (annotation paragraph).
-c *Keys*	Capitalizes, with SMALL CAPS, the fields whose key letters are in the *Keys* variable. For example, Jack becomes JACK .
-f *Number*	Sets the footnote number to the specified number instead of the default of 1. With labels rather than numbers, this flag has no effect. See the **-k** flag and the **-l** flag.
-k *Label*	Instead of numbering references, uses labels as specified in a reference data line beginning with **%***Label*. By default, the *Label* variable value is **L.**
-l *Letter***,***Digit*	Instead of numbering references, uses labels made from the senior author's last name and the year of publication. Only the first specified letters of the last name and the last specified digits of the date are used. If either the *Letter* variable or the *Digit* variable is omitted, the entire name or date, respectively, is used.
-n	Does not search the default **/usr/share/dict/papers/Ind** file .If the **REFER** environment variable is set, the specified file is searched instead of the default file. In this case, the **-n** flag has no effect.
-p *Reference*	Takes the *Reference* variable as a file of references to be searched. The default file is searched last.
-s *Keys*	Sorts references by fields whose key letters are specified by the *Keys* variable string. Renames reference numbers in text accordingly. Implies the **-e** flag. The key letters specified by the *Keys* variable can be followed by a number to indicate how many such fields are used, with q + (plus sign) indicating a very large number. The default value is **AD**, which sorts first by senior author and then by date. For example, to sort on all authors and then title, enter `-sA+T`**.** It is important to note that blank spaces at the end of lines in bibliography fields cause the records to sort and reverse incorrectly. Sorting large numbers of references can cause a core dump.

Example

Following is an example of a **refer** command entry:

```
%A M.E. Lesk

%T Some Applications of Inverted Indexes on the UNIXSystem

%B UNIXProgrammer's Manual

%V 2b

%I Bell Laboratories
%C Murray Hill, NJ

%D 1978
```

Files

Item	Description
/usr/share/dict/papers/Ind	Contains the default reference file.
/usr/lbin/refer	Contains companion programs.

refile Command

Purpose

Moves files between folders.

Syntax

refile [**-src** **+***Folder*] [**-draft**] [**-file** *File*] [*Messages*] [**-nolink** | **-link**] [**-nopreserve** | **-preserve**] **+***Folder* ...

Description

The **refile** command moves messages between folders. If you do not specify a source folder, the **refile** command uses the current folder as the source. If you specify a destination folder that does not exist, the system requests permission to create it.

The **refile** command also copies messages from one folder to another. When moving a message, by default, the system does not keep a copy of the message in the original folder. To leave a copy behind, use the **-preserve** flag.

Flags

Item	Description
-draft	Copies the current draft message from your mail directory.
-file *File*	Copies the specified file. The file must be in valid message format. Use the **inc** command to format and file new messages correctly.
+*Folder*	Copies the messages to the specified folder. Any number of folders can be specified.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-link	Leaves the messages in the source folder or file after they are copied.

Item	Description
Messages	Specifies the messages to be copied. You can specify several messages, a range of messages, or a single message. Use the following references to specify messages: ***Number*** Number of the message. ***Sequence*** A group of messages specified by the user. Recognized values include: **all** All the messages in a folder. **cur or . (period)** Current message. This is the default. **first** First message in a folder. **last** Last message in a folder. **next** Message following the current message. **prev** Message preceding the current message. **/DT>** If the **-link** and **all** flags are used together, the current message in the current folder does not change. Otherwise, if a message is specified, the refiled message becomes the current message.
-nolink	Removes the messages from the source folder or file after they are copied. This flag is the default.
-nopreserve	Renumbers the messages that are copied. Renumbering begins with a number one higher than the last message in the destination folder. This flag is the default.
-preserve	Preserves the message numbers of copied messages. If messages with these numbers already exist, the **refile** command issues an error message and does not alter the contents of the folders.
-src +*Folder*	Identifies the source folder. By default, the system uses the current folder.

Profile Entries

The following entries are part of the *UserMHDirectory*/**.mh_profile** file:

Item	Description
`Current-Folder:`	Sets the default current folder.
`Folder-Protect:`	Sets the protection level for your new folder directories.
`Path:`	Specifies the *UserMhDirectory*.
`rmmproc:`	Specifies the program used to remove messages from a folder.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To refile the current message from the current folder and place it in a new folder called `meetings`, enter:

   ```
   refile  +meetings
   ```

   The system responds with a message similar to the following:

   ```
   Create folder "/home/jeanne/Mail/meetings"?
   ```

   Enter y to create the folder. A copy of the original message is not retained in the current folder.
2. To copy the current message from the current folder and to the `meetings` folder, enter:

   ```
   refile  -link +meetings
   ```

   The original message remains in the current folder.
3. To refile the current message draft into the `test` folder, enter:

   ```
   refile  -draft +test
   ```

   A copy of the message draft is not retained in the current folder.
4. To refile the current message from the current folder and into several folders, enter:

   ```
   refile  +tom +pat +jay
   ```

   A copy of the message is not retained in the current folder.

Files

Item	Description
$HOME/.mh_profile	Sets the MH user profile.
/usr/bin/refile	Contains the **refile** command.

refresh Command

Purpose

Requests a refresh of a subsystem or group of subsystems.

Syntax

refresh [**-h** *Host*] { **-g** *Group*| **-p** *SubsystemPID*| **-s** *Subsystem*}

Description

The **refresh** command sends the System Resource Controller a subsystem refresh request that is forwarded to the subsystem. The refresh action is subsystem-dependent.

> **Note:** The **refresh** command is unsuccessful if the communication method for the subsystems is signals.

Flags

Item	Description
-g *Group*	Specifies a group of subsystems to refresh. The **refresh** command is unsuccessful if the *Group* name is not contained in the subsystem object class.
-h *Host*	Specifies the foreign *Host* machine on which this refresh action is requested. The local user must be running as "root". The remote system must be configured to accept remote System Resource Controller requests. That is, the **srcmstr** daemon (see **/etc/inittab**) must be started with the **-r** flag and the **/etc/hosts.equiv** or **.rhosts** file must be configured to allow remote requests.
-p *SubsystemPID*	Specifies a particular instance of the subsystem to refresh.
-s *Subsystem*	Specifies a subsystem to refresh. The *Subsystem* name can be the actual subsystem name or the synonym name for the subsystem. The **refresh** command is unsuccessful if *Subsystem* name is not contained in the subsystem object class.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To refresh the a group, like `tcpip`, enter:

   ```
   refresh  -g tcpip
   ```

2. To refresh a subsystem, like `xntpd`, enter:

   ```
   refresh  -s xntpd
   ```

Files

Item	Description
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration Object Class.
/etc/services	Defines the sockets and protocols used for Internet services.
/dev/SRC	Specifies the **AF_UNIX** socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.

refrsrc Command

Purpose

Refreshes the resources within the specified resource class.

Syntax

`refrsrc` [`-h`] [`-TV`] *resource_class*

Description

The `refrsrc` command refreshes the resources within the specified resource class. Use this command to force the Resource Monitoring and Control (RMC) subsystem to detect new instances of resources in cases where the configuration could be altered by operating system commands (`mkfs`, for example).

This command makes a request to the RMC subsystem to refresh the configuration of the resources within a resource class. The request is actually performed by the linked resource manager.

Any application that is monitoring resources in the specified resource class may receive events as the configuration is refreshed.

Flags

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software-service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

resource_class
Specifies the resource class name.

Security

The user needs read permission for the *Resource_class* specified in `refrsrc` to run `refrsrc`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for information about the ACL file and how to modify it.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output.

The command output and all verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To refresh the configuration of the resources in class IBM.FileSystem, enter:

```
refrsrc IBM.FileSystem
```

Location

/opt/rsct/bin/refrsrc

refsensor Command

Purpose

Refreshes a sensor or a microsensor defined to the resource monitoring and control (RMC) subsystem.

Syntax

To refresh a sensor:

`refsensor` [–a | –n *host1*[,*host2*...] | –N { *node_file* | "–" }] [–h] [–v | –V] *sensor_name*

To refresh a microsensor:

`refsensor –m [–a | –n host1[,host2...] | –N { node_file | "–" } ] [–h ] [ –v | –V ] sensor_name`

Description

The `refsensor` command refreshes a sensor or microsensor resource that is defined to the RMC subsystem. *Sensors* and *microsensors* are RMC resources with attributes that can be monitored. Sensors and microsensors must be monitored for `refsensor` to run successfully.

A sensor can be refreshed using `refsensor` in one of two ways: either by running the sensor command that is defined for the sensor resource or by specifying values for specific sensor attributes. A microsensor can be refreshed using `refsensor` to query the values of the microsensor's load module. Use the `-m` flag to refresh a microsensor.

When the `refsensor` command runs, it does not affect the interval, if any, that is defined (using `mksensor`) for running the sensor command or for querying the microsensor load module . That is, if a monitored sensor or microsensor is being updated every 60 seconds, running `refsensor` does not cause the interval timer to be reset to 60 seconds.

The `refsensor` command runs on any node. If you want `refsensor` to run on all of the nodes in a domain, use the `-a` flag. If you want `refsensor` to run on a subset of nodes in a domain, use the `-n` flag. Instead of specifying multiple node names using the `-n` flag, you can use the `-N` *node_file* flag to indicate that the node names are in a file. Use `-N "–"` to read the node names from standard input.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM `nodegrp` command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

To have `refsensor` update specific sensor attributes, specify one or more *attr=value* parameters. Only the attributes specified will be updated. No other sensor attributes will be updated. The sensor attributes that can be specified as parameters are:

Float32
The type `float32` attribute for this sensor resource

Float64
The type `float64` attribute for this sensor resource

Int32
The type `int32` attribute for this sensor resource

Int64
The type `int64` attribute for this sensor resource

Quantum
The type `quantum` attribute for this sensor resource

String
The type `string` attribute for this sensor resource

Uint32
The type `uint32` attribute for this sensor resource

Uint64
The type `uint64` attribute for this sensor resource

For example, to update the `Int32` and `Float32` sensor attributes for the sensor named `Sensor1`, enter:

```
refsensor Sensor1 Int32=45 Float32=7.8
```

Microsensor attributes cannot be updated separately.

Flags

–a
Refreshes sensors that match the specified name on all nodes in the domain.
The CT_MANAGEMENT_SCOPE environment variable determines the cluster scope. If CT_MANAGEMENT_SCOPE is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management domain and a peer domain exist, `refsensor -a` with CT_MANAGEMENT_SCOPE not set will run in the management domain. In this case, to run in the peer domain, set CT_MANAGEMENT_SCOPE to 2.

–m
Specifies that the resource to be refreshed is a microsensor resource.

–n *host1*[,*host2*...]
Specifies one or more nodes on which the sensor should be refreshed. By default, the sensor is refreshed on the local node. This flag is only appropriate in a management domain or a peer domain.

–N { *node_file* | "–" }
Specifies that node names are read from a file or from standard input.

Use `-N` *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (#) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use `-N "–"` in a management domain or a peer domain to read the node names from standard input.

–h
Writes the command's usage statement to standard output.

–v | –V
Writes the command's verbose messages to standard output.

Parameters

sensor_name
Specifies the name of the sensor to be refreshed.

attr=value
Specifies which sensor attributes will be refreshed and the values to which they will be set.

Security

To refresh sensors using this command, you need write permission for the `IBM.Sensor` resource class.

To refresh microsensors using this command, you need write permission for the `IBM.MicroSensor` resource class.

Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0
The command has run successfully.

1
An incorrect combination of flags and parameters has been entered.

4
The sensor is not monitored and cannot be refreshed.

6
No sensor resources were found.

n
Based on other errors that can be returned by the RMC subsystem.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If this environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled.

The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the `rsct.core` fileset for AIX and `rsct.core-3.1.0.0-0.`*platform*`.rpm` package for Linux, Solaris, and Windows, where *platform* is `i386`, `ppc`, `ppc64`, `s390`, or `x86_64`.

Examples

1. To refresh the sensor called `Sensor1` so that its defined sensor command is run, enter:

```
refsensor Sensor1
```

2. To refresh the sensor called `Sensor1` so that `Int32` is set to `50`, `Float32` is set to `123.45`, and `String` is set to `"test input"`, enter:

```
refsensor Sensor1 Int32=50 Float32=123.45 String="test input"
```

3. To refresh the sensor called `Sensor1` on the nodes that are listed in the `/u/joe/common_nodes` file so that `Sensor1`'s defined sensor command is run, enter:

```
refsensor -N /u/joe/common_nodes Sensor1
```

where /u/joe/common_nodes contains:

```
# common node file
#
node1.myhost.com    main node
node2.myhost.com    backup node
```

4. To refresh the microsensor called IBM.Sensor1 so that the attribute values are queried using the defined microsensor load module, enter:

```
refsensor -m IBM.Sensor1
```

Location

/opt/rsct/bin/refsensor

regcmp Command

Purpose

Compiles patterns into C language **char** declarations.

Syntax

regcmp [**-**] *File* [*File* ...]

Description

The **regcmp** command compiles the patterns in *File* and places output in a *File***.i** file, or a *File***.c** file when the **-** option is specified. The resulting compiled patterns are initialized **char** declarations. Each entry in *File* must be a C variable name followed by one or more blanks, followed by a pattern enclosed in **"** **"** (double quotation marks).

The output of the **regcmp** command is C source code. A resulting *File***.i** file can be included in C programs, and a resulting *File***.c** file can be a file parameter to the **cc** command.

A C language program that uses the output of the **regcmp** command should use the **regex** subroutine to apply it to a string.

In most cases, the **regcmp** command makes unnecessary the use of the **regcmp** subroutine in a C language program, saving execution time and program size.

Flag

Item	Description
-	Places the output in a *File***.c** file. The default is to put the output in *File.***i**.

Examples

1. To compile the patterns in stdin1 and the patterns in stdin2, enter:

```
regcmp stdin1 stdin2
```

This creates the stdin1.i and stdin2.i files.

2. To creates stdin1.c and stdin2.c files, enter:

```
regcmp - stdin1 stdin2
```

Note: Assuming that the same `stdin1` and `stdin2` files are used in both examples, the resulting `stdin1.i` and `stdin1.c` files are identical, and the resulting `stdin2.i` and `stdin2.c` files are identical.

File

Item	Description
/usr/ccs/bin/regcmp	Contains the **regcmp** command.

rembak Command

Purpose

Sends a print job to a queue on a remote server.

Syntax

rembak **-S** *Server* **-P** *Queue* [**-R**] [**-N** *Filter*] [**-L**] [**-p**] [**-q**] [**-x**] [**-#** *JobNumber*] [**-u** *UserName*] [**-X**] [**-o** *Option*] [**-T** *Timeout*] [**-C**] [**-D** *DebugOutputFile*] [*File* ...]

Description

The **rembak** command sends a job to be queued on a remote server. The request can either be a print job, a status request, a job cancel request, or a request to kill the remote queuing system. The server and the queue flags are required. All the other flags are optional, depending on what needs to be done.

This command should only be called by the **qdaemon** command. It is not intended to be entered on the command line by a user. See the **enq** command for details on how to issue a print job request, or use the System Manager Interface Tool (SMIT) to request a print job.

Flags

Item	Description
-# *JobNumber*	Specifies the *JobNumber* to cancel.
-C	Sends control file first. The **lpd** protocol allows two handshaking sequences for processing a print job. The default consists of sending the data files(s) first followed by the control file. The other sequence is to send the control file first followed by the data file(s). If **-C** is specified, **rembak** will send the control file first followed by the data file(s).
-D *DebugOutputfile*	Turns on the debugging option for **rembak**. If no output file name is specified, or if there are any problems creating or writing to the output file, the debugging option is ignored. If the output file specified already exists, new debugging output is appended to the end of it.
-L	Indicates a long (verbose) status request from the remote queue.

Item	Description
-N *Filter*	Indicates the machine type of the remote server. The filter name is specified by the **s_statfilter** attribute in the **/etc/qconfig** file. Values for the *filter* variable include the following: **/usr/lib/lpd/aixshort** Indicates the server is another AIX machine. **/usr/lib/lpd/aixv2short** Indicates the server is an RT with an AIX Version 2 operating system. **/usr/lib/lpd/bsdshort** Indicates the server is a bsd machine **/usr/lib/lpd/attshort** Indicates the server is an AT&T machine
-o *Option*	Specifies an *Option* to be sent to the backend on the remote server. (These *Options* are passed through the **rembak** command.)
-p	Indicates that the port range used by **rembak** is restricted to ports below 1023.
-P *Queue*	Specifies the name of the *Queue* on the remote server where the print job is sent.
-q	Indicates a short (abbreviated) status request from the remote queue.
-R	Restarts the remote queuing system. **Note:** The **-R** flag is not supported when sending a request to an operating system. The **lpd** daemon does not support such a request. The **-R** flag is supported only for compatibility with other systems.
-S *Server*	Specifies the name of the remote print *Server* where the print request is sent.
-T *Timeout*	Sets a timeout period, in minutes, for **rembak** to wait for acknowledgements from the remote server. If no value is specified, a default timeout of 90 seconds is used. This default is also used if Timeout is 0 or a negative value.
-u *UserName@HostName*	Cancels a print job for *UserName* that was submitted from the *HostName* machine. **Note:** The queuing system does not support multibyte host names.
-X	Specifies that the **rembak** command send the **-o** *Option* to the remote server, even if the remote server is a non-AIX machine. If the remote is a non-AIX machine, then the *Option* is sent without the **-o** flag. Thus, **-o -abc** is sent as **-abc**. To use the **-X** flag on a remote queue, the following line for the specific queue must be included in the **/etc/qconfig** file: `backend = /usr/lib/lpd/rembak -X` The **qprt**, **lpr** and other queuing commands are not guaranteed to work when -X is specified on a queue. Use the **enq** command.
-x	Cancels a job request. Use the **-#** *JobNumber* flag or the **-u** *UserName* flag to cancel a request.

Examples

1. To print the files `spinach`, `asparagus`, and `broccoli` on the queue `popeye` on the remote server `olive`, which is an RT with an AIX Version 2 operating system, enter:

```
rembak -S olive -P popeye -N /usr/lib/lpd/aixv2short spinach asparagus broccoli
```

2. To issue a verbose status request to `olive` for the queue `popeye`, enter:

```
rembak -S olive -P popeye -N /usr/lib/lpd/aixv2short -L
```

3. To cancel job number 23 on a remote server submitted by user `sweetpea` from machine `bluto`, which is a Version 3 machine, enter:

```
rembak -S olive -P popeye -N /usr/lib/lpd/aixv2short -x -#23 -u sweetpea@bluto
```

Files

Item	Description
/usr/lib/lpd/rembak	Contains the **rembak** command.
/etc/hosts.lpd	Contains host names that are allowed to do print requests.
/etc/hosts.equiv	Contains host names that are allowed to do print requests.

remove Command

Purpose

Deletes files from **var/adm/acct/sum** and **var/adm/acct/nite** subdirectories.

Syntax

/usr/sbin/acct/remove

Description

The **remove** command deletes all **/var/adm/acct/sum(x)/wtmp***, **/var/adm/acct/sum(x)/pacct***, and **/var/adm/acct/nite(x)/lock*** files. The **remove** command must be scheduled with the **cron** daemon. Also, the **remove** command should be run at the end of every accounting period, rather than every night.

Security

Access Control: This command should grant execute (x) access only to members of the **adm** group.

Files

Item	Description
/usr/sbin/acct	The path to the accounting commands.
/var/adm/acct/nite	Contains accounting data files.
/var/adm/acct/nitex	Contains accounting data files when user names greater than 8 characters are used.
/var/adm/acct/sum	Cumulative directory for daily accounting records.
/var/adm/acct/sumx	Cumulative directory for daily accounting records when user names greater than 8 character are used.

removevsd Command

Purpose

Removes a set of virtual shared disks.

Syntax

removevsd
{**-v** *vsd_names* | **-a**} [**-f**]

Description

Use this command to remove the logical volumes associated with the virtual shared disks. Volume groups are not removed with this command.

If the virtual shared disk is configured on any of the nodes on the system partition, this command is unsuccessful, unless the **-f** flag is specified.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit delete_vsd
```

and select the **Remove a Virtual Shared Disk** option.

Flags

-v
Specifies the virtual shared disk name or names that are to be removed by this command.

-a
Specifies that the command should remove all virtual shared disks in the RSCT peer domain.

-f
Forces the system to unconfigure the virtual shared disks and remove them. If **-f** is not specified and any of the virtual shared disks that are to be removed are configured, the command is unsuccessful.

Parameters

vsd_name
Specifies a virtual shared disk. If the virtual shared disk is not in the stopped state, you will get an error message.

Security

You must have `root` authority to run this command.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to *RSCT Administration Guide* .

Examples

To unconfigure and remove all defined virtual shared disks in a system or system partition, enter:

```
removevsd -a -f
```

Location

`/opt/rsct/vsd/bin/removevsd`

rendev Command

Purpose

Renames a device.

Syntax

```
rendev -l Name  -n NewName [-u]
```

Description

The **rendev** command enables devices to be renamed. The device to be renamed, is specified with the **-l** flag, and the new desired name is specified with the **-n** flag.

The new desired name must not exceed 15 characters in length. If the name has already been used or is present in the `/dev` directory, the operation fails. If the name formed by appending the new name after the character r is already used as a device name, or appears in the `/dev` directory, the operation fails.

If the device is in the Available state, the **rendev** command must unconfigure the device before renaming it. This is similar to the operation performed by the **rmdev –l Name** command. If the unconfigure operation fails, the renaming will also fail. If the unconfigure succeeds, the **rendev** command will configure the device, after renaming it, to restore it to the Available state. The **–u** flag may be used to prevent the device from being configured again after it is renamed.

Note: Disk drive devices that are members of the root volume group, or that will become members of the root volume group (by means of LVM or install procedures), must not be renamed. Renaming such disk drives may interfere with the ability to recover from certain scenarios, including boot failures.

Some devices may have special requirements on their names in order for other devices or applications to use them. Using the **rendev** command to rename such a device may result in the device being unusable.

Note: To protect the configuration database, the **rendev** command cannot be interrupted once it has started. Trying to stop this command before completion, could result in a corrupted database.

Flags

Item	Description
-l *Name*	Specifies the device, indicated by the *Name* parameter, to be renamed in the customized devices object.
-n *NewName*	Specifies the new name, indicated by the *NewName* parameter, to be assigned to the device.
-u	Optional flag, which indicates that the device is not to be configured after it is renamed.

Examples

1. To rename disk hdisk5 to hdisk2, enter:

   ```
   rendev -l hdisk5 -n hdisk2
   ```

2. To rename disk hdisk3 to ootvg, enter:

   ```
   rendev -l hdisk3 -n ootvg
   ```

The second command fails because **ootvg** appended to **r** results in the name **rootvg**, which conflicts with the rootvg volume group name.

renice Command

Purpose

Alters the nice value of running processes.

Syntax

renice [**-n** *Increment*] [**-g** | **-p** | **-u**] *ID ...*

Description

The **renice** command alters the nice value of one or more running processes. The *nice value* is the decimal value of the system scheduling priority of a process. By default, the processes affected are specified by their process IDs. When you specify a process group, the request applies to all processes in the process group.

The nice value is determined in an implementation-dependent manner. If the requested increment raises or lowers the nice value of the executed utility beyond implementation-dependent limits, the limit whose value was exceed is used.

If you do not have root user authority, you can only reset the priority of processes you own and can only increase their priority within the range of 0 to 20, with 20 being the lowest priority. If you have root user authority, you can alter the priority of any process and set the priority to any value in the range -20 to 20. The specified *Increment* changes the priority of a process in the following ways:

Item	Description
1 to **20**	Runs the specified processes slower than the base priority.
0	Sets priority of the specified processes to the base scheduling priority.
-20 to **-1**	Runs the specified processes quicker than the base priority.

The **renice** command maps these values to those actually used by the kernel.

Notes:

1. If you do not have root user authority, you cannot increase the nice value of processes (even if you had originally decreased their priorities).
2. You cannot use the **renice** command to change a process to run at a constant priority. To do this, use the **setpriority** system call.

Flags

Item	Description
-g	Interprets all IDs as unsigned decimal integer process group IDs.
-n *Increment*	Specifies the number to add to the nice value of the process. The value of *Increment* can only be a decimal integer from -20 to 20. Positive increment values cause a lower nice value. Negative increment values require appropriate privileges and cause a higher nice value.
-p	Interprets all IDs as unsigned integer process IDs. The -p flag is the default if you specify no other flags.
-u	Interprets all IDs as user name or numerical user IDs.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion
>0	An error occurred.

Examples

1. To alter the system scheduling priority so that process IDs 987 and 32 have lower scheduling priorities, enter:

   ```
   renice -n 5 -p 987 32
   ```

2. To alter the system scheduling priority so that group IDs 324 and 76 have higher scheduling priorities (if the user has the appropriate privileges to do so), enter:

   ```
   renice -n -4 -g 324 76
   ```

3. To alter the system scheduling priority so that numeric user ID 8 and user `sas` have lower scheduling priorities, enter:

   ```
   renice -n 4 -u 8 sas
   ```

Files

Item	Description
/usr/sbin/renice	Contains the **renice** command.
/etc/passwd	Maps user names to user IDs.

reorgvg Command

Purpose

Reorganizes the physical partition allocation for a volume group.

Syntax

reorgvg [**-i**] *VolumeGroup* [*LogicalVolume ...*]

Description

The **reorgvg** command reorganizes the placement of allocated physical partitions within the *VolumeGroup*, according to the allocation characteristics of each logical volume. Use the *LogicalVolume* parameter to reorganize specific logical volumes; highest priority is given to the first logical volume name in the *LogicalVolume* parameter list and lowest priority is given to the last logical volume in the parameter list. The volume group must be varied on and must have free partitions before you can use the **reorgvg** command.

The relocatable flag of each logical volume must be set to **y** with the **chlv -r** command for the reorganization to take effect; otherwise, the logical volume is ignored.

Note:

1. The **reorgvg** command does not reorganize the placement of allocated physical partitions for any striped logical volumes.
2. At least one free physical partition (PP) must exist on the specified volume group for the **reorgvg** command to run successfully. For mirrored logical volumes, one free PP per physical volume (PV) is required in order for the **reorgvg** command to maintain logical volume strictness during execution; otherwise the **reorgvg** command still runs, but moves both copies of a logical partition to the same disk during its execution.
3. To use this command, you must either have root user authority or be a member of the **system** group.
4. If you enter the **reorgvg** command with the volume group name and no other arguments, the entire volume group is reorganized.
5. You cannot use the **reorgvg** command on a snapshot volume group or a volume group that has a snapshot volume group.
6. You cannot use the **reorgvg** command on a volume group that has an active firmware assisted dump logical volume.

You could also use the System Management Interface Tool (SMIT)**smit reorgvg** fast path to run this command.

Flags

Item	Description
-i	Specifies physical volume names read from standard input. Only the partitions on these physical volumes are organized.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To reorganize logical volumes `lv03`, `lv04`, and `lv07` on volume group `vg02`, enter:

   ```
   reorgvg vg02 lv03 lv04 lv07
   ```

   Only the listed logical volumes are reorganized on `vg02`.
2. To reorganize only the partitions located on physical volumes `hdisk4` and `hdisk6` that belong to logical volumes `lv203` and `lv205`, enter:

   ```
   echo "hdisk4 hdisk6" | reorgvg -i vg02 lv203 lv205
   ```

   The partitions located on physical volumes `hdisk4` and `hdisk6` of volume group `vg02`, that belong to logical volumes `lv203` and `lv205`, are reorganized.

Files

Item	Description
/usr/sbin/reorgvg	Directory where the **reorgvg** command resides.
/tmp	Directory where the temporary files are stored while the command is running.

repl Command

Purpose

Replies to a message.

Syntax

repl [**+***Folder*] [**-draftfolder +***Folder* | **-nodraftfolder**] [*Message*] [**-draftmessage***Message*] [**-annotate** [**-noinplace** | **-inplace**] | **-noannotate**] [**-cc** *Names...*] [**-nocc** *Names...*] [**-query** | **-noquery**] [**-fcc +***Folder*] [**-form** *FormFile*] [**-editor** *Editor* | **-noedit**] [**-format** | **-noformat**] [**-filter** *File*] [**-width** *Number*] [**-whatnowproc** *Program* | **-nowhatnowproc**]

Description

The **repl** command starts an interface enabling you to compose a reply to a message. By default, the command drafts a reply to the current message in the current folder. If you do not specify the **-draftfolder** flag, or if the `Draft-Folder:` entry in the **$HOME/.mh_profile** file is undefined, the **repl** command searches your MH directory for a **draft** file. If you specify a folder, that folder becomes the current folder.

When you enter the **repl** command, the system places the `To:`, `cc:`, and `In-Reply-To:` fields in the draft and prompts you to enter the text of the reply. To exit the editor, press Ctrl-D. After exiting the editor, the **repl** command starts the MH **whatnow** command. You can see a list of available **whatnow** subcommands by pressing the Enter key at the `What now?` prompt. With these subcommands, you can re-edit, list, and send a reply, or end the processing of the **repl** command.

> **Note:** A line of dashes or a blank line must be left between the header and the body of the message for the message to be identified when it is sent.

The **repl** command uses the definitions in the **/etc/mh/replcomps** file to format the reply message. You can create a **replcomps** file in your MH directory or use the **-form** flag to define an alternate reply format. To leave a copy of the original message in the reply message, use the **-filter** flag.

To annotate the original message with redistribution information, use the **-annotate** flag. This flag annotates the original message with the `Resent:` field and the current date and time. A message is annotated only if you send the reply before you exit **repl** command processing.

Flags

Item	Description
-annotate	Annotates the message being replied to with the time and date of the reply. You can use the **-inplace** flag to preserve links to an annotated message.
-cc *Names*	Specifies the users who will be listed in the `cc:` field of the reply. You can specify the following variables for *Names*: **all**, **to**, **cc**, and **me**. The default is **-cc all**.
-draftfolder +*Folder*	Places the draft message in the specified folder. If **+***Folder* is not specified, then `Current-Folder` is assumed.
-draftmessage *Message*	Specifies the draft message. If you specify **-draftfolder** without the **-draftmessage** flag, the default message is new. If you specify this flag without the **-draftfolder** flag, the system creates the draft in the default file, *UserMHdirectory*/**draft**.
-editor *Editor*	Identifies the initial editor for composing the reply. If you do not specify the **-editor** flag, the **comp** command selects the default editor specified by the `Editor:` entry in your **$HOME/.mh_profile** file.

Item	Description
-fcc +*Folder*	Places a file copy of the reply in the specified folder. If you do not specify this flag, the **repl** command will not produce a file copy.
-filter *File*	Reformats the message being replied to and places the reformatted message in the body of the reply. You must specify a *File* variable with this flag. The **-filter** flag uses the format file acceptable to the **mhl** command.
+*Folder*	Identifies the folder that contains the message to reply to. If a folder is not specified, then `Current-Folder` is used.
-form *FormFile*	Specifies a reply format. The **repl** command treats each line in the specified format file as a format string.
-format	Removes duplicate addresses from the `To:`, `cc:`, and `Bcc:` fields and standardizes these fields using the columns specified by the **-width** flag. The **-format** flag indicates if Internet style is to be used, which serves as the default.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-inplace	Forces annotation to be done in place in order to preserve links to the annotated message.
Message	Specifies a message. If you specify both a message to reply to and a message draft, you must use the **-draftmessge** flag. Use the following to define a message: ***Number*** Number of the message. **cur or . (period)** Current message. The default reply message. **first** First message in a folder. **last** Last message in a folder. **new** New message that is created. The default draft message is **new**. **next** Message following the current message. **prev** Message preceding the current message.
-noannotate	Prevents annotation. This flag is the default.
-nocc *Names*	Allows you to specify the users who will not be listed in the `cc:` field of the reply. You can specify the following for *Names*: **all**, **to**, **cc**, and **me**.
-nodraftfolder	Places the draft in the file *UserMhDirectory***/draft**.
-noedit	Suppresses the initial edit.
-noformat	Suppresses both removal of duplicate addresses from the `To:`, `cc:`, and `Bcc:` fields, and standardization of these fields.
-noinplace	Prevents annotation in place. This flag is the default.
-noquery	Automatically builds the `To:` and `cc:` fields. This flag is the default.

Item	Description
-nowhatnowproc	Prevents interactive processing for the **repl** command. This flag prevents editing.
-query	Queries you for permission to include each address in the `To:` and `cc:` fields.
-whatnowproc *Program*	Starts the specified command string as the program to guide you through the reply tasks. The default is the **whatnow** program.
-width *Number*	Sets the width of the address fields. The default is 72 columns.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Alternate-Mailboxes:`	Specifies the mailboxes.
`Current-Folder:`	Sets the default current folder.
`Draft-Folder:`	Sets the default folder for drafts.
`Editor:`	Sets the default editor.
`fileproc:`	Specifies the program used to refile messages.
`mhlproc:`	Specifies the program used to filter the message for which you are creating a reply.
`Msg-Protect:`	Sets the protection level for the new message files.
`Path:`	Specifies the user's MH directory.
`whatnowproc:`	Specifies the program used to prompt `What now?` questions.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To reply to the current message in the current folder, enter:

   ```
   repl
   ```

   The system responds with text similar to the following:

   ```
   To: patrick@venus
   cc: tom@thomas
   Subject: Re: Meeting on Monday
   In-reply-to: (Your message of Thu, 21 Jul 88 13:39:34 CST.)
                <8807211839.AA01868>
   --------------
   ```

   You can now enter your reply. When you finish entering the body of your reply, press the Ctrl-D key sequence to exit the editor. The system responds with the following:

   ```
   What now?
   ```

Enter send to send the reply. If you want to see a list of subcommands, press the Enter key. In this example, you are sending a reply to the current message in the current folder.

2. To send a reply to message 4 in the inbox folder, enter:

   ```
   repl  +inbox 4
   ```

   The system responds with a message similar to the following:

   ```
   To: dawn@chaucer
   cc: jay@venus
   Subject: Re: Status Report
   In-reply-to: (Your message of Thu, 21 Jul 88 13:39:34 CST.)
                <8807211839.AA01868>
   --------------
   ```

   You can now enter your reply. When you finish entering the body of your reply, press the Ctrl-D key sequence to exit the editor. The system responds with the following:

   ```
   What now?
   ```

   Enter send to send the reply. If you want to see a list of subcommands, press the Enter key.

3. To keep track of your reply to the current message in the current folder, use the **-annotate** flag to place a copy of the date and time in the message you are replying to, as follows:

   ```
   repl  -annotate
   ```

   The system responds with a message similar to the following:

   ```
   To: patrick@venus
   cc: tom@thomas
   Subject: Re: Meeting on Friday
   In-reply-to: (Your message of Mon, 17 Apr 89 13:39:34 CST.)
                <8904171839.AA01868>
   --------------
   ```

   You can now enter your reply. When you finish entering the body of your reply, press the Ctrl-D key sequence to exit the editor. The system responds with the following:

   ```
   What now?
   ```

   Enter send to send the reply. If you quit the editor without sending the reply, the annotation does not occur.

Files

Item	Description
$HOME/.mh_profile	Specifies the user's MH profile.
/etc/mh/replcomps	Contains the MH default reply template.
UserMhDirectory/**replcomps**	Contains the user's default reply form.
/usr/bin/repl	Contains the **repl** command.
UserMhDirectory/**draft**	Contains the current message draft.

replacepv Command

Purpose

Replaces a physical volume in a volume group with another physical volume.

Syntax

replacepv [**-f**] {*SourcePhysicalVolume* | *SourcePhysicalVolumeID* } *DestinationPhysicalVolume*

replacepv [**-R**] *dir_name* [*DestinationPhysicalVolume*]

Description

The **replacepv** command replaces allocated physical partitions and the data they contain from the *SourcePhysicalVolume* to *DestinationPhysicalVolume*. The specified source physical volume cannot be the same as *DestinationPhysicalVolume*.

Note:

1. The *DestinationPhysicalVolume* must not belong to a volume group.
2. The *DestinationPhysicalVolume* size must be at least the size of the *SourcePhysicalVolume*.
3. The **replacepv** command cannot replace a *SourcePhysicalVolume* with stale logical volume unless this logical volume has a non-stale mirror.
4. You cannot use the **replacepv** command on a snapshot volume group or a volume group that has a snapshot volume group.
5. Running this command on a physical volume that has an active firmware assisted dump logical volume temporarily changes the dump device to **/dev/sysdumpnull**. After the migration of logical volume is successful, this command calls the **sysdumpdev -P** command to set the firmware assisted dump logical volume to the original logical volume.
6. The VG corresponding to the SourcePhysicalVolume is examined to determine if a PV type restriction exists. If a restriction exists, the DestinationPhysicalVolume is examined to ensure that it meets the restriction. If it does not meet the PV type restriction, the command will fail.

The allocation of the new physical partitions follows the policies defined for the logical volumes that contain the physical partitions being replaced.

Flags

Item	Description
-f	Forces to replace a *SourcePhysicalVolume* with the specified *DestinationPhysicalVolume* unless the *DestinationPhysicalVolume* is part of another volume group in the Device Configuration Database or a volume group that is active.
-R *dir_name*	Recovers **replacepv** if it is interrupted by <ctrl-c>, a system crash, or a loss of quorum. When using the **-R** flag, you must specify the directory name given during the initial run of **replacepv**. This flag also allows you to change the *DestinationPhysicalVolume.*

Security

Access Control: You must have root authority to run this command.

Examples

1. To replace physical partitions from `hdisk1` to `hdisk6`, enter:

```
replacepv hdisk1 hdisk6
```

Files

Item	Description
/usr/sbin	Directory where the **replacepv** command resides.

Item	Description
/tmp	Directory where the temporary files are stored while the command is running.

repquota Command

Purpose

Summarizes quotas for a file system.

Syntax

repquota [**-v**] [**-c**] [**-g**] [**-u**] [**-l**] { -a | *FileSystem ...* }

Description

The **repquota** command prints a summary of quotas and disk usage for a file system specified by the *FileSystem* parameter. If the **-a** flag is specified instead of a file system, the **repquota** command prints the summary for all file systems enabled with quotas in the **/etc/filesystems** file. By default, both user and group quotas are printed.

For each user or group, the **repquota** command prints:

- Number of existing user or group files
- Amount of disk space being used by the user or group
- User or group quotas

Flags

Item	Description
-a	Specifies that quotas are printed for all file systems enabled with quotas in the **/etc/filesystems** file.
-c	Changes the output of the command to a colon-delineated format.
-g	Specifies that only group quotas are printed.
-l	Enables long user names to be printed on the repquota report. The default behavior of the report will be to truncate the name at 9 characters. If the **-l** option is specified, the full user name will be used.
-u	Specifies that only user quotas are printed.
-v	Prints a header line before the summary of quotas for each file system.

Security

Access Control: Only the root user can execute this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To print a summary of user quotas in the /u file system, enter:

```
repquota -u /u
```

The system prints the following information:

```
                  Block  limits                 File limits
User      used    soft    hard   grace   used   soft  hard  grace
root  --  3920     0       0             734     0     0
davec +-    28     8      30     3 days    3     0     0
keith --    48     0       0               7     0     0
```

The + printed in the first column next to davec indicates that the user has exceeded established block limits. If there were a + in the second column, it would indicate that the user had exceeded established file limits.

Files

Item	Description
quota.user	Specifies user quotas.
quota.group	Specifies group quotas.
/etc/filesystems	Contains file system names and locations.
/etc/group	Contains basic group attributes.
/etc/passwd	Contains user names and locations.

reset Command

Purpose

Initializes terminals.

Syntax

reset [**-e** *C*] [**-k** *C*] [**-i** *C*] [**-**] [**-s**] [**-n**] [**-I**] [**-Q**] [**-m** [*Identifier*] [*TestBaudRate*] **:***Type*] ... [*Type*]

Description

The **reset** command is a link to the **tset** command. If the **tset** command is run as the **reset** command, it performs the following actions before any terminal-dependent processing is done:

- Set Cooked and Echo modes to on
- Turn off cbreak and Raw modes
- Turn on new-line translation
- Restore special characters to a sensible state.

Any special character that is found to be NULL or -1 is reset to its default value. All flags to the **tset** command can be used with the **reset** command.

The **reset** command is most useful when a program dies and leaves a terminal in an undesirable state. The sequence <LF>reset<LF> (where <LF> is Ctrl-J, the line feed) may be required to get the **reset** command to run successfully since carriage-return might not work in this state. The <LF>reset<LF> sequence frequently will not be echoed.

Flags

Item	Description
-	The name of the terminal decided upon is output to standard output. This is intended to be captured by the shell and placed in the **TERM** environment variable.

Item	Description
-e *C*	Set the erase character to the character specified by the C variable on all terminals. The default is the backspace character on the terminal, usually ^ (cedilla). The character C can either be typed directly or entered using the ^ (cedilla).
-I	Suppresses transmission of terminal initialization strings.
-i *C*	Is similar to the **-e** flag, but uses the interrupt character rather than the erase character. The C variable defaults to ^C. The ^ character can also be used for this option.
-k *C*	Is similar to the **-e** flag, except uses the line-kill character rather than the erase character. The C variable defaults to ^X. The kill character is left alone if **-k** is not specified. The ^ character can also be used for this option.
-m*IdentifierTestbaudRate***:***Type*	Specifies which terminal type (in the *Type* parameter) is usually used on the port identified in the *Identifier* parameter. A missing identifier matches all identifiers. You can optionally specify the baud rate in the *TestBaudRate* parameter.
-n	On systems with the Berkeley 4.3 tty driver, specifies that the new tty driver modes should be initialized for this terminal. For a CRT, the CRTERASE and CRTKILL modes are set only if the baud rate is 1200 bps or greater. See the **tty** file for more information.
-Q	Suppresses printing of the `Erase set to` and `Kill set to` messages.
-s	Prints the sequence of **csh** commands that initialize the **TERM** environment variable, based on the name of the terminal decided upon.

Files

Item	Description
/usr/share/lib/terminfo/?/*	Contains the terminal capability database.

resetrsrc Command

Purpose

Resets a resource that is, forces the resource to move to the offline state.

Syntax

To reset one or more resources, using data entered on the command line:

`resetrsrc -s "`*selection_string*`"`` [-N { `*node_file* | `"-" }] [-h] [-TV] `*resource_class* [*arg*=*value...*]

`resetrsrc -r [-h] [-TV] `*resource_handle* [*arg*=*value...*]

To reset one or more resources using command arguments that are predefined in an input file:

`resetrsrc -f `*resource_data_input_file*` -s "`*selection_string*`" [ -N { `*node_file* | `"-" } ] [-h] [-TV] `*resource_class*

`resetrsrc -f `*resource_data_input_file*` -r [-h] [-TV] `*resource_handle*

To display the names and data types of the command arguments:

`resetrsrc -l [-h] `*resource_class*

Description

The `resetrsrc` command requests that the resource monitoring and control (RMC) subsystem force one or more resources offline. The request is performed by the appropriate resource manager.

To reset one or more resources, use the `-s` flag to force offline all of the resources that match the specified selection string. To reset one specific resource, use the `-r` flag to specify the resource handle that represents that specific resource.

Instead of specifying multiple node names in *selection_string*, you can use the `-N` *node_file* flag to indicate that the node names are in a file. Use `-N "-"` to read the node names from standard input.

Use the `-l` flag to determine whether the specified resource class accepts any additional command arguments.

The successful completion of this command does not guarantee that the resource is offline, only that the resource manager successfully received the request to force this resource offline. Monitor the resource dynamic attribute `OpState` to determine when the resource is forced offline. Register an event for the resource, specifying the `OpState` attribute, to know when the resource is offline. Or, intermittently run the `lsrsrc` command until you see that the resource is offline (the value of `OpState` is 2). For example:

```
lsrsrc -s 'Name == "/filesys1"' -t IBM.FileSystem Name OpState
```

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM `nodegrp` command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

Parameters

resource_class
Specifies the name of the resource class that contains the resources that you want to force offline.

resource_handle
Specifies the resource handle that corresponds to the resource you want to force offline. Use the `lsrsrc` command to obtain a list of valid resource handles. The resource handle must be enclosed within double quotation marks, for example:

```
"0x4017 0x0001 0x00000000 0x0069684c 0x0d4715b0 0xe9635f69"
```

arg=value...
Specifies one or more pairs of command argument names and values.

arg
Specifies the argument name.

value
Specifies the value for this argument. The value data type must match the definition of the argument data type.

Command arguments are optional. If any *arg=value* pairs are entered, there must be one *arg=value* pair for each command argument defined for the offline function for the specified resource class.

Use `resetrsrc -l` to get a list of the command argument names and data types for the specific resource class.

Flags

-f *resource_data_input_file*
Specifies the name of the file that contains resource argument information. The following contents of the file is displayed:

```
PersistentResourceArguments::
argument1 = value1
argument2 = value2
```

-l
Lists the command arguments and data types. Some resource managers accept additional arguments that are passed to the offline request. Use this flag to list any defined command arguments and the data types of the command argument values.

-N { *node_file* | "-" }
Specifies that node names are read from a file or from standard input. Use `-N` *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (#) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use `-N "-"` to read the node names from standard input.

The CT_MANAGEMENT_SCOPE environment variable determines the scope of the cluster. If CT_MANAGEMENT_SCOPE is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and CT_MANAGEMENT_SCOPE is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-r
Forces offline the specific resource that matches the specified resource handle.

-s "*selection_string*"
Specifies the selection string. All selection strings must be enclosed within either double or single quotation marks. If the selection string contains double quotation marks, enclose the entire selection string in single quotation marks. For example:

```
-s 'Name == "testing"'
-s 'Name ?= "test"'
```

Only persistent attributes can be listed in a selection string.

-h
Writes the command usage statement to standard output.

-T
Writes the command trace messages to standard error. For your software service organization use only.

-V
Writes the command verbose messages (if there are any available) to standard output.

Environment variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command

is being run. The resource class or resources that are displayed or modified by the command are on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Standard output

When the `-h` flag is specified, this command usage statement is written to standard output. When the `-V` flag is specified, this command verbose messages (if there are any available) are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
No resources were found that match the specified selection string.

Security

You need write permission for the *resource_class* specified in `resetrsrc` to run `resetrsrc`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *Administering RSCT* guide for information about the ACL file and how to modify it.

Implementation specifics

This command is part of the `rsct.core.rmc` fileset for AIX and `rsct.core-3.1.0.0-0.`*platform*`.rpm` package for Linux, Solaris, and Windows, where *platform* is `i386`, `ppc`, `ppc64`, `s390`, or `x86_64`.

Location

/opt/rsct/bin/resetrsrc

Examples

Suppose that you have a peer domain called `foo` with three defined nodes: nodeA, nodeB, and nodeC. nodeA has two Ethernet cards: `ent0` and `ent1`.

1. Suppose nodeA is online and `ent0` (on nodeA) is also online. To force `ent0` offline on nodeA, run this command on nodeA:

   ```
   resetrsrc -s 'Name == "ent0"' IBM.EthernetDevice
   ```

2. Suppose nodeA and nodeB are online, `ent0` (on nodeA) is also online, and you are currently logged on to nodeB. To force `ent0` offline on nodeA, run this command on nodeB:

   ```
   resetrsrc -s 'NodeName == "nodeA" AND Name == "ent0"' IBM.EthernetDevice
   ```

3. Suppose nodeA and nodeB are online and file system `/filesys1` is defined and mounted on nodeB. To force `/filesys1` offline on nodeB, run this command on nodeA:

   ```
   resetrsrc -s 'NodeName == "nodeB" AND Name == "/filesys1"' IBM.FileSystem
   ```

4. Suppose the resource handle for `ent0` on nodeA is:

   ```
   0x406b 0x0001 0x00000000 0x0069564c 0x0dc1f272 0xb9de145e
   ```

   To force `ent0` offline on nodeA, run this command on nodeA:

   ```
   resetrsrc -r "0x406b 0x0001 0x00000000 0x0069564c 0x0dc1f272 0xb9de145e"
   ```

5. To reset `ent0` on nodeA and nodeB, using the `/tmp/common/node_file` file:

   ```
   # common node file
   #
   nodeA
   nodeB
   #
   ```

   as input, enter:

   ```
   resetrsrc -s 'Name == "ent0"' -N /tmp/common/node_file \
   IBM.EthernetDevice
   ```

resize Command

Purpose

Sets the **TERMCAP** environment variable and terminal settings to the current window size.

Syntax

resize [**-c** | **-u**] [**-s** [*Rows Columns*]]

Description

The **resize** command utility prints a shell command for setting the **TERM** and **TERMCAP** environment variables to indicate the current size of the xterm window from which the command is run. For this output to take effect, the **resize** command must either be evaluated as part of the command line (usually done with a shell alias or function) or else be redirected to a file that can then be read in. From the C shell (usually known as **/bin/csh**), the following alias could be defined in the user's **.cshrc** file:

```
%  alias rs 'set noglob; `eval resize`'
```

After resizing the window, the user would enter:

```
%  rs
```

Users of versions of the Bourne shell (usually known as **/bin/sh**) that do not have command functions will need to send the output to a temporary file and then read it back in with the . (dot) command:

```
$  resize >/tmp/out
$  . /tmp/out
```

Flags

Item	Description
-c	Indicates that C shell commands should be generated even if the user's current shell is not **/bin/csh**.
-u	Indicates that Bourne shell commands should be generated even if the user's current shell is not a Bourne shell.
-s [*Rows Columns*]	Indicates that Sun console escape sequences will be used instead of the special xterm escape code. If the *Rows* and *Columns* parameters are given, the **resize** command will ask the xterm window to resize itself. However, the window manager may choose to disallow the change.

Note: The -**c** or -**u** must appear to the left of -**s** if both are specified.

File

Item	Description
/etc/termcap	Provides modification for the base termcap entry.

resource_data_input Information File

Purpose

Describes how to use an input file for passing resource class information, such as resource attribute names and values, to the resource monitoring and control (RMC) command-line interface (CLI).

Description

You can use the `-f` flag with most RMC commands to specify the name of a resource data input file when you want to pass resource persistent attribute values and other information to the RMC CLI. This is useful when typing information about the command line would be too cumbersome or prone to typographical

errors. The data in this file is used for defining resources or for changing the persistent attribute values of a resource or resource class. The resource data input file, which must be in POSIX format, has no set location. It can be a temporary file or a permanent file, depending on your requirements.

The **chrsrc**, **mkrsrc**, **resetrsrc**, **rmrsrc**, **runact**, **startrsrc**, and **stoprsrc** commands read this file when they are issued with the `-f` flag. The **lsactdef**, **lsrsrc**, and **lsrsrcdef** commands generate a file with this format when they are issued with the `-i` flag.

Keywords are used in the input file to indicate which type of data is listed in the related stanza:

ResourceAction
Resource action element names and values for the resource action when starting an action. The **runact** command reads in the resource action elements. These elements are ignored if the input file is read by `runact -c`.

ResourceClassAction
Resource class action element names and values for the resource class action when starting a class action. The **runact** command reads in the resource action elements.

PersistentResourceArguments
Resource command argument names and values for those commands that accept them: **mkrsrc**, **resetrsrc**, **rmrsrc**, **startrsrc**, and **stoprsrc**. Command arguments are optional and are defined by the resource class. Specify the `-l` option with these commands to see the command arguments for a resource class.

PersistentResourceAttributes
Persistent attribute names and values for one or more resources for a specific resource class used to define a new resource or change attribute values for an existing resource. The persistent resource attributes are read in by the commands **mkrsrc** and **chrsrc**. These attributes are ignored if the input file is read by the **chrsrc** command that is specified with the `-c` flag.

PersistentResourceClassAttributes
Persistent attribute names and values for a resource class used to change the attribute values of an existing resource class. The persistent resource class attributes are read in by the **chrsrc** command only when the `-c` flag is specified.

In general, a *resource_data_input* file is a flat text file with the following format. **Bold** words are literal. Text that precedes a single colon (:) is an arbitrary label and can be any alphanumeric text.

```
PersistentResourceAttributes::
# This is a comment
    label:
      AttrName1  = value
      AttrName2  = value
      AttrName3  = value
    another label:
        Name        = name
        NodeNumber  = 1
    ⋮
    ::

PersistentResourceClassAttributes::
# This is a comment
    label:
      SomeSettableAttrName         = value
      SomeOtherSettableAttrName     = value
    ::
⋮

PersistentResourceArguments::
# This is a comment
    label:
      ArgName1  = value
      ArgName2  = value
      ArgName3  = value
    ::
⋮
```

See the Examples section for more details.

Some notes about formatting follow:

- The keywords PersistentResourceAttributes, PersistentResourceClassAttributes, and PersistentResourceArguments are followed by two colons (::).
- The order of the keyword stanzas is not significant in the file. For example, PersistentResourceClassAttributes can precede PersistentResourceClass. It does not affect the portion of the data that is read in by the calling CLI.
- Individual stanza headings (beneath the keywords) are followed by one colon (:), for example: `c175n05 resource info:`.
- White space at the beginning of lines is not significant. Tabs or spaces are suggested for readability.
- Any line with a pound sign (#) as the first printable character is a comment.
- Each entry on an individual line is separated by white space (spaces or tabs).
- Blank lines in the file are not significant and are suggested for readability.
- There is no limit to the number of resource attribute stanzas included in a particular PersistentResourceAttributes section.
- There is no limit to the number of resource class attribute stanzas included in a particular PersistentResourceClassAttributes section. Typically, there is only one instance of a resource class. In this case, only one stanza is expected.
- If only one resource attribute stanza is included in a particular PersistentResourceAttributes section, the *label:* line can be omitted. This also applies to the ResourceAction section.
- If only one resource class attribute stanza is included in a particular PersistentResourceClassAttributes section, the *label:* line can be omitted. This also applies to the ResourceClassAction section.
- Values that contain spaces must be enclosed in quotation marks.
- A double colon (::) indicates the end of a section. If a terminating double colon is not found, the next Reserved Keyword or end of file signals the end of a section.
- Double quotation marks included within a string that is surrounded by double quotation marks must be escaped. (\").

  Note: Double quotation marks can be nested within single quotation marks.

  Examples:

  - `"Name == \"testing\""`
  - `'Name == "testing"'`

    This syntax is preferred if your string is a selection string and you are going to cut and paste to the command line.
- Single quotation marks included within a string that is surrounded by single quotation marks must be escaped. (\').

  Note: Single quotation marks can be nested within double quotation marks.

  Here are some examples:

  - `'Isn\'t that true'`
  - `"Isn't that true"`

    This syntax is preferred if you are going to cut and paste to the command line.
- The format you use to enter data in a *resource_data_input* file might not be the same format used on the command line. The shell you choose to run the commands in has its own rules regarding quotation marks. Refer to the documentation for your shell for these rules, which determine how to enter data on the command line.

Implementation specifics

This information is part of the `rsct.core.rmc` fileset for AIX and `rsct.core-3.1.0.0-0.`*platform*`.rpm` package for Linux, Solaris, and Windows, where *platform* is i386, ppc, ppc64, s390, or x86_64.

Location

/opt/rsct/man/resource_data_input.7

Examples

1. This sample **mkrsrc** command:

```
mkrsrc -f /tmp/my_resource_data_input_file IBM.Example
```

uses the sample input file /tmp/my_resource_data_input_file for the IBM.Example resource class. The contents of the input file look like this:

```
PersistentResourceAttributes::
# Resource 1 - only set required attributes
resource 1:
    Name="c175n04"
    NodeList = {1}
# Resource 2 - setting both required and optional attributes
# mkrsrc -e2 IBM.Example displays required and optional
# persistent attributes
resource 2:
    Name="c175n05"
    NodeList = {1}
    Int32 = -99
    Uint32 = 99
    Int64 = -123456789123456789
    Uint64 = 123456789123456789
    Float32 = -9.89
    Float64 = 123456789.123456789
    String = "testing 123"
    Binary = 0xaabbccddeeff
    RH = "0x0000 0x0000 0x00000000 0x00000000 0x00000000 0x00000000"
    SD = [hello,1,{2,4,6,8}]
    Int32Array = {-4, -3, -2, -1, 0, 1, 2, 3, 4}
    Int64Array = {-4,-3,-2,-1,0,1,2,3,4}
    Uint32Array = {0,1,2,3,4,5,6}
    Uint64Array = {0,1,2,3,4,5,6}
    Float32Array = {-3.3, -2.2, -1.2, 0, 1, 2.2, 3.3}
    Float64Array = {-3.3, -2.2, -1.2, 0, 1, 2.2, 3.3}
    StringArray = {abc,"do re mi", 123}
    BinaryArray = {"0x01", "0x02", "0x0304"}
    RHArray     = {"0x0000 0x0000 0x00000000 0x00000000 0x00000000 0x00000000",
                   "0xaaaa 0xaaaa 0xbbbbbbbb 0xcccccccc 0xdddddddd 0xeeeeeeee"}
    SDArray     = {[hello,1,{0,1,2,3}],[hello2,2,{2,4,6,8}]}
```

2. This sample **chrsrc** command:

```
chrsrc -f  /tmp/Example/ch_resources -s 'Name == "c175n05"' IBM.Example
```

uses the sample input file /tmp/Example/ch_resources to change the attribute values of existing IBM.Example resources. The contents of the input file look like this:

```
PersistentResourceAttributes::
# Changing resources that match the selection string entered
# when running chrsrc command.
    resource 1:
        String            = "this is a string test"
        Int32Array        = {10,-20,30,-40,50,-60}
```

3. This sample **rmrsrc** command:

```
rmrsrc -l IBM.Examplebar
```

shows the optional command arguments:

```
rmrsrc IBM.Examplebar ExampleInt32=int32 ExampleUint32=uint32
```

4. This sample **rmrsrc** command:

```
rmrsrc -f /tmp/Examplebar/rm_resources -s 'Name == "c175n05"' IBM.Examplebar
```

uses the sample input `/tmp/Examplebar/rm_resources` file to specify the optional command arguments for **rmrsrc** command. The contents of the input file look like this:

```
PersistentResourceArguments::
# Specifying command arguments when running rmrsrc command.
resource 1:
    ExampleInt32      =   1
        ExampleUint32     =   0
```

restart-secldapclntd Command

Purpose

The **restart-secldapclntd** script is used to stop the currently running **secldapclntd** daemon process and then restart it.

Syntax

/usr/sbin/restart-secldapclntd [**-C** *CacheSize*] [**-p** *NumOfThread*] [**-t** *CacheTimeOut*] [**-T** *HeartBeatIntv*] [**-o** *ldapTimeOut*]

Description

The **restart-secldapclntd** script stops the **secldapclntd** daemon if it is running, and then restarts it. If the **secldapclntd** daemon is not running, it simply starts it.

Flags

By default, the **secldapclntd** daemon reads the configuration information specified in the **/etc/security/ldap/ldap.cfg** file at startup. If the following options are given in command line when starting **secldapclntd** process, the options from the command line will overwrite the values in the **/etc/security/ldap/ldap.cfg** file.

Item	Description
-C *CacheSize*	Sets the maximum cache entries used by the **secldapclntd** daemon to CacheSize number of entries. Valid range is 100-10,000 entries for user cache. The default is 1000. The group cache entries will be 10% of the user cache entries.
-o *ldapTimeOut*	Timeout period in seconds for LDAP client requests to the server. This value determines how long the client will wait for a response from the LDAP server. Valid range is 0 - 3600 (1 hour). Default is 60 seconds. Set this value to 0 to disable the timeout and force the client to wait indefinitely.
-p *NumOfThread*	Sets the number of thread used by the **secldapclntd** daemon to **NumOfThread** threads. Valid range is 1-1000. The default is 10.
-t *CacheTimeout*	Sets the cache to expire in CacheTimeout seconds. Valid range is 60- 3600 seconds. The default is 300 seconds.
-T *HeartBeatIntv*	Sets the time interval of heartbeat between this client and the LDAP server. Valid values are 60-3,600 seconds. Default is 300.

Security

A user with the **aix.security.ldap** authorization is authorized to use this command.

Examples

1. To restart the **secldapclntd** daemon, type:

```
/usr/sbin/restart-secldapclntd
```

2. To restart the **secldapclntd** with using 30 threads and cache timeout value of 500 seconds, type:

```
/usr/sbin/restart-secldapclntd -p 30 -t 500
```

Files

Item	Description
/etc/security/ldap/ldap.cfg	Contains information needed by the **secldapclntd** daemon to connect to the server.

restbase Command

Purpose

Reads the base-customized information from the boot image, and restores it into the Device Configuration database used during system boot phase 1.

Syntax

restbase [**-o** *File*] [**-d** *Path*] [**-v**]

Description

The **restbase** command reads the base-customized information from the boot disk and puts it in the specified Device Configuration database directory. By default, the base information is read from the boot disk. If no Device Configuration database directory is specified, then the **restbase** command restores this information into the **/etc/objrepos** directory. You can use the **-o** flag to specify a file, other than the boot disk, from which to read the base-customized information.

> **Attention:** The **restbase** command is intended to be executed only during phase 1 of system boot. Do not execute it in a run-time environment or you could destroy the Device Configuration database.

Flags

Item	Description
-o *File*	Specifies a file that contains base-customized data.
-d *Path*	Specifies a directory containing the base Device Configuration database.
-v	Causes verbose output to be written to standard output.

Examples

1. To restore base-customized information and see verbose output, enter:

```
restbase -v
```

2. To restore base information into an alternate device database, enter:

```
restbase -d /tmp/objrepos
```

Files

Item	Description
/usr/lib/objrepos/PdDv	Contains entries for all known device types supported by the system.
/etc/objrepos/CuDv	Contains entries for all device instances defined in the system.
/etc/objrepos/CuAt	Contains customized device-specific attribute information.
/etc/objrepos/CuDep	Describes device instances that depend on other device instances.
/etc/objrepos/CuDvDr	Stores information about critical resources that need concurrency management through the use of the Device Configuration Library routines.

restore Command

Purpose

Extracts files from archives that are created with the **backup** command.

Syntax

To restore files archived by file name

restore -x [d M n O Q v q e] [-b *Number*] [-L *Label*] [-I *Label*] [-f *Device*] [-s *SeekBackup*] [-E { force | ignore | warn }] [*File* ...]

To list files archived by file name

restore -T | -t [a l n q v Q] [-b*Number*] [-f*Device*] [-s*SeekBackup*]

To restore files archived by file system

restore -r [B O n q v y] [-b*Number*] [-f*Device*] [-s*SeekBackup*]

To restore files archived by file system

restore -R [B O n v y] [-b *Number*] [-f *Device*] [-s *SeekBackup*]

To restore files archived by file system

restore -i [O h m n q v y] [-b*Number*] [-f *Device*] [-s *SeekBackup*]

To restore files archived by file system

restore -x [B O h n m q v y] [-b*Number*] [-f*Device*] [-s*SeekBackup*] [*File* ...]

To restore files beginning at a specified volume number

restore -X *Number* [-MdnqveOQ] [-b *Number*] [-f *Device*] [-s *Number*] [-E { force | ignore | warn }] [*File* ...]

To list files archived by file system

restore -t | -T [B a l n h q v y] [-b *Number*] [-f *Device*] [-s *SeekBackup*] [*File* ...]

To restore file attributes archived by file name

restore -Pstring [B d qv Q] [b*Number*] [s *SeekNumber*] [-L *Label*] [-I *Label*] [-f *Device*] [*File* ...]

To restore file attributes archived by file system

restore -Pstring [hqv] [b *Number*] [s *SeekNumber*] [-f *Device*] [*File* ...]

Description

The **restore** command reads archives created by the **backup** command and extracts the files that are stored on them. These archives can be in either file name or file system format. An archive can be stored on disk, diskette, or tape. Files must be restored by using the same method that was used to archive the files. This operation requires that you know the format of the archive. The archive format can be determined by examining the archive volume header information that is displayed when you use the `-T` flag. When the `-x`, `-r`, `-T`, or `-t` flags are used, the **restore** command automatically determines the archive format.

Individual files can be restored from either file name or file system archives by using the `-x` flag and specifying the file name. The file name must be specified as it exists on the archive. Files can be restored interactively from file system archives by using the `-i` flag. The names of the files on an archive can be written to standard output by using the `-T` flag.

Users must have write access to the file system device or have Restore authorization to extract the contents of the archive.

The diskette device, `/dev/rfd0`, is the default media for the **restore** command. To restore from standard input, specify a `-` (dash) with the `-f` flag. You can also specify a range of devices, such as `/dev/rfd0`.

Notes:

1. If you are restoring from a multiple-volume archive, the **restore** command reads the volume that mounted, prompts you for the next volume, and waits for your response. After the next volume is inserted, press the Enter key to continue restoring files.
2. If an archive, created by using the **backup** command, is made to a tape device with the device block size set to 0, it is necessary for you to have explicit knowledge of the block size that was used when the tape was created to restore from the tape.
3. Multiple archives can exist on a single tape. When multiple archives are restored from the tape, the **restore** command expects the input device to be a no-retension-on-open, no-rewind-on-close tape device. Do not use a no-rewind tape device for restoring unless either the `-B`, `-s`, or `-X` flag is specified. For more information on using tape devices, see the **rmt** special file.

File system archives

File system archives are also known as i-node archives because the method used to archive the files. A file system name is specified with the **backup** command, and the files within that file system are archived based on their structure and layout within the file system. The **restore** command restores the files on a file system archive without any special understanding of the underlying structure of the file system.

When you restore the file system archives, the **restore** command creates and uses a file named `restoresymtable`. This file is created in the current directory. The file is necessary for the **restore** command to do incremental file system restores.

Note: Do not remove the `restoresymtable` file if you run incremental file system backups and restores.

The *File* parameter is ignored when you use either the `-r` or the `-R` flag.

File name Archives

File name archives are created by specifying a list of file names to archive to the **backup** command. The **restore** command restores the files from a file name archive without any special understanding of the underlying structure of the file system. The **restore** command allows for metacharacter to be used when you specify files for archive extraction. This process provides the capability to extract files from an archive that is based on pattern matching. A pattern file name must be enclosed in single quotations, and patterns must be enclosed in brackets (...).

About sparse files

Files in the operating system file system that contain long strings of Nulls can be stored efficiently when compared to the other files. If a string of Nulls spans an entire allocation block, that whole block is not stored on disk at all. Files where one or more blocks are omitted in this way are called sparse files. The missing blocks are also known as holes.

Note: Restores the non-sparse files as nonsparse because they were archived by the name format of the **backup** command for both packed and unpacked files. It is necessary to know the sparseness and nonsparseness of the file being restored before you archive the files. This check is required because by enabling the `-e` flag, the flag restores the sparse files as nonsparse. This flag must be enabled only if the files to be restored are non-sparse consisting of more than 4 KB Nulls. If the `-e` flag is specified during the restore operation, it successfully restores all normal files normally and nonsparse database files as nonsparse.

Flags

Item	Descriptor
`-a`	Specified with the `t` and T option, the `-a` option displays the list of files in the archive, along with their permissions.
`-B`	Specifies that the archive must be read from standard input. Normally, the **restore** command examines the actual medium to determine the backup format. When you use a | (pipe), this examination cannot occur. As a result, the archive is assumed to be in file system format, and the device is assumed to be standard input (`-f`).
`-b`*Number*	Specifies the number of 512-byte blocks for backups done by name. For backups that are done by i-node, the flag specifies the number of 1024-byte blocks to read in a single output. When the **restore** command reads from tape devices, the default is 100 for backups by name and 32 for backups by i-node. The read size is the number of blocks that are multiplied by the block size. The default read size for the **restore** command reading from tape devices is 51200 (100 * 512) for backups by name and 32768 (32 * 1024) for backups by i-node. The read size must be an even multiple of the tapes physical block size. If the read size is not an even multiple of the tapes physical block size and it is in fixed block mode (nonzero), the **restore** command tries to determine a valid value for *Number*. If successful, the **restore** command changes *Number* to the new value, write a message about the change to standard output, and continues. If unsuccessful in finding a valid value for *Number*, the **restore** command writes an error message to standard error and exits with a nonzero return code. Larger values for the *Number* parameter result in larger physical transfers from the tape device. The value of the `-b` flag is always ignored when the **restore** command reads from diskette. In this case, the command always reads in clusters that occupy a complete track.
`-d`	Indicates that, if the *File* parameter is a directory, all files in that directory must be restored. This flag can be used when the archive is in file name format.

Item	Descriptor
`-e`	Specifies to not restore sparse files actively. If a file has a block that is aligned and sized areas that are Null populated, then the restore operation creates physical space for those file system blocks to be allocated and filled with Nulls. The file size that is specified in bytes corresponds to the space taken within the file system. This flag must be enabled only if files are to be restored are nonsparse consisting of more than 4 KB Nulls. If the `-e` flag is specified during **restore**, it successfully restores all normal files normally and nonsparse database files as nonsparse.
`-E`	The `-E` option extracts beginning at a specified volume number and requires one of the following arguments. If you omit the `-E` option, `warn` is the default behavior. **force** Fails the restore operation on a file if the fixed extent size or space reservation of the file cannot be preserved. **ignore** Ignores any errors in preserving extent attributes. **warn** Issues a warning if the space reservation or the fixed size of the file cannot be preserved.
`-f`*Device*	Specifies the input device. To receive input from a named device, specify the *Device* variable as a path name such as `/dev/rmt0`. To receive input from the standard output device, specify a - (minus sign). The - (minus) feature allows to pipe the input of the**restore** command from the **dd** command. You can also specify a range of archive devices. The range specification must be in the following format: `/dev/deviceXXX-YYY` where *XXX* and *YYY* are whole numbers, and *XXX* must always be less than *YYY*; for example, `/dev/rfd0-3`. All devices in the specified range must be of the same type. For example, you can use a set of 8 mm, 2.3GB tapes or a set of 1.44MB diskettes. All tape devices must be set to the same physical tape block size. If the *Device* variable specifies a range, the**restore** command automatically goes from one device in the range to the next. After all the specified devices are exhausted , the**restore** command halts and requests that new volumes be mounted on the range of devices.
`-h`	Restores only the actual directory, not the files that are contained in it. This flag can be used when the archive is in file system format. This flag is ignored when used with the `-r` or `-R` flags.
`-I` *Label*	The **restore** command applies this integrity label for files without security labels in the archive. The label that is supplied must exist on the system. This option is valid only for restoring files by name on Trusted AIX.

Item	Descriptor
`-i`	Restores the selected files interactively from a file system archive. The following are the subcommand for the `-i` flag: **cd*Directory*** Changes the current directory to the specified directory. **add [*File*]** Specifies that the *File* parameter is added to the list of files to extract. If *File* is a directory, that directory and all the files that are contained in it are added to the extraction list (unless the `-h` flag is used). If *File* is not specified, the current directory is added to the extraction list. **delete [*File*]** Specifies that the *File* parameter is to be removed from the list of files to be extracted. If *File* is a directory, that directory and all the files that are contained in it are removed from the extraction list (unless the `-h` flag is used). **ls [*Directory*]** Displays the directories and files that are contained within the *Directory* parameter. Directory names are displayed with a / (slash) after the name. Files and directories, within the specified directory, that are on the extraction list are displayed with an * (asterisk) before the name. If verbose mode is on, the i-node number of the files and directories is also displayed. If the *Directory* parameter is not specified, the current directory is used. **extract** Restores all the directories and files on the extraction list. **pwd** Displays the full path name of the current directory. **verbose** Causes the **`ls`** subcommand to display the i-node number of files and directories. More information about each file is also displayed as it is extracted from the archive. **setmodes** Sets the owner, mode, and time for all directories added to the extraction list. **quit** Causes **`restore`** to exit immediately. Any files on the extraction list are not restored. **help** Displays a summary of the subcommand.
`-l`	Specified with the `-t` and `-T` option. When specified, displays a detailed list of files, which includes the timestamp, file permissions, file size, owner, and group. The `-l` option overrides the `-a` option.
`-L`*Label*	The **`restore`** command applies this sensitivity label for files without security labels in the archive. The label that is supplied must exist on the system. This option is valid only for restoring files by name on Trusted AIX.

Item	Descriptor
-M	Sets the access and modification times of restored files to the time of restoration. If a restored file is an archive that is created by the **ar** command, the modification times in all the member headers are also set to the time of restoration. You can specify the -M flag only when you are restoring individually named files and only if the -x or -X flags are also specified. When the -M flag is not specified, the **restore** command maintains the access and modification times as displayed on the backup medium. The-M flag is used when the data is in the AIX 4.2 backup by-i-node or by-name format.
-m	Renames restored files to the file's i-node number as it exists on the archive. This function is useful if a few files are being restored and you want these files that are restored under a different file name. Since any restored archive members are renamed to their i-node numbers, directory hierarchies and links are not preserved. Directories and hard links are restored as regular files. The -m flag is used when the archive is in file system format.
-n	By default the restore command restores any ACLs, PCLs, or named extended attributes in the archive. The -n flag causes the **restore** command to ignore any ACLs, PCLs, or named extended attributes in the archive and not restore them.When the archived files contain Encrypted file system (EFS) information, the EFS extended attributes are restored even if the -n flag is specified. On Trusted AIX systems, the -n option causes the **restore** command to ignore Trusted AIX security attributes. For more information about EFS restoration, see Backup and restore in *Security*.
-O	Causes the **restore** command to ignore Trusted AIX security attributes.
-Pstring	Restore only the file attributes. Does not restore the file contents. If the file specified does not exist in the target directory path, the files are not created. This flag restores file attributes selectively depending on the flags that are specified in the string parameter. String parameter can be a combination of the following characters: **A** restore all attributes. **a** restore only the permissions of the files. **o** restore only the ownership of the files. **t** restore only the timestamp of the files. **c** restore only the ACL attributes of the files. **Note:** Among the existing options for the **restore** command, options v, h, b, s, f, B, d, and q are valid with the P option. The P option can be used with both file name and file system archives. If the File argument is a symbolic link, then the metadata of the target file is modified and not that of the symbolic link. **Warning:** Usage of the -P flag overwrites the attributes of files that are owned by another user when run by the superuser.

Item	Descriptor
`-Q`	Specifies that the command must exit when an error is encountered, for backups done by name. This process does not attempt to recover and continue processing the archive, when an error occurs.
`-q`	Specifies that the first volume is ready to use and that the **restore** command cannot prompt you to `mount the volume and hit Enter`. If the archive spans multiple volumes, the **restore** command prompts you for the subsequent volumes.
`-r`	Restores all files in a file system archive. The `-r` flag is only used to restore complete level 0 backups or to restore incremental backups after a level 0 backup is restored. The `restoresymtable` file is used by **restore** to pass information between incremental restores. This file must be removed when the last incremental backup is restored. The *File* parameter is ignored when use the `-r` flag.
`-R`	Requests a specific volume of a multiple-volume, file system archive. The `-R` flag allows a previously interrupted restore to be restarted. The *File* parameter is ignored when you use the `-R` flag. When the **restore** command is restarted, it functions similar to the `-r` flag.
`-s`*SeekBackup*	Specifies the backup to seek and restore on a multiple-backup tape archive. The `-s` flag is only applicable when the archive is written to a tape device. To use the `-s` flag properly, a no-rewind-on-close and no-retension-on-open tape device, such as `/dev/rmt0.1` or `/dev/rmt0.5`, must be specified. If the `-s` flag is specified with a rewind tape device, the **restore** command displays an error message and exits with a nonzero return code. If a no-rewind tape device is used and the `-s` flag is not specified, a default value of `-s1` is used. The value of the *SeekBackup* parameter must be in the range of 1 to 100 inclusive. It is necessary to use a no-rewind-on-close, no-retension-on-open tape device because of the behavior of the `-s` flag. The value that is specified with `-s` is relative to the position of the tapes read/write head and not to an archives position on the tape. For example, to restore the first, second, and fourth backups from a multiple-backup tape archive, the respective values for the `-s` flag would be `-s1`, and `-s2`.
`-t`	Displays information about the backup archive. If the archive is in file system format, a list of files that are found on the archive is written to standard output. The name of each file is preceded by the i-node number of the file as it exists on the archive. The file names that are displayed are relative to the root (/) directory of the file system that was backed up. If the *File* parameter is not specified, all the files on the archive are listed. If the *File* parameter is used, then just that file is listed. If the *File* parameter refers to a directory, all the files that are contained in that directory are listed. If the archive is in file name format, information that is contained in the volume header is written to standard error. This flag can be used to determine whether the archive is in the file name or the file system format.
`-T`	Displays information about the backup archive. If the archive is in file name format, the information that is contained in the volume header is written to standard error, and a list of files that are found on the archive is written to standard output. The *File* parameter is ignored for file name archives. If the archive is in file system format, the behavior is identical to the `-t` flag.

Item	Descriptor
-v	Displays information when the file name is restored . If the archive is in file name format and either the -x or -T flag is specified, the size of the file as it exists on the archive is displayed in bytes. Directory, block, or character device files are archived with a size of 0. Symbolic links are listed with the size of the symbolic link. Hard links are listed with the size of the file, which is how they are archived. Once the archive is read, a total of these sizes is displayed. If the archive is in file system format, directory and nondirectory archive members are distinguished.
-x	Restores individually named files that are specified by the *File* parameter. If the *File* parameter is not specified, all the archive members are restored. If the *File* parameter is a directory and the archive is in file name format, only the directory is restored. If the *File* parameter is a directory and the archive is in file system format, all the files that are contained in the directory are restored. The file names that are specified by the *File* parameter must be the same as the names shown by the **restore**-T command. Files are restored with the same name they were archived with. If the file name was archived by using a relative path name (`./filename`), the file is restored relative to the current directory. If the archive is in file system format, files are restored relative to the current directory. The **restore** command automatically creates any needed directories. When you use this flag to restore file system backups, you are prompted to enter the beginning volume number. The **restore** command allows for shell-style pattern matching metacharacters to be used when files for archive extraction is specified . The rules for matching metacharacters are the same as used in shell pathname "globbing," namely: *** (asterisk)** Matches zero or more characters, but not a '.' (period) or '/' (slash). **? (question mark)** Matches any single character, but not a '.' (period) or '/' (slash). **[ ] (brackets)** Matches any one of the characters that are enclosed within the brackets. If a pair of characters that are separated by a dash is contained within the brackets, the pattern matches any character that lexically falls between the two characters in the current local. Additionally, a '.' (period) or a '/' (slash) within the brackets does not match a '.' (period) or a '/' (slash) in a file name. **\ (backslash)** Matches the immediately following character, preventing its possible interpretation as a metacharacter.
-X*VolumeNumber*	Begins restoring from the specified volume of a multiple-volume, file name backup. When the **restore** command is started, the command behaves similar to the -x flag. The -X flag applies to file name archives only.
-y	Continues restoring when tape errors are encountered. Normally, the **restore** command request input to continue. In either case, all data in the read buffer is replaced with zeros. The -y flag applies only when the archive is in file system format.
-?	Displays a usage message.

Exit Status

This command returns the following exit values:

Item	Descriptor
0	Successful completion.
>0	An error occurred.

Security

On Trusted AIX systems, only authorized users can run the **restore** command.

Item	Descriptor
aix.fs.manage.restore	Required to run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list the names of files in either a file name or file system archive on the diskette device /dev/rfd0, type:

   ```
   restore -Tq
   ```

   The archive is read from the /dev/rfd0 default restore device. The names of all the files and directories that are contained in the archive are displayed. For file system archives, the file name is preceded by the i-node number of the file as it exists on the archive. The -q flag tells the **restore** command that the first volume is available and is ready to be read. As a result, you are not prompted to mount the first volume.

2. To restore a specific file, type:

   ```
   restore -xvqf myhome.bkup system.data
   ```

   This command extracts the file system.data into the current directory from the archive myhome.bkup. The archive in this example is in the current directory. File and directory names must be specified as they are displayed when the -T flag is used. The -v flag displays information during the extraction. This example applies to both file name and file system archives.

3. To restore a specific directory and the contents of that directory from a file name archive, type:

   ```
   restore -xdvqf /dev/rmt0 /home/mike/tools
   ```

   The -x flag tells **restore** to extract files by their file name. The -d tells **restore** to extract all the files and subdirectories in the /home/mike/tools directory. File and directory names must be specified as they are displayed when you use the -T flag. If the directories do not exist, they are created.

4. To restore a specific directory and the contents of that directory from a file system archive, type:

   ```
   restore -xvqf /dev/rmt0 /home/mike/tools
   ```

   This command extracts files by file name. File and directory names must be specified as they are displayed when you use the -T flag. If the directories do not exist, they are created.

5. To restore an entire file system archive, type:

   ```
   restore -rvqf /dev/rmt0
   ```

This command restores the entire file system that is archived on the tape device, `/dev/rmt0`, into the current directory. This example assumes you are in the root directory of the file system to be restored. If the archive is part of a set of incremental file system archives, the archives must be restored in increasing backup-level order beginning with level 0 (for example, `0`, `1`, and `2`).

6. To restore the fifth and ninth backups from a single-volume, multiple-backup tape, type:

```
restore -xvqs 5 -f/dev/rmt0.1
restore -xvqs 4 -f/dev/rmt0.1
```

The first command extracts all files from the fifth archive on the multiple-backup tape that is specified by `/dev/rmt0.1`. The `.1` designator specifies the tape device that is not retensioned when it is opened and rewound when it is closed. It is necessary to use a no-rewind-on-close, no-retension-on-open tape device because of the behavior of the `-s` flag. The second command extracts all the files from the fourth archive (relative to the current location of the tape head on the tape). After the fifth archive is restored, the tape read/write head is in a position to read the archive. To extract the ninth archive on the tape, you must specify a value of 4 with the `-s` flag. This is because the `-s` flag is relative to your position on the tape and not to an archives position on the tape. The ninth archive is the fourth archive from your current position on the tape.

7. To restore the fourth backup, which begins on the sixth tape on a 10-tape multiple-backup archive, put the sixth tape into the tape drive and type:

```
restore -xcs 2 -f /dev/rmt0.1 /home/mike/manual/chap3
```

Assuming the fourth backup is the second backup on the sixth tape, specifying `-s 2` advances the tape head to the beginning of the second backup on this tape. The **restore** command then restores the specified file from the archive. If the backup continues onto subsequent volumes and the file is not restored, the **restore** command instructs you to insert the next volume until the end of the backup is reached. The `-f` flag specifies the no-rewind, no-retension tape device name.

Note: The `-s` flag specifies the backup number relative to the tape inserted in the tape drive, not to the overall 10-tape archive.

8. To improve the performance on streaming tape devices, pipe the dd command to the **restore** command by typing:

```
dd if=/dev/rmt0 bs=64b | restore -xf- -b64
```

The dd command reads the archive from the tape by using a block size of 64 512-byte blocks and writes the archive to standard output. The **restore** command reads the standard input by using a block size of 64 512-byte blocks. The value of the block size that is used by the dd command to read the archive from the tape must be an even multiple of the block size that was used to create the tape with the **backup** command. For example, the following **backup** command cannot be used to create the archive that this example extracts:

```
find /home -print | backup -ivqf/dev/rmt0 -b64
```

This example applies to archives in file name format only. If the archive was in file system format, the **restore** command must include the `-B` flag.

9. To improve the performance of the **restore** command on the 9348 Magnetic Tape Unit Model 12, you can change the block size by typing:

```
chdev -l DeviceName -a BlockSize=32k
```

10. To restore non-sparse database files, type:

```
restore  -xef  /dev/rmt0
```

11. To restore files that were sparse before archive as sparse, type:

```
restore  -xf  /dev/rmt0
```

12. To restore only the permissions of the files from the archive, type:

```
restore -Pa -vf /dev/rmt0
```

13. To restore only the ACL attributes of the files from the archive, type:

```
 restore -Pc -vf /dev/rmt0
```

14. To view the table of contents along with the file permissions, type:

```
restore -Ta -vf /dev/rmt0
```

15. To view the table of contents of file name archive along with the timestamps and file permissions, type:

```
restore -Tl -vf /dev/rmt0
```

16. To view the table of contents of file system archive along with the timestamps and file permissions, type:

```
restore -tl -vf /dev/rmt0
```

Files

Item	Descriptor
/dev/rfd0	Specifies the default restore device.
/usr/sbin/restore	Contains the **restore** command.

restorevgfiles Command

Purpose

Restores files from a backup source.

Syntax

restorevgfiles [**-b** *blocks*] [**-f** *device*] [**-a**] [**-n**] [**-s**] [**-d** *path*] [**-D**] [*file_list*]

Description

The **restorevgfiles** command restores files from tape, file, CD-ROM, or their volume group backup source. The **restorevgfiles** command also works for multi-volume backups such as multiple CDs, DVDs, USB disks, or tapes.

The **restorevgfiles** and **listvgbackup -r** commands perform identical operations and should be considered interchangeable. The **restorevgfiles** command automatically applies the **-r** flag. The **-r** flag, while redundant, is retained for compatibility purposes and will cause no unusual behavior if specified. For a complete description of the **-r** flag, see the **listvgbackup** command.

Flags

Item	Description
-b *blocks*	Specifies the number of 512-byte blocks to read in a single input operation, as defined by the *blocks* parameter. If the *blocks* parameter is not specified, the number of blocks read will default to 100.
-f *device*	Specifies the type of device containing the backup (file, tape, CD-ROM, or other source) as defined by the *device* parameter. When **-f** is not specified, *device* will default to **/dev/rmt0**.

Item	Description
-a	Verifies the physical block size of the tape backup, as specified by the **-b** *block* flag. You may need to alter the block size if necessary to read the backup. The **-a** flag is valid only when a tape backup is used.
-n	Does not restore ACLs, PCLs, or extended attributes.
-s	Specifies that the backup source is a user volume group and not **rootvg**.
-d *path*	Specifies the directory path to which the files will be restored, as defined by the *path* parameter. If the **-d** parameter is not used, the current working directory is used. This can be a problem if the current working directory is root. We recommend writing to a temporary folder instead of to root.
-D	Produces debug output.

Parameters

Item	Description
file_list	Identifies the list of files to be restored. The full path of the files relative to the current directory should be specified in the space-separated list. All files in the specified directory will be restored unless otherwise directed. If you are restoring all files in a directory, we recommend writing to a temporary folder instead of to root.

Examples

1. To read the backup stored at **/dev/cd1** and restore all files to the **/data/myfiles** directory, enter:

   ```
   restorevgfiles -f /dev/cd1 -s -d /data/myfiles
   ```

2. To read the user vg backup from the default device at 20 512-byte blocks at a time and restore the **/myapp/app.h** file to the current directory, enter:

   ```
   restorevgfiles -b 20 -s ./myapp/app.h
   ```

3. To read the backup stored at **/dev/cd1** and restore the **/myapp/app.c** file to the **/data/testcode** directory, enter:

   ```
   restorevgfiles -f /dev/cd1 -s -d /data/testcode ./myapp/app.c
   ```

4. To read the backup stored at **/dev/usbms0** and restore all files to the **/data/myfiles** directory, enter the following command:

   ```
   restorevgfiles –f /dev/usbms0 –s –d /data/myfiles
   ```

Files

Item	Description
/usr/bin/restorevgfiles	Contains the **restorevgfiles** command

restvg Command

Purpose

Restores the user volume group and all its containers and files.

Syntax

restvg [**-b** *Blocks*] [**-d** *FileName*][**-f** *Device*] [**-l**] [**-q**] [**-r**] [**-s**] [**-n**] [**-P** *PPsize*] [*DiskName ...*]

Description

The **restvg** command restores the user volume group and all its containers and files, as specified in the **/tmp/vgdata/***vgname***/***vgname***.data** file (where *vgname* is the name of the volume group) contained within the backup image created by the **savevg** command.

The **restvg** command restores a user volume group. The **bosinstall** routine reinstalls the root volume group (**rootvg**). If the **restvg** command encounters a **rootvg** volume group in the backup image, the **restvg** command exits with an error.

If a **yes** value has been specified in the EXACT_FIT field of the **logical_volume_policy** stanza of the **/tmp/vgdata/***vgname***/***vgname***.data** file, the **restvg** command uses the map files to preserve the placement of the physical partitions for each logical volume. The target disks must be of the same size or larger then the source disks specified in the **source_disk_data** stanzas of the *vgname***.data** file.

Note:

- To view the files in the backup image or to restore individual files from the backup image, the user must use the **restore** command with the **-T** or **-x** flag, respectively. (Refer to the **restore** command for more information.)
- When you run the **varyonvg** command on the volume group, the logical track group (LTG) size will be set to the common max transfer size of the disks.

Flags

Item	Description
-b *Blocks*	Specifies the number of 512-byte blocks to read in a single input operation. If this parameter is not specified the default of 100 is used by the **restore** command. Larger values result in larger physical transfers to tape devices.
DiskName...	Specifies the names of disk devices to be used instead of the disk devices listed in the *vgname***.data** file. Target disk devices must be defined as empty physical volumes; that is, they must contain a physical volume identifier and must not belong to a volume group. If the target disk devices are new, they must be added to the system using the **mkdev** command. If the target disk devices belong to a volume group, they must be removed from the volume group using the **reducevg** command.
-d *FileName*	The -d flag is an optional flag, which, if specified, must be followed by a filename. This file will be used as the **vgname.data** file instead of the one contained within the backup image being restored. The filename can be specified by either a relative or an absolute pathname.
-f *Device*	Specifies the device name of the backup media. The default is **/dev/rmt0**.

Item	Description
-l	Displays useful information about a volume group backup. This flag requires the **-f** *device* flag. This flag causes **restvg** to display information such as volume group, date and time backup was made, uname output from backed up system, oslevel, recommended maintenance and technology level, backup size in megabytes, and backup shrink size in megabytes. The shrink size is the size of the data on all filesystems. The full size is the total size of each filesystem (unused + data). The **-l** flag also displays the logical volume and filesystem information of the backed up volume group, equivalent to running "**lsvg -l** *vgname*".
`-n`	Specifies that the existing MAP files are ignored. The `-n` flag overrides the value of the `EXACT_FIT` field in the logical_volume_policy stanza of the *vgname*`.data` file.
-P *PPsize*	Specifies the number of megabytes in each physical partition. If not specified, **restvg** uses the best value for the *PPsize*, dependent upon the largest disk being restored to. If this is not the same as the size specified in the *vgname***.data** file, the number of partitions in each logical volume will be appropriately altered with respect to the new *PPsize*. If a *PPsize* is specified that is smaller than appropriate for the disk sizes, the larger *PPsize* will be used. If a *PPsize* is specified that is larger than appropriate for the disk sizes, the specified larger *PPsize* will be used.
-q	Specifies that the usual prompt not be displayed before the restoration of the volume group image. If this flag is not specified, the prompt displays the volume group name and the target disk-device names.
-r	Recreates a volume groups structure only. This allows restvg to create (for the specified backup *FileName* or *Device*) the volume group, logical volumes, and filesystems, from the backup, without restoring any files or data. This is useful for users who use third party software for restoring data and just need all the AIX logical volume structure in place. **Note:** be used with either the **-f** *Device* flag or the **-d** *FileName* flag. This is because **restvg** requires a backup image or *vgname***.data** file to get all the information it needs to recreate the logical volume structure of the volume group desired.
-s	Specifies that the logical volumes be created at the minimum size possible to accommodate the file systems. This size is specified by the value of `LV_MIN_LPS` field of the **lv_data** stanza of the *vgname***.data** file (where *vgname* is the name of the volume group). The **-s** flag overrides the values of the `SHRINK` and `EXACT_FIT` fields in the **logical_volume_policy** stanza of the *vgname***.data** file. The **-s** flag causes the same effect as values of `SHRINK=yes` and `EXACT_FIT=no` would cause.

Examples

1. To restore the volume group image from the **/dev/rmt1** device, onto the **hdisk2** and **hdisk3** disks, enter:

```
restvg -f/dev/rmt1 hdisk2 hdisk3
```

2. To restore the volume group image saved in **/mydata/myvg** file onto the disks specified in the *vgname***.data** file contained within the backup image, enter:

```
restvg -f/mydata/myvg
```

3. To recreate the volume group logical volume structure without restoring any files using only the *vgname***.data** file **/home/my_dir/my_vg.data**, enter:

```
restvg -r -d /home/my_dir/my_vg.data
```

   Note: *vgname***.data** files can be created for a volume group by using the **mkvgdata** command.

4. To recreate the volume group logical volume structure without restoring any files using the *vgname***.data** file inside of the volume group backup located on the tape in **/dev/rmt0**, enter the following:

```
restvg -r -f /dev/rmt0
```

5. To display volume group information about the volume group backed up on the tape in **/dev/rmt0**, enter:

```
restvg -l -f /dev/rmt0
```

6. To restore the volume group image from the **/dev/usbms0** device, onto the disks specified in the **vgname.data** file contained within the backup image, enter the following command:

```
restvg –f /dev/usbms0
```

   Note: For information about backing up a volume group, see the **listvgbackup** command. To restore individual files from a volume group backup, see the **restorevgfiles** command.

restwpar Command

Purpose

Restores a workload partition.

Syntax

`restwpar` [`-a`] [`-A`] [`-b` *Blocks*] [`-B` *devexportsFile*] [`-C`] [`-d` *Directory*] [`-f` *Device*] [`-F`] [`-h` *hostName*] [`-i` *imagedataFileName*] [`-k`] [`-K`] [`-M` *mkwparFlags*] [`-n` *WparName* [`-r`] [`-s`] [`-S` { `a` | `A` | `f` | `F` | `n` }] [`-U`] [`-w` *wparSpecificationFile*]

Description

The **restwpar** command creates a workload partition from a workload partition backup image that was created by the **savewpar**, **mkcd**, or **mkdvd** command.

Warning: The **restwpar** command should not be run while an AIX Live Update operation is in progress.

A workload partition backup image contains an `image.data` file and a workload partition specification file that is used to establish the characteristics of workload partition *WparName*. You can use the `-i` and `-w` flags to override these default files.

If you do not specify the `-f` flag, the `/dev/rmt0` device is used as the input device.

If you specify a value of Yes in the EXACT_FIT field of the `logical_volume_policy` stanza of the `/tmp/wpardata/`*WparName*`/image.data` file, the **restwpar** command uses the map files to preserve the placement of the physical partitions for each logical volume.

If user volume groups are configured with a rootvg WPAR, then they are not automatically imported after restoring a rootvg WPAR.

Notes:

- To view the files in the backup image or to restore individual files from the backup image, use the **lssavewpar**, **restwparfiles**, or **restore** command with the `-T` or the `-x` flag.
- For shared workload partitions (WPARs), if the installation history of the source system is different from the installation history of the target system, the **restwpar** command and the **syncroot** command might fail for few filesets. You might see a failure message that is similar to the following example for the **syncroot** command, at the end of the `restwpar` operation:

  ```
  syncroot: Error synchronizing installp software
  syncroot: Returns Status = FAILURE
  ```

  You must restore or migrate the shared WPAR to a logical partition (LPAR) that has the installation history similar to the installation history of the source LPAR.

Flags

Item	Description
`-a`	Automatically resolves conflicting static settings if required. Resolvable settings are name, host name, base directory, and network configuration.
`-A`	Starts the workload partition each time when the `/etc/rc.wpars` command is run, which is added to the global `/etc/inittab` to run on each system start. The default is not to start the workload partition automatically.
`-b` *Blocks*	Specifies the number of 512-byte blocks to read in a single input operation. If you do not specify the *Blocks* parameter, the default value of 100 is used by the `restore` command. Larger values result in larger physical transfers to tape devices.
`-B` *devexportsFile*	Specifies a substitute file that can be used as the master device exports file. This file must match the format of a Device exports File. If you do not specify a file name, the `/etc/wpars/devexports` file is used.
`-C`	Forces the creation of the named workload partition, even when a compatibility check fails between the system from the backup image and the system where the backup is being restored. If the workload partition is not compatible with the target system. It might not be operable. If the operating system of the global system is at a later technology level or service pack level than the WPAR that has different modification or fix levels in the VRMF (version, release, modification and fix level), the workload partition (WPAR) can be synchronized with the new global system. Different factors affect the success of the synchronization. Review the logs after the synchronization operation is complete. Any updates that are applied to the new global system must be committed, and the updates to the WPAR must be committed before you back up the WPAR. If the new global system is installed on a system that is running AIX 6100-08 or 7100-02 technology levels, or earlier, you must run the **cp_bos_updates** command before you restore the workload partition for the synchronization to work.

Item	Description
-d *Directory*	Specifies a base directory for the workload partition. If you do not specify a directory name, the directory name from the WPAR specification file is used.
-f *Device*	Specifies the device name of the backup media. The default value is /dev/rmt0.
-F	Forces the creation of the named workload partition. If the named workload partition exists, it is stopped if active, and then removed, before the new workload partition is created.
-h *hostname*	Specifies a host name for theworkload partition. If not specified, the **mkwpar** command uses the workload partition name for the host name.
-i *imagedataFileName*	An optional flag that specifies a file name. The file is used as theimage.data file instead of the one contained within the backup image that is being restored.
-k	Creates logical volumes with minimum sizes from the backup.
-K	Creates the post-installation customization script.
-M *mkwparFlags*	Specifies the flags to pass directly to the **mkwpar** command to create the workload partition. The -M flag is used to pass other flags to the **mkwpar** command. If a flag is passed through its own option and through the -M flag, both flags are passed to the **mkwpar** command. **Note:** The *mkwparFlags* value cannot include the -iand -f flags as these flags are reserved for use by the **restwpar** command. Specifying the -i or -f flag as the *mkwparFlags* value causes an error.
-n *WparName*	Specifies the name for the workload partition to be created. If you do not specify the -n flag, the *WparName* is taken from the WPAR specification file.
-r	Duplicates the network name resolution configuration from the global system. The following files, if they exist, are copied into the workload partition: • /etc/resolv.conf • /etc/hosts • /etc/netsvc.conf • /etc/irs.conf • /etc/networks If the NSORDER environment variable is defined in the calling environment, it is added to the /etc/environment file of the workload partition.
-s	Starts the workload partition after it is created.

Item	Description
-S { a | A | f | F | n }	Specifies the type of synchronization to use after files are restored from the backup to synchronize the levels of software in the workload partition with the levels of the software in the global environment. **a** Causes additional installations with no removal of software. This option is the default. **A** Causes additional installations with no removal of software, and ignores any errors in synchronization. **Important:** If you specify -S A, the workload partition might be in an unusable state. **f** Causes additional installations, software rejection, and deinstallation. **F** Causes additional installations, software rejection, and deinstallation. This option ignores any errors in synchronization. **Important:** If you specify -S F, the workload partition might be in an unusable state. **n** Prevents the synchronization processing after the files are restored. **Important:** If you specify -S n, the workload partition might be in an unusable state.
-U	Specifies that the existing MAP files are ignored. The -U flag overrides the value of the EXACT_FIT field in the logical_volume_policy stanza of the *WparName*.data file.
-w *wparSpecificationFile*	An optional flag that specifies a file name. The file is used as the WPAR specification file rather than the version in the WPAR backup image by the **mkwpar** command.

Examples

1. To restore the workload partition image from the /dev/rmt1 device, enter the following command:

```
restwpar -f/dev/rmt1
```

2. To restore the workload partition image that is saved in the /mydata/wpar.img file with name mywpar and base directory /wpars/mywpar, enter the following command:

```
restwpar -f/mydata/wpar.img -n mywpar -d /wpars/mywpar
```

3. To restore the workload partition image from the /dev/usbms0 device, enter the following command:

```
restwparfiles -f/dev/usbms0
```

restwparfiles Command

Purpose

Restores files from a workload partition backup source.

Syntax

restwparfiles [**-b** *blocks*] [**-f** *device*] [**-a**] [**-m**] [**-n**] [**-d** *path*] [**-D**] [**-V**] [*file_list*]

Description

The **restwparfiles** command restores files from tape, file, CD-ROM, or other workload partition backup source. The **restwparfiles** command also works for multivolume backups such as multiple CDs, DVDs, USB disks, or tapes.

Flags

Item	Description
-a	Verifies the physical block size of the tape backup, as specified by the **-b** *blocks* flag. You might need to alter the block size to read the backup. The **-a** flag is valid only when you specify the device in the **-f** flag as tape.
-b *blocks*	Specifies the number of 512-byte blocks to read in a single input operation, as defined by the *blocks* parameter. If you do not specify the *blocks* parameter, the default number of blocks to read is 100.
-d *path*	Specifies the directory path where the files are restored, as defined by the *path* parameter. If you do not specify the **-d** flag, the current working directory is used. **Restriction:** The directory path where the files are restored must not be root (**/**) in the global environment, either through the use of **-d /** or if the current working directory is **/** and the **-d** flag is not specified.
-D	Produces debug output.
-f *device*	Specifies the device containing the backup (file, tape, CD-ROM, or other source) as defined by the *device* parameter. When you do not specify the **-f** flag, the default device is **/dev/rmt0**.
-m	Restores only informational and control files from the image. Use the flag to restore the **image.data** and **wpar.spec** files from the backup image. Files are restored under the **./.savewpar_dir/** directory.
-n	Specifies that ACLs, PCLs, or extended attributes are not to be restored.
-V	Verifies a tape backup. The **-V** flag requires the **-f** *device* flag and can be used to specify only tape devices. The **-V** flag causes the **restwparfiles** command to verify the readability of each file header on the volume group backup and print any errors that occur to the standard error log (**stderr**) file.

Parameters

Item	Description
file_list	Identifies the list of files to be restored. Specify the full path of the files relative to the current directory in the space-separated list. All files in the specified directory are restored unless directed. If you are restoring all files in a directory, write to a temporary folder instead of the **root** directory.

Examples

1. To read the backup stored on the **/dev/cd1** device and restore all files to the **/data/myfiles** directory, enter the following command:

```
restwparfiles -f /dev/cd1 -d /data/myfiles
```

2. To read the backup from the default device at twenty 512-byte blocks at a time and restore the **/myapp/app.c** file to the current directory, enter the following command:

```
restwparfiles -b 20 ./myapp/app.h
```

3. To read the backup stored on the **/dev/cd1** device and restore the **/myapp/app.c** file to the **/data/testcode** directory, enter the following command:

```
restwparfiles -f /dev/cd1 -d /data/testcode ./myapp/app.c
```

4. To read the backup stored at **/dev/usbms0** and restore all files to the **/data/myfiles** directory, enter the following command:

```
restwparfiles –f /dev/usbms0 –d /data/myfiles
```

resumevsd Command

Purpose

Activates an available virtual shared disk.

Syntax

`resumevsd` [`-p` | `-b` | `-l` *server_list*] {`-a` | *vsd_name* ...}

Description

The **resumevsd** command brings the specified virtual shared disks from the suspended state to the active state. The virtual shared disks remains available. Read and write requests which had been held while the virtual shared disk was in the suspended state are resumed.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Resume a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Flags

-p
Specifies that the primary server node defined for the global volume group is to be the active server. The `-p` flag is not valid for CVSD.

-b
Specifies that the secondary server node defined for the global volume group is to be the active server. The `-b` flag is not valid for CVSD.

-a
Specifies that all the virtual shared disks that have been defined are to be resumed.

-l
Passes the `server_list` to the driver.

Parameters

vsd_name
Specifies a virtual shared disk.

Security

You must have `root` authority to run this command.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to *RSCT Administration Guide* .

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Examples

To bring the virtual shared disk **vsd1vg1n1** from the suspended state to the active state, enter:

```
resumevsd vsd1vg1n1
```

Location

`/opt/rsct/vsd/bin/resumevsd`

rev Command

Purpose

Reverses characters in each line of a file.

Syntax

rev [*File ...*]

Description

The **rev** command copies the named files to standard output, reversing the order of characters in every line. If you do not specify a file, the **rev** command reads standard input.

Examples

To reverse characters in each line of a file, type:

```
rev file
```

If the `file` file contains the text:

```
abcdefghi
123456789
```

then the **rev** command displays:

```
ihgfedcba
987654321
```

Files

Item	Description
/usr/bin/rev	Contains the **rev** command.

revnetgroup Command

Purpose

Reverses the listing of users and hosts in network group files in NIS maps.

Syntax

/usr/sbin/revnetgroup { **-h** | **-u** } [*File*]

Description

The **revnetgroup** command reverses the order in which hosts and users are listed in the **/etc/netgroup** file. The **revnetgroup** command is called from the **/var/yp/Makefile** file to produce output for creating either the **netgroup.byuser** or **netgroup.byhost** NIS map. Each line in the output file begins with a key formed by concatenating the host or user name with the domain name. Following the key is a list of groups to which the host or user belongs. The list is preceded by a tab, and each group is separated by a comma.

> **Note:** The list of groups does not use the names of universal groups (groups that include all users in the network). Universal groups are listed under * (asterisk).

The **revnetgroup** command takes an optional file name if the default **/etc/netgroup** file is not desired. This feature provides users with flexibility to create custom network group maps.

Flags

Item	Description
-h	Produces output for creating the **netgroup.byhost** map.
-u	Produces output for creating the **netgroup.byuser** map.

Examples

1. To cause the **/etc/netgroup** file to list user names before host names, modify the appropriate stanza in the **/var/yp/Makefile** to read:

   ```
   revnetgroup -u
   ```

2. To create a new network group file, called newgroup, in the **/etc** directory, modify the appropriate stanza in the **/var/yp/Makefile** to read:

   ```
   revnetgroup -h newgroup
   ```

   The **-h** flag used in this example causes the new **/etc/newgroup** file to list host names before user names.

Files

Item	Description
/etc/netgroup	Contains lists of users and hosts in network groups.
/var/yp/Makefile	Contains rules for making NIS maps.

rexd Daemon

Purpose

Executes programs for remote machines.

Syntax

/usr/sbin/rpc.rexd

Description

The **rexd** daemon executes programs for remote machines when a client issues a request to execute a program on a remote machine. The **inetd**daemon starts the **rexd** daemon from the **/etc/inetd.conf** file.

Noninteractive programs use standard file descriptors connected directly to TCP connections. Interactive programs use pseudo-terminals, similar to the login sessions provided by the **rlogin** command. The **rexd** daemon can use the network file system (NFS) to mount the file systems specified in the remote execution request. Diagnostic messages are normally printed on the console and returned to the requester.

> **Note:** A root user cannot execute commands using **rexd** client programs such as the **on** command.

Files

Item	Description
/tmp_rex/rexd	Contains temporary mount points for remote file systems.
/etc/exports	Lists the directories that the server can export.
inetd.conf	Starts RPC daemons and other TCP/IP daemons.
/etc/passwd	Contains an entry for each user that has permission to log in to the machine.

rexec Command

Purpose

Executes commands one at a time on a remote host.

Syntax

rexec [**-a**][**-d** | **-n**] [**-i**] *Host Command*

Description

The **/usr/bin/rexec** command executes a command on the specified remote host.

The **rexec** command provides an automatic login feature by checking for a **$HOME/.netrc** file that contains the user name and password to use at the remote host. If such an entry is not found or if your

system is operating in secure mode (see the **securetcpip** command), the **rexec** command prompts for a valid user name and password for the remote host. In both cases, **rexec** causes **rexecd** on the remote system to use the default `compat` login authentication method for the user. **rexecd** does not look at the **/etc/security/user** file on the remote system for alternative authentication methods. You can also override the automatic login feature by specifying the **-n** flag on the **rexec** command line.

Restriction: Any user with a user ID less than or equal to 128 cannot log in to the remote Trusted AIX system.

Flags

Item	Description
-a	Indicates the standard error of the remote command is the same as standard output. No provision is made for sending arbitrary signals to the remote process.
-d	Enables socket-level debugging.
-i	Prevents reading the stdin.
-n	Prevents automatic login. With the **-n** flag specified, the **rexec** command prompts for a user name and password to use at the remote host, rather than searching for a **$HOME/.netrc** file.

Parameters

Item	Description
Command	Specifies the command, including any flags or parameters, to be executed on the remote host.
Host	Specifies in alphanumeric form the name of the host where the command is to be executed.

Examples

1. To execute the **date** command on a remote host, enter:

   ```
   rexec host1 date
   ```

   The output from the date command is now displayed on the local system. In this example, the **$HOME/.netrc** file on the local host contains a user name and password valid at the remote host.

   If you do not have a valid entry in the **$HOME/.netrc** file for the remote host, you will be prompted for your login ID and password. After you have entered the requested login information, the output from the date command is displayed on the local system.

2. To override the automatic login feature and execute the **date** command on a remote host, enter:

   ```
   rexec -nhost1 date
   ```

   Enter your name and password when prompted.

   The output from the date command is now displayed on the local system.

3. To list the directory of another user on a remote host, enter:

   ```
   rexec host1 ls -l /home/karen
   ```

   The directory listing of user `karen` on remote host `host1` is displayed on the local system.

   If you do not have a valid entry in the **$HOME/.netrc** file for the remote host, you will be prompted for your login ID and password. After you have entered the requested login information, the directory listing of user `karen` on remote host `host1` is displayed on the local system.

rexecd Daemon

Purpose

Provides the server function for the **rexec** command.

Syntax

> **Note:** The **rexecd** daemon is normally started by the **/etc/inetd.conf** or **kill -1** *InetdPID* command to inform the **inetd** daemon of the changes to its configuration file.

Note: The **rexecd** daemon ignores invalid options and if the **syslog** facility is enabled, the information will be logged to the system log.

Flags

Item	Description
-s	Enables socket-level debugging.
-c	Prevents reverse name resolution. When the **-c** flag is not specified, the **rexecd** daemon will fail if the reverse name resolution of the client fails.

Service Request Protocol

When the **rexecd** daemon receives a request, it initiates the following protocol:

1. The server reads characters from the socket up to a null (\0) byte and interprets the resulting string as an ASCII number (decimal).
2. If the number received is nonzero, the **rexecd** daemon interprets it as the port number of a secondary stream to be used for standard error output. The **rexecd** daemon then creates a second connection to the specified port on the client machine.
3. The **rexecd** daemon retrieves a null-terminated user name of up to 16 characters on the initial socket.

Security

The **rexecd** daemon is a PAM-enabled application with a service name of *rexec*. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the **usw** stanza of **/etc/security/login.cfg**, to PAM_AUTH as the root user.

The authentication mechanisms used when PAM is enabled depend on the configuration for the **rexec** service in **/etc/pam.conf**. The **rexecd** daemon requires **/etc/pam.conf** entries for the **auth**, **account**, and **session** module types. Listed below is a recommended configuration in **/etc/pam.conf** for the **rexec** service:

```
#
# AIX rexec configuration
#
rexec auth      required     /usr/lib/security/pam_aix

rexec account   required     /usr/lib/security/pam_aix

rexec session   required     /usr/lib/security/pam_aix
```

rgb Command

Purpose

Creates the database used by the X-Window system server for colors.

Syntax

rgb [*DatabaseName*] [*<InputfileName*]

Description

The **rgb** command reads lines from standard input and inserts them into its database to associate color names with specific red, green, and blue (RGB) values.

The **rgb** command produces two output files: *DatabaseName*.dir and *DatabaseName*.pag. If you do not specify a database file name, the default names **rgb.dir** and **rgb.pag** are used.

Each color entry is in the form:

```
Red Green Blue Colorname
```

where the *Red*, *Green*, and *Blue* elements are integer values ranging from 0-255. The actual color is determined by how the elements are combined. Each element can range from no intensity (0) to full intensity (255). The *Colorname* parameter can be descriptive or fanciful. For example, the sequence `250 250 250` could be named `white` or `snow`. Two or more entries can share the same element numbers or names.

Parameters

Item	Description
DatabaseName	Specifies the database to create for the output data.
<InputFileName	Specifies the name of the input file.

Examples

1. The following example shows a portion of an input file:

```
248 248 255     ghost white
245 245 245     white smoke
255 250 240     floral white
253 245 230     old lace
250 240 230     linen
255 218 185     peach puff
255 248 220     cornsilk
255 250 205     lemon chiffon
245 255 250     mint cream
240 255 255     azure
```

2. The following example generates the output files **Newcolor.dir** and **Newcolor.pag**.

```
rgb Newcolor < rgb.txt
```

   where **Newcolor** is the *DatabaseName* and **rgb.txt** is the *InputFileName*.

Files

Item	Description
/usr/lib/X11/rgb.txt	The default rgb database input file.

ripquery Command

Purpose

Queries the RIP gateways.

Syntax

ripquery [**-1**] [**-2**] [**-**[**a5**] *authkey*] [**-n**] [**-N** *dest***[/***mask***]** [**-p**] [**-r**] [**-v**] [**-w** *time*] *gateway*...

Description

The **ripquery** command is used to request all routes known by a RIP *gateway* by sending a RIP **REQUEST** or **POLL** command. The routing information in any routing packets returned is displayed numerically and symbolically. The **ripquery** command is intended to be used as a tool for debugging *gateway*s, not for network management. SNMP is the preferred protocol for network management.

Flags

Item	Description
-1	Send the query as a version 1 packet.
-2	Send the query as a version 2 packet (default).
-[**a5**] *authkey*	Specifies the authentication password to use for queries. If **-a** is specified, an authentication type of SIMPLE will be used, if **-5** is specified, an authentication type of MD5 will be used, otherwise the default is an authentication type of NONE. Authentication fields in incoming packets will be displayed, but not validated.
-n	Prevents the address of the responding host from being looked up to determine the symbolic name.
-N *dest***[/***mask***]**	Specifies that the query should be for the specified *dest/mask* instead of complete routing table. The specification of the optional mask implies a version 2 query. Up to 23 requests about specific destinations may be included in one packet.
-p	Uses the RIP **POLL** command to request information from the routing table. This is the default. If there is no response to the RIP **POLL** command, the RIP **REQUEST** command is tried. **gated** responds to a **POLL** command with all the routes learned via RIP.
-r	Uses the RIP **REQUEST** command to request information from the *gateway*'s routing table. Unlike the RIP **POLL** command, all *gateway*s should support the RIP **REQUEST**. If there is no response to the RIP **REQUEST** command, the RIP **POLL** command is tried. **gated** responds to a **REQUEST** command with all the routes he anounces out the specified interface.
-v	Version information about **ripquery** is displayed before querying the *gateway*s.
-w *time*	Specifies the time in seconds to wait for the initial response from a *gateway*. The default value is 5 seconds.

rksh Command

Purpose

Invokes the restricted version of the Korn shell.

Syntax

rksh [-i] [{ + | - }{ a e f h k m n p t u v x }] [-o *Option ...*] [-c *String* | -s | *File* [*Parameter*]]

Note: Preceding a flag with + (plus) rather than - (minus) turns off the flag.

Description

The `rksh` command invokes a restricted version of the Korn shell. It allows administrators to provide a controlled shell environment to the users. There is also a restricted version of `rksh` available for the enhanced Korn shell, called `rksh93`.

With a restricted shell a user cannot:

- Change the current working directory.
- Set the value of the SHELL, ENV, or PATH variable.
- Specify the pathname of a command that contains a / (slash).
- Redirect output of a command with > (right caret), >| (right caret, pipe symbol), <> (left caret, right caret), or >> (two right carets).

Flags

Item	Description
`-a`	Exports automatically all subsequent parameters that are defined.
`-c` *String*	Causes the Korn shell to read commands from the *String* variable. This flag cannot be used with the **-s** flag or with the *File*[*Parameter*] parameter.
`-e`	Executes the **ERR** trap, if set, and exits if a command has a nonzero exit status. This mode is disabled while reading profiles.
`-f`	Disables file name substitution.
`-h`	Designates each command as a tracked alias when first encountered.
`-i`	Indicates that the shell is interactive. An interactive shell is also indicated if shell input and output are attached to a terminal (as determined by the **ioctl** subroutine). In this case, the **TERM** environment variable is ignored (so that the **kill 0** command does not kill an interactive shell) and the **INTR** signal is caught and ignored (so that a wait state can be interrupted). In all cases, the **QUIT** signal is ignored by the shell.
`-k`	Places all parameter assignment arguments in the environment for a command, not just those arguments that precede the command name.
`-m`	Runs background jobs in a separate process and prints a line upon completion. The exit status of background jobs is reported in a completion message. On systems with job control, this flag is turned on automatically for interactive shells.
`-n`	Reads commands and checks them for syntax errors, but does not execute them. This flag is ignored for interactive shells.

Item	Description
-o *Option*	Prints the current option settings and an error message if you do not specify an argument. You can use this flag to enable any of the following options: **allexport** Same as the **-a** flag. **errexit** Same as the **-e** flag. **bgnice** Runs all background jobs at a lower priority. This is the default mode. **emacs** Enters an emacs-style inline editor for command entry. **gmacs** Enters a gmacs-style inline editor for command entry. **ignoreeof** Does not exit the shell when it encounters an end-of-file character. You must use the **exit** command, or override the flag and exit the shell by pressing the Ctrl-D key sequence more than 11 times. **keyword** Same as the **-k** flag. **markdirs** Appends a **/** (slash) to all directory names that are a result of filename substitution. **monitor** Same as the **-m** flag. **noclobber** Prevents redirection from truncating existing files. When you specify this option, use the redirection symbol **>|** (right caret, pipe symbol) to truncate a file. **noexec** Same as the **-n** flag. **noglob** Same as the **-f** flag. **nolog** Prevents function definitions from being saved in the history file. **nounset** Same as the **-u** flag. **privileged** Same as the **-p** flag. **verbose** Same as the **-v** flag. **trackall** Same as the **-h** flag. **vi** Enters the insert mode of a vi-style inline editor for command entry. Entering escape character 033 puts the editor into the move mode. A return sends the line. **viraw** Processes each character as it is typed in vi mode. **xtrace** Same as the **-x** flag. You can set more than one option on a single **rksh** command line.

Item	Description
-p	Disables the processing of the **$HOME/.profile** file when you use the shell as a login shell.
-s	Causes the **rksh** command to read commands from the standard input. Shell output, except for the output of the special commands, is written to file descriptor 2. This parameter cannot be used with the **-c** flag or with the *File*[*Parameter*] parameter.
-t	Exits after reading and executing one command.
-u	Treats unset parameters as errors when substituting.
-v	Prints shell input lines as they are read.
-x	Prints executed commands and their arguments.

Files

Item	Description
/usr/bin/rksh	Contains the path name to the restricted Korn shell.
/tmp/sh*	Contains temporary files that are created when a shell is opened.

rlogin Command

Purpose

Connects a local host with a remote host.

Syntax

rlogin *RemoteHost* [**-e** *Character*] [**-8**] [**-l** *User*] [**-f** | **-F**] [**-k** *realm*]

Description

The **/usr/bin/rlogin** command logs into a specified remote host and connects your local terminal to the remote host.

The remote terminal type is the same as that given in the **TERM** local environment variable. The terminal or window size is also the same, if the remote host supports them, and any changes in size are transferred. All echoing takes place at the remote host, so except for delays, the terminal connection is transparent. The Ctrl-S and Ctrl-Q key sequences stop and start the flow of information, and the input and output buffers are flushed on interrupts.

Remote Command Execution

When using the **rlogin** command, you can create a link to your path using a host name as the link name. For example:

```
ln -s /usr/bin/rsh HostName
```

Entering the host name specified by the *HostName* parameter with an argument (command) at the prompt, automatically uses the **rsh** command to remotely execute the command specified on the command line of the remote host specified by the *HostName* parameter.

Entering the host name specified by the *HostName* parameter without an argument (command) at the prompt, automatically uses the **rlogin** command to log in to the remote host specified by the *HostName* parameter.

In addition to the preceding conditions, the **rlogin** command also allows access to the remote host if the remote user account does not have a password defined. However, for security reasons, the use of a password on all user accounts is recommended.

The **rlogin** command execs (using the **exec** command) the **/usr/sbin/login** file to validate a user. This 1) allows all user and device attributes to take effect on telnet connections and 2) causes remote logins to count against the maximum number of login sessions allowable at a time (determined by the `maxlogins` attribute). Attributes are defined in the **/etc/security/user** and **/etc/security/login.cfg** files.

POSIX Line Discipline

The **rlogind** and **telnetd** daemons use POSIX line discipline to change the line discipline on the local TTY. If POSIX line discipline is not used on the local TTY, echoing other line disciplines may result in improper behavior. TCP/IP must have POSIX line discipline to function properly.

Flags

Item	Description
-8	Allows an 8-bit data path at all times. Otherwise, unless the start and stop characters on the remote host are not Ctrl-S and Ctrl-Q, the **rlogin** command uses a 7-bit data path and parity bits are stripped.
-e *Character*	Changes the escape character. Substitute the character you choose for *Character*.
-f	Causes the credentials to be forwarded. This flag will be ignored if Kerberos 5 is not the current authentication method. Authentication will fail if the current DCE credentials are not marked forwardable.
-F	Causes the credentials to be forwarded. In addition, the credentials on the remote system will be marked forwardable (allowing them to be passed to another remote system). This flag will be ignored if Kerberos 5 is not the current authentication method. Authentication will fail if the current DCE credentials are not marked forwardable.
-k *realm*	Allows the user to specify the realm of the remote station if it is different from the local systems realm. For these purposes, a realm is synonymous with a DCE cell. This flag will be ignored if Kerberos 5 is not the current authentication method.
-l *User*	Changes the remote user name to the one you specify. Otherwise, your local user name is used at the remote host.

Security

There are multiple authentication methods, each requiring different things to be set in order to allow the connection.

For Standard Authentication

The remote host allows access only if one or both of the following conditions is satisfied:

- The local host is included in the remote **$HOME/.rhosts** file in the remote user account.

Although you can set any permissions for the **$HOME/.rhosts** file, it is recommended that the permissions of the .rhosts file be set to 600 (read and write by owner only).

Note: The **AUTHSTATE** environment variable indicates the registry to which the user authenticates. For example, an LDAP user that is defined on the LDAP server has the **AUTHSTATE** set to LDAP if the user logs in to the remote system with a password. But if a user is authenticated through an entry in the **$HOME/.rhosts and /etc/hosts.equiv** files, the **AUTHSTATE** environment variable for that user is set to compat regardless of where the user ID is defined.

For Kerberos 5 Authentication

The remote host allows access only if all of the following conditions are satisfied:

- The local user has current DCE credentials.
- The local and remote systems are configured for Kerberos 5 authentication (On some remote systems, this may not be necessary. It is necessary that a daemon is listening to the klogin port).
- The remote system accepts the DCE credentials as sufficient for access to the remote account. See the html

rlogind Daemon

Purpose

Provides the server function for the **rlogin** command.

Syntax

> **Note:** The **rlogind** daemon is normally started by the **inetd** daemon. It can also be controlled from the command line, using SRC commands.

/usr/sbin/rlogind [**-a**] [**-c**] [**-l**] [**-n**] [**-s**]

Description

The **/usr/sbin/rlogind** daemon is the server for the **rlogin** remote login command. The server provides a remote login facility.

Changes to the **rlogind** daemon can be made using the System Management Interface Tool (SMIT) or System Resource Controller (SRC), by editing the **/etc/inetd.conf** or **/etc/services** file. Entering `rlogind` at the command line is not recommended. The **rlogind** daemon is started by default when it is uncommented in the **/etc/inetd.conf** file.

The **rlogind** daemon ignores unrecognized options and log this information through the **syslog** service if the **syslog** service is enabled in the system.

The **inetd** daemon get its information the /**etc**/**inetd.conf** file and the /**etc**/**services** file.

After changing the **/etc/inetd.conf** or /**etc**/**services** file, run the **refresh -s inetd** or **kill -1** *InetdPID* command to inform the **inetd** daemon of the changes to its configuration file.

Service Request Protocol

When the **rlogind** daemon receives a service request, the daemon initiates the following protocol:

1. The **rlogind** daemon checks the source port number for the request. If the port number is not in the range 512-1023, the **rlogind** daemon terminates the connection.
2. The **rlogind** daemon uses the source address of the initial connection request to determine the name of the client host. If the name cannot be determined, the **rlogind** daemon uses the dotted-decimal representation of the client host address.

Error Messages

The following error messages are associated with the **rlogind** daemon:

Item	Description
Try again.	A **fork** command made by the server has failed.
/usr/bin/shell:	No shell. The shell specified for the shell variable cannot be started. The shell variable may also be a program.

Flags

Item	Description
-a	Disables **pty** speed enhancement feature.
-c	Suppresses the sanity check of a host name lookup.
-l	Prevents any authentication based on the user's **$HOME/.rhosts** file. However, a root user is automatically logged in when there is a **.rhosts** file in root's home directory as specified by the **/etc/passwd** file.
-n	Disables transport-level keep-alive messages. The messages are enabled by default.
-s	Turns on socket level debugging.

Security

The **rlogind** daemon is a PAM-enabled application with a service name of *rlogin*. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the **usw** stanza of **/etc/security/login.cfg**, to PAM_AUTH as the root user.

The authentication mechanisms used when PAM is enabled depend on the configuration for the **rlogin** service in **/etc/pam.conf**. The **rlogind** daemon requires **/etc/pam.conf** entries for the **auth**, **account**, **password**, and **session** module types. Listed below is a recommended configuration in **/etc/pam.conf** for the **rlogin** service:

```
#
# AIX rlogin configuration
#
rlogin auth      sufficient   /usr/lib/security/pam_rhosts_auth
rlogin auth      required     /usr/lib/security/pam_aix

rlogin account   required     /usr/lib/security/pam_aix

rlogin password  required     /usr/lib/security/pam_aix

rlogin session   required     /usr/lib/security/pam_aix
```

Examples

> **Note:** The arguments for the **rlogind** daemon can be specified by using SMIT or by editing the **/etc/inetd.conf** file.

1. To start the **rlogind** daemon, enter the following:

   ```
   startsrc -t rlogin
   ```

   This command starts the **rlogind** subserver.
2. To stop the **rlogind** daemon normally, enter the following:

   ```
   stopsrc -t rlogin
   ```

   This command allows all pending connections to start and existing connections to complete but prevents new connections from starting.
3. To force stop the **rlogind** daemon and all **rlogind** connections, enter the following:

   ```
   stopsrc -f -t rlogin
   ```

   This command terminates all pending connections and existing connections immediately.
4. To display a short status report about the **rlogind** daemon, enter the following:

   ```
   lssrc -t rlogin
   ```

   This command returns the daemon's name, process ID, and state (active or inactive).

rm Command

Purpose

Removes (unlinks) files or directories.

Syntax

rm [**-f**] [**-r**] [**-R**] [**-i**] [**-e**] *File ...*

Description

The **rm** command removes the entries for the specified *File* parameter from a directory. If an entry is the last link to a file, the file is then deleted. If you do not have write permission for a file and the standard input is a terminal, you are prompted with the file name and ask to confirm that you want to delete the file. If you type a y (for yes), the file is deleted, type any other character and the file is not deleted. You do not need read or write permission for the file you want to remove. However, you must have write permission for the directory containing the file.

If the file is a symbolic link, the link is removed, but the file or directory that the symbolic link refers to remains. You do not need write permission to delete a symbolic link, if you have write permission in the directory.

If either of the files **.** (dot) or **..** (dot, dot) are specified as the base name portion of the *File* parameter, the **rm** command writes a diagnostic message to standard error and does nothing more with such parameters.

The **rm** command writes a prompt to standard error and reads a line from standard input if the **-f** flag is not specified, and either the *File* parameter does not have write permission and the standard input is a workstation, or the **-i** flag is specified. If the response is not affirmative, the **rm** command does nothing more with the current file and proceeds to the next file.

The files owned by other users cannot be removed if the sticky bit of the directory is set and the directory is not owned by the user.

> **Note:** The **rm** command supports the **—** (dash, dash) parameter as a delimiter that indicates the end of the flags.

An attempt to remove a file or directory that has been exported for use by the NFS version 4 server will fail with a message saying that the resource is busy. The file or directory must be unexported for NFS version 4 use before it can be removed.

Flags

Item	Description
-e	Displays a message after each file is deleted.
-f	Does not prompt before removing a write-protected file. Does not display an error message or return error status if a specified file does not exist. If both the **-f** and **-i** flags are specified, the last one specified takes affect.
-i	Prompts you before deleting each file. When you use the **-i** and **-r** flags together, the **rm** command also prompts before deleting directories. If both the **-i** and **-f** flags are specified, the last one specified takes affect.

Item	Description
-r	Permits recursive removal of directories and their contents when the *File* parameter is a directory. This flag is equivalent to the **-R** flag.
-R	Permits recursive removal of directories and their contents when the *File* parameter is a directory. This flag is equivalent to the **-r** flag.

Exit Status

This command returns the following exit values:

Item	Description
0	If the **-f** flag was not specified, all the named directory entries were removed; otherwise, all the existing named directory entries were removed.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To delete a file, enter:

   ```
   rm myfile
   ```

   If there is another link to this file, then the file remains under that name, but the name `myfile` is removed. If `myfile` is the only link, the file itself is deleted.

2. To delete a file without first receiving a confirmation prompt, enter:

   ```
   rm -f core
   ```

   No confirmation prompt is issued before the **rm -f** command attempts to remove the file named `core`. However, an error message displays if the `core` file is write-protected and you are not the owner of the file or you do not have root authority. No error message displays when the **rm -f** command attempts to remove nonexistent files.

3. To delete files one by one, enter:

   ```
   rm -i mydir/*
   ```

   After each file name is displayed, enter y to delete the file, or press the Enter key to keep it.

4. To delete a directory tree, enter:

   ```
   rm -ir manual
   ```

   This command recursively removes the contents of all subdirectories of the `manual` directory, prompting you regarding the removal of each file, and then removes the `manual` directory itself, for example:

   ```
   You:  rm -ir manual
   System: rm: Select files in directory manual? Enter y for yes.
   You:  y
   System: rm: Select files in directory manual/draft1? Enter y for yes.
   You:  y
   ```

```
System: rm: Remove manual/draft1?
You:  y
System: rm: Remove manual/draft1/chapter1?
You:  y
System: rm: Remove manual/draft1/chapter2?
You:  y
System: rm: Select files in directory manual/draft2? Enter y for yes.
You:  y
System: rm: Remove manual/draft2?
You:  y
System: rm: Remove manual?
You:  y
```

Here, the **rm** command first asks if you want it to search the `manual` directory. Because the `manual` directory contains directories, the **rm** command next asks for permission to search `manual/draft1` for files to delete, and then asks if you want it to delete the `manual/draft1/chapter1` and `manual/draft1/chapter2` files. The **rm** command next asks for permission to search the `manual/draft2` directory. Then asks for permission to delete the `manual/draft1`, `manual/draft2`, and `manual` directories.

If you deny permission to remove a subdirectory (for example, `manual/draft2`), the **rm** command does not remove the `manual` directory. Instead, you see the message: `rm: Directory manual not empty`.

Files

Item	Description
/usr/bin/rm	Contains the **rm** command.

rmail Command

Purpose

Handles remote mail received through Basic Networking Utilities (BNU).

Syntax

rmail *User*

Description

The **rmail** command interprets incoming mail received through the **uucp** command. It collapses `From` header lines in the form generated by the **bellmail** command into a single line of the form:

```
return-path!sender
```

The **rmail** command passes the processed mail on to the **sendmail** command. The *User* parameter must specify a user recognized by the **sendmail** command.

rmauth Command

Purpose

Removes one or more user-defined authorizations.

Syntax

rmauth [**-R** *load_module*] [**-h**] *Name*

Description

The **rmauth** command removes the user-defined authorization identified by the *Name* parameter. The command only removes existing user-defined authorizations in the authorization database. You cannot remove system-defined authorizations with this command. If an authorization is being referenced in the privileged command database, it cannot be removed until the authorization is no longer referenced by the database.

By default, the **rmauth** command only attempts to remove the specified authorization from the authorization database. You must remove authorizations from the lowest level of a hierarchy before the higher level can be removed. If you specify a higher level authorization and lower-level authorizations still exist, the command fails. To remove a hierarchy of authorizations, specify the **-h** flag. With the **-h** flag, any lower-level authorization beneath the specified authorization is also removed. If any of the lower level authorizations is being referenced in the privileged command database, no authorizations are removed and the entire operation fails.

If the system is configured to use databases from multiple domains, the **rmauth** command finds the first match from the database domains in the order that was specified by the **secorder** attribute of the authorizations stanza in the **/etc/nscontrol.conf** file. Meanwhile, the **rmauth** command removes that authorization entry from the domain. If any matching authorizations from the rest of the domains exist, they are not affected. Use the **-R** flag to remove an authorization from a specific domain.

When the system is operating in enhanced role based access control (RBAC) mode, modifications made to the authorization database are not used for security considerations until the database is sent to the kernel security tables using the **setkst** command.

Flags

Item	Description
-h	Allows removal of a hierarchy of authorizations.
-R *load_module*	Specifies the loadable module to use for the authorization deletion.

Parameters

Item	Description
Name	Specifies the authorization to remove.

Security

The **rmauth** command is a privileged command. You must have the **aix.security.role.remove** authorization to run the command:

Item	Description
aix.security.auth.remove	Required to run the command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files Accessed

File	Mode
/etc/security/authorizations	rw

Examples

1. To remove the `custom.test` authorization, use the following command:

```
rmauth custom.test
```

2. To remove the `custom` authorization and all of its children authorizations, use the following command:

```
rmauth -h custom
```

3. To remove the `custom.test` authorization from LDAP, use the following command:

```
rmauth -h custom.test
```

rmaudrec Command

Purpose

Removes records from the audit log.

Syntax

`rmaudrec` [`-a` | `-n` *node_name1*[`,`*node_name2*]...] [`-S` *subsystem_name*]
`-s` *selection_string* [`-h`] [`-V`]

Description

The `rmaudrec` command is used to delete records in the audit log. The audit log is a facility for recording information about the system's operation. It can include information about the normal operation of the system as well as failures and other errors. It augments the error log functionality by conveying the relationship of the error relative to other system activities. All detailed information about failures is still written to the AIX error log.

Records are created in the audit log by subsystems that have been instrumented to do that. For example, the event response subsystem runs in the background to monitor administrator-defined conditions and then invokes one or more actions when a condition becomes true. Because this subsystem runs in the background, it is difficult for the operator or administrator to understand the total set of events that occurred and the results of any actions that were taken in response to an event. Because the event response subsystem records its activity in the audit log, the administrator can easily view its activity as well as that of other subsystems. In addition, records may sometimes need to be removed explicitly, which can be done using this command.

Each record in the audit log contains named fields. Each field contains a value that provides information about the situation corresponding to the record. For example, the field named `Time` indicates the time at which the situation occurred. Each record has a set of common fields and a set of subsystem-specific fields. The common fields are present in every record in the audit log. The subsystem-specific fields vary from record to record. Their names are only significant when used with a subsystem name because they may not be unique across all subsystems. Each record is derived from a template that defines which subsystem-specific fields are present in the record and defines a format string that is used to generate a message describing the situation. The format string may use record fields as inserts. A subsystem typically has many templates.

The field names can be used as variables in a *selection string* to choose which records are deleted. The selection string is matched against each record using the referenced fields of each record to perform the match. Any records that match will be removed. The selection string is specified with the `-s` flag.

A selection string is an expression composed of field names, constants, and operators. The syntax of a selection string is very similar to an expression in the C programming language. For information on how to specify selection strings, see the *Administering RSCT* guide.

The common field names are:

Field	Description
Time	Specifies the time when the situation occurred that the record corresponds to. The value is a 64-bit integer and represents the number of microseconds since UNIX Epoch (00:00:00 GMT January 1, 1970). See the constants below for specifying the time in more user-friendly formats.
Subsystem	Specifies the subsystem that generated the record. This is a string.
Category	Indicates the importance of the situation corresponding to the audit record, as determined by the subsystem that generated the record. The valid values are: `0` (informational) and 1 (error).
SequenceNumber	Specifies the unique 64-bit integer that is assigned to the record. No other record in the audit log will have the same sequence number.
TemplateId	Specifies the subsystem-dependent identifier that is assigned to records that have the same content and format string. This value is a 32-bit unsigned integer.
NodeName	Specifies the name of the node from which the record was obtained. This field name cannot be used in a selection string.

In addition to the constants in expressions, you can use the following syntax for dates and times with this command:

#mmddhhmmyyyy
: This format consists of a sequence of decimal characters that are interpreted according to the pattern shown. The fields in the pattern are, from left to right: *mm* = month, *dd* = day, *hh* = hour, *mm* = minutes, *yyyy* = year. For example, `#010523042002` corresponds to January 5, 11:04 PM, 2002. The fields can be omitted from right to left. If not present, the following defaults are used: year = the current year, minutes = 0, hour = 0, day = 1, and month = the current month.

#-mmddhhmmyyyy
: This format is similar to the previous one, but is relative to the current time and date. For example, the value `#-0001` corresponds to one day ago and the value `#-010001` corresponds to one month and one hour ago. Fields can be omitted starting from the right and are replaced by 0.

The audit records considered for deletion and matched against the selection string can be restricted to a specific subsystem by using the `-S` flag. If this flag is specified, the subsystem-specific field names can be used in the selection string in addition to the common field names.

The nodes from which audit log records are considered for deletion can be restricted to a set of specific nodes by using the `-n` flag. If this flag is specified, the search will be limited to the set of nodes listed. Otherwise, the search will be performed for all nodes defined within the current management scope as determined by the CT_MANAGEMENT_SCOPE environment variable.

It is advisable to first use the `lsaudrec` command with the same `-s` and `-n` flag values to list the records that will be deleted. This minimizes the possibility of the selection string matching more records than intended.

Flags

-a
: Specifies that records from all nodes in the domain are to be removed. If both the `-n` and the `-a` flags are omitted, records from the local node only are removed.

-n *node_name1*[,*node_name2*]...
: Specifies the list of nodes containing audit log records that will be examined and considered for deletion if they meet the other criteria, such as matching the specified selection string. Node group names can also be specified, which are expanded into a list of node names. If both the `-n` and the `-a` flags are omitted, records from the local node only will be deleted.

-S *subsystem_name*
Specifies a subsystem name. If this flag is present, only records identified by *subsystem_name* are considered for deletion. The records to be deleted can be further restricted by the `-s` flag. If the subsystem name contains any spaces, it must be enclosed in single or double quotation marks.

For backward compatibility, the subsystem name can be specified using the `-n` flag *only* if the `-a` and the `-S` flags are *not* specified.

-s *selection string*
Specifies a selection string. This string is evaluated against each record in the audit log. If the evaluation results in a non-zero result (TRUE), the record is removed from the audit log. If the selection string contains any spaces, it must be enclosed within single or double quotation marks. For information on how to specify selection strings, see the *RSCT: Administration Guide* .

The names of fields within the record can be used in the expression. If the `-S` flag is not specified, only the names of common fields can be used. See the **Description** for a list of the common field names and their data types. If the `-S` flag is specified, the name of any field for the specified subsystem as well as the common field names can be used.

If this flag is not specified, no records will be removed from the audit log.

-h
Writes the command's usage statement to standard output.

-V
Writes the command's verbose messages to standard error.

Parameters

***field_name1* [*field_name2*...]**
Specifies one or more fields in the audit log records to be displayed. The order of the field names on the command line corresponds to the order in which they are displayed. If no field names are specified, `Time`, `Subsystem`, `Severity`, and `Message` are displayed by default. If the management scope is not local, `NodeName` is displayed as the first column by default. See the **Description** for information about these and other fields.

Security

In order to remove records from an audit log when the `-S` flag is omitted, a user must have write access to the target resource class on each node from which records are to be removed. When the `-S` flag is specified, the user must have write access to the audit log resource corresponding to the subsystem identified by the `-S` flag on each node from which records are to be removed.

Authorization is controlled by the RMC access control list (ACL) file that exists on each node.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon is established. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that can be affected by this command.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines (in conjunction with the `-a` and `-n` flags) the management scope that is used for the session with the RMC daemon. The management scope determines the set of possible target nodes where audit log records can be deleted. If the `-a` and `-n` flags are not specified, local scope is used. When either of these flags is specified, CT_MANAGEMENT_SCOPE is used to determine the management scope directly. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output.

Standard Error

If the `-V` flag is specified and the command completes successfully, a message indicating the number of records that were deleted will be written to standard error.

Examples

1. To remove all records from the audit log on every node in the management scope defined by the CT_MANAGEMENT_SCOPE environment variable, enter:

   ```
   rmaudrec -s "Time > 0"
   ```

   or

   ```
   rmaudrec -s "SequenceNumber >= 0"
   ```

2. To remove all records more than a week old on every node in the management scope defined by the CT_MANAGEMENT_SCOPE environment variable, enter:

```
rmaudrec -s "Time < #-0007"
```

3. To remove all records that are more than a day old and created by the abc subsystem on nodes mynode and yournode, enter:

```
rmaudrec -S abc -s "Time < #-0001" -n mynode,yournode
```

Location

/opt/rsct/bin/rmaudrec

rmC2admin Command

Purpose

Remove the configuration files for a distributed C2 System host.

Syntax

rmC2admin [**-m**]

Description

The **rmC2admin** command replaces the distributed C2 System symbolic links with the actual files. The directory **/etc/data.shared** will be removed. When the **-m** flag is used, the **hd10sec** file system and **/etc/data.master** directory will be removed as well. This option should only be used after all other hosts in the C2 System have replaced their administrative host with another system or removed the C2 configuration files as well.

The entries for the system initialization scripts in **/etc/inittab** will also be removed, and rebooting this system will result in the system not being configured for C2 mode.

Executing this command in multi-user mode will result in the user definitions from the C2 System being retained. Executing this command in single-user mode will result in the user definitions from the C2 System being removed and the root user being the only valid user ID.

The system should be rebooted immediately after executing this command so that the changes may take effect.

Flags

Item	Description
-m	The host was configured as the administrative master

Exit Status

0
The C2 System administrative host information has been successfully removed.

1
The system was not configured to operate in C2 mode.

2
The system was not installed with the C2 option.

3
An error occurred removing the C2 System administrative host information.

4
An invalid command line option was used.

Files

Item	Description
/usr/sbin/rmC2admin	Contains the **rmC2admin** command.

rmCCadmin Command

Purpose

Remove the configuration files for a distributed Common Criteria enabled System host.

Syntax

rmCCadmin [**-m**]

Description

The **rmCCadmin** command replaces the distributed Common Criteria enabled System symbolic links with the actual files. The directory **/etc/data.shared** will be removed. When the **-m** flag is used, the **hd10sec** file system and **/etc/data.master** directory will be removed as well. This option should only be used after all other hosts in the Common Criteria enabled System have replaced their administrative host with another system or removed the Common Criteria enabled configuration files as well.

The entries for the system initialization scripts in **/etc/inittab** will also be removed, and rebooting this system will result in the system not being configured for Common Criteria enabled mode.

Executing this command in multi-user mode will result in the user definitions from the Common Criteria enabled System being retained. Executing this command in single-user mode will result in the user definitions from the Common Criteria enabled System being removed and the root user being the only valid user ID.

The system should be rebooted immediately after executing this command so that the changes may take effect.

Flags

Item	Description
-m	The host was configured as the administrative master

Exit Status

0
The Common Criteria enabled System administrative host information has been successfully removed.

1
The system was not configured to operate in Common Criteria enabled mode.

2
The system was not installed with the Common Criteria enabled option.

3
An error occurred removing the Common Criteria enabled System administrative host information.

4
An invalid command line option was used.

Files

Item	Description
/usr/sbin/rmCCadmin	Contains the **rmCCadmin** command.

rmccli information file

Purpose

Provides general information about resource monitoring and control (RMC) and related commands.

Description

The general information about RMC and related commands, including data types, terminology, and references to related information follows.

Command structure and use

The RMC commands might be grouped into categories that represent the different operations that can be run on resource classes and resources:

- Creating and removing resources: **mkrsrc**, **rmrsrc**
- Modifying resources: **chrsrc**, **refrsrc**
- Viewing definitions and data: **lsrsrc**, **lsrsrcdef**
- Viewing actions: **lsactdef**
- Running actions: **runact**

The RMC commands can be run directly from the command line or called by user-written scripts. In addition, the RMC commands are used as the basis for higher-level commands, such as the event response resource manager (ERRM) commands.

Data display information

The flags that control the display function for the RMC CLI routines, in order of precedence are:

1. `-l` for long display. This flag is the default display format.

   For example, the command:

   ```
   lsrsrc -s 'Name == "c175n05"' IBM.Foo Name NodeList SD Binary RH Int32Array
   ```

   produces the following output:

```
Persistent Attributes for Resource: IBM.Foo
resource 1:
        Name       = "c175n05"
        NodeList   = {1}
        SD         = ["testing 1 2 3",1,{0,1,2}]
        Binary     = "0xaabbcc00 0xeeff"
        RH         = "0x0000 0x0000 0x00000000 0x00000000 0x00000000 0x00000000"
        Int32Array = {1,5,-10,1000000}
```

2. `-t` for tabular display.

   For example, the command:

   ```
   lsrsrc -s 'Name ?= "Page"' -t IBM.Condition Name EventExpression
   ```

   produces the following output:

   ```
   Persistent Attributes for Resource: IBM.Condition

   Name                  EventExpression
   "Page space out rate" "VMPgSpOutRate > 500"
   "Page fault rate"     "VMPgFaultRate > 500"
   "Page out rate"       "VMPgOutRate > 500"
   ```

```
"Page in rate"        "VMPgInRate > 500"
"Page space in rate"  "VMPgSpInRate > 500"
```

3. –x for suppressing headers when printing.
4. –d for colon (:) delimited display.

   For example, the command:

   ```
   lsrsrc -xd -s 'Name == "c175n05"' IBM.Foo Name Int32 Uint32Array SD Binary
   ```

   produces the following output:

   ```
   c175n05:-100:{}:["hel  lo1",1,{0,1,2}]:"0xaabbcc00 0xeeff":
   ```

   Note the use of the –x flag along with the –d flag.
5. –D*delimiter* for string-delimited display.

   For example, the command:

   ```
   lsrsrc -xD:: -s 'Name == "c175n05"' IBM.Foo Name Int32 Uint32Array SD Binary
   ```

   produces the following output:

   ```
   c175n05::-100::{}::["hel  lo1",1,{0,1,2}]::"0xaabbcc00 0xeeff"::
   ```

   Note the use of the –x flag along with the –D*Delimiter* flag.

When output of any list command **lsrsrc** **lsrsrcdef** is displayed in the tabular output format, the printing column width might be truncated. If more characters need to be displayed (as in the case of strings) use the –l flag to display the entire field.

Data input formatting

Binary data for attributes of binary type can be entered in the following formats:

- "0x*nnnnnnnn* 0x *nnnnnnnn* 0x *nnnn*..."
- "0x*nnnnnnnnnnnnnnnnnnn*..."
- 0x *nnnnnnnnnnnnnnnn*...

Integer data for attributes of one of the integer types can be entered as:

- A decimal constant that begins with a non-zero digit (`Int32=45`, for example)
- An octal constant that begins with a prefix of `0`, which is optionally followed by a combination of decimal numbers in the range `0` to `7` (`Int32=055`, for example)
- A hexadecimal constant that begins with a prefix of `0x` or `0X` followed a combination of decimal numbers in the range `a` to `f` and `A` to `F` (`Int32=0x2d`, for example)

Be careful when you specify strings as input data. Strings that contain:

- No white space or non-alphanumeric characters can be entered as input without enclosing quotation marks
- White space or other alphanumeric characters must be enclosed in quotation marks
- Single quotation marks (`'`) must be enclosed by double quotation marks (`"`), as shown in this example: `"this is a string with 'single quotation marks'"`

Selection strings must be enclosed in double quotation marks, unless the selection string itself contains double quotation marks, in which case the selection string must be enclosed in single quotation marks. For information about how to specify selection strings, see the *Administering RSCT* Guide.

- Sample selection string input: `"NodeNumber == 1"`
- Selection string input where double quotation marks are part of the selection string: `'Name == "c175n05"'`

Structured data (SD) types must be enclosed in square brackets: `[hello,1,{2,4,6,8}]`

When structured data (SD) is supplied as command-line input to the RMC commands, enclose the SD in single quotation marks: `SD='[hello,1,{2,4,6,8}]'`

Arrays of any type must be enclosed in braces {}:

- Array of integers: `{-4, -3, -2, -1, 0, 1, 2, 3, 4}`
- Array of strings: `{abc, "do re mi", 123}`
- Array of structured data: `{[hello,1,{0,1,2,3}],[hello2,2,{2,4,6,8}]}`

Arrays of any type with more than one element must be enclosed in quotation marks. For example:

- **mkrsrc** IBM.Foo Name=testing NodeList={1} Uint32Array='{1,2,3}'
- **mkrsrc** IBM.Foo Name=testing NodeList='{1}' Uint32_array='{1,2,3}'

Arrays of strings and arrays of structured data must always be enclosed in quotation marks.

When arrays of structured data or arrays that contain strings enclosed in quotation marks are supplied as command-line input to the RMC commands, enclose the entire array in single quotation marks:

- Array of strings: `mkrsrc IBM.Foo Name="c175n05" NodeList={1} StringArray='{"a string","a different string"}'`
- Array of structured data: `mkrsrc IBM.Foo Name="c175n05" NodeList={1} SDArray='{["string 1",1,{1,1}],["string 2",2,{1,2,3}]}'`

For more examples, see the `resource_data_input`.

Data output formatting

String data is always displayed in either double or single quotation marks as:

- A description attribute that equals the string "This is a string that contains white space" is displayed in the long format as:

```
Description = "This is a string that contains white space"
```

- A description attribute value that equals an empty string "" is displayed in long format as:

```
Description = ""
```

- A description attribute value that equals a string that contains a new-line character at the end of the string is displayed in long format as:

```
Description = "This string ends with a new-line character..."
```

- A selection string that contains double quotation marks is displayed in long format as:

```
SelectionString = 'Name == "c175n05"'
```

- A name attribute value that equals the string "c175n05" is displayed in long format as:

```
Name = "c175n05"
```

Binary data is displayed as follows:

```
"0x nnnnnnnn 0x nnnnnnnn 0x nnnnnnnn 0x nnnnnnnn"
```

Naming conventions

The following variable names are used throughout the RMC command man pages:

Variable	Description
attr	The name of a resource class or a resource attribute
resource_class	The name of a resource class

Node groups

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and by using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

Terminology

attribute

Attributes are either persistent or dynamic. A resource class is defined by a set of persistent and dynamic attributes. A resource is also defined by a set of persistent and dynamic attributes. Persistent attributes define the configuration of the resource class and resource. Dynamic attributes define a state or a performance-related aspect of the resource class and resource. In the same resource class or resource, an attribute name can be specified as either persistent or dynamic, but not both.

resource

An entity in the system that provides a set of services. Examples of hardware entities are processors, disk drives, memory, and adapters. Examples of software entities are database applications, processes, and file systems. Each resource in the system has one or more attributes that define the state of the resource.

resource class

A broad category of system resource, for example: node, file system, adapter. Each resource class has a container that holds the functions, information, dynamic attributes, and conditions that apply to that resource class. For example, the "`/tmp space used`" condition applies to a file system resource class.

resource manager

A process that maps resource and resource-class abstractions into calls and commands for one or more specific types of resources. A resource manager can be a stand-alone daemon, or it can be integrated into an application or a subsystem directly.

To see all of the resource classes that are defined in the system, run the **lsrsrc** command without any flags or parameters. To see all of the resources that are defined in the system for the `IBM.FileSystem` resource class, enter:

```
lsrsrc IBM.FileSystem
```

selection string

Must be enclosed within either double or single quotation marks. If the selection string contains double quotation marks, enclose the entire selection string in single quotation marks, for example:

```
-s 'Name == "testing"'
-s 'Name ?= "test"'
```

Only persistent attributes can be listed in a selection string.

Flags

-h

Writes the command usage statement to standard output.

-T

Writes the command trace messages to standard error. For your software service organization use only.

-V

Writes the command verbose messages (if there are any available) to standard output.

All RMC commands include a `-T` flag and a `-V` flag. Use the `-T` flag only when your software service organization instructs you to turn on tracing. Trace messages are not translated. Use the `-V` flag, which indicates "verbose" mode, to see more information about the command. Verbose messages (if there are

any available) are contained in message catalogs and are translated based on the locale in which you are running and other criteria.

Environment variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. The CT_IP_AUTHENT environment variable is valid, if the CT_CONTACT environment variable is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Standard output

When the -h flag is specified, this command usage statement is written to standard output. When the -V flag is specified, these command verbose messages (if there are any available) are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
No resources were found that match the specified selection string.

Security

Permissions are specified in the access control list (ACL) file on the contacted system.

Implementation specifics

This information is part of the `rsct.core.rmc` fileset for AIX and `rsct.core-3.1.0.0-0.`*platform*`.rpm` package for Linux, Solaris, and Windows, where *platform* is `i386`, `ppc`, `ppc64`, `s390`, or `x86_64`.

Location

/opt/rsct/man/rmccli
`/opt/rsct/man/rmccli.7` - For Linux platform.

rmcctrl Command

Purpose

Manages the resource monitoring and control (RMC) subsystem.

Syntax

`rmcctrl { -a | -A | -b || -B || -d | -k | -K | -m {R | E | D} | -M {R | E | D} | -p | -P | -q | -Q | -s | -t` *n* `| -T | -u` *n* `| -U | -v` *n* `| -V | -w n | -W | -x | -X | -z | -h }`

Description

The `rmcctrl` command controls the operation of the resource monitoring and control (RMC) subsystem. The subsystem is under the control of the system resource controller (SRC) with a subsystem name of `ctrmc` and a subsystem group name of `rsct`. The RMC subsystem definition is added to the subsystem object class and then started when Reliable Scalable Cluster Technology (RSCT) is installed. In addition, an entry is made in the `/etc/inittab` file so that the RMC subsystem is started automatically when the system is started.

Note: While the RMC subsystem can be stopped and started by using the `stopsrc` and `startsrc` commands, you can use the `rmcctrl` command to perform these functions.

Flags

-a
Adds the RMC subsystem to the subsystem object class and places an entry at the end of the `/etc/inittab` file.

-A
Adds and starts the RMC subsystem.

-b
Sets the idle timeout for the RMC API client session to n seconds. If the RMC daemon finds no activity in the session for the last n seconds, it is closed.

-B
Sets the idle timeout for the RMC API client session to a default value of 0 seconds (that is it is disabled).

-d
Deletes the RMC subsystem from the subsystem object class and removes the RMC entry from the `/etc/inittab` file.

-k
Stops the RMC subsystem.

-K
Stops the RMC subsystem and all resource managers.

-m
Specifies the RMC subsystem client message policy. This policy applies to messages sent between the RMC subsystem and any command that is listed in the *RSCT: Technical Reference,* when the command is run on a different node than the RMC subsystem (in other words, the CT_CONTACT environment variable is set). These messages are sent by using TCP/IP.

This flag is supported on RSCT version 2.3.1.0 or later. The "Enabled" policy must be used if the commands are from an earlier version of RSCT.

R
Indicates that the client message policy is "Required". "Required" means that the connection remains open only if message authentication can (and will) be used.

E
Indicates that the client message policy is "Enabled". "Enabled" is the default; message authentication is used if both sides of the connection support it.

D
Indicates that the client message policy is "Disabled". "Disabled" means that message authentication is not used.

-M
Specifies the RMC subsystem daemon message policy. This policy applies to messages sent between the RMC subsystem daemons within a management domain cluster. These messages are sent by using the User Datagram Protocol (UDP).

This flag is supported on RSCT release 2.4.1.0 or later. When specified, the indicated message policy takes effect the next time the RMC subsystem is started.

R
Indicates that the daemon message policy is "Required". "Required" means that two daemons communicate only if message authentication can (and will) be used.

E
Indicates that the daemon message policy is "Enabled". "Enabled" is the default; message authentication is used if the sending and receiving daemons support it.

D
Indicates that the daemon message policy is "Disabled". "Disabled" means that message authentication is not used. Disabling message authentication can result in the loss of function if all of the nodes in the cluster are not configured the same.

-p
Enables remote client connections.

-P
Disables remote client connections.

-q
Enables remote client connections the next time the RMC subsystem is started.

-Q
Disables remote client connections the next time the RMC subsystem is started.

-s
Starts the RMC subsystem.

-t *n*
Sets the client message timeout value to *n* seconds. This timeout value must include the following actions:

- Receiving the first message of the start session protocol after the RMC subsystem accepts a client connection.
- Receiving the complete client message by the RMC subsystem, after the initial message is received

If either of these time limits is exceeded, the client session is closed. The minimum acceptable value is 10; the maximum is 86400.

When specified, this value takes effect the next time the RMC subsystem is started.

-T
Sets the client message timeout to the default value of 10 seconds.

When specified, this value takes effect the next time the RMC subsystem is started.

-u *n*
Sets the start session timeout value to *n* seconds. Within this amount of time, the start session processing must complete for a new client session; otherwise, the session is closed. The minimum acceptable value is 60; the maximum is 86400.

When specified, this value takes effect the next time the RMC subsystem is started.

-U
Sets the start session timeout value to the default value of 300 seconds.

When specified, this value takes effect the next time the RMC subsystem is started.

-v *n*
Sets the first command timeout value to *n* seconds. If a first command timer is set when a client session is established with the RMC subsystem, the first command must arrive within the specified number of seconds after the start session processing completes; otherwise, the session is closed. The minimum acceptable value is 10; the maximum is 86400.

When specified, this value takes effect the next time the RMC subsystem is started.

-V
Sets the first command timeout value to the default value of 10 seconds.

When specified, this value takes effect the next time the RMC subsystem is started.

-w *n*
Sets the first command threshold value to *n* client sessions. Once the number of client sessions exceeds this value, the RMC subsystem enables a first command timer on each new, unauthenticated session. If the threshold is set to 0, the first command timeout function is disabled. The maximum value is 150.

When specified, this value takes effect the next time the RMC subsystem is started.

-W
Sets the first command threshold value to the default value of 150 client sessions.

When specified, this value takes effect the next time the RMC subsystem is started.

-x
Enables first command timeouts for non-`root` authenticated client sessions and for unauthenticated client sessions.

When specified, this value takes effect the next time the RMC subsystem is started.

-X
Disables first command timeouts for non-root authenticated sessions.

When specified, this value takes effect the next time the RMC subsystem is started.

-z
Stops the RMC subsystem and all resource managers, but the command does not return until the RMC subsystem and the resource managers are stopped.

-h
Writes the command's usage statement to standard output.

Security

Privilege control: only the root user must run (x) access to this command.

Exit Status

0
The command is successful.

1
The command was not successful.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output.

Examples

1. To add the RMC subsystem, enter:

```
rmcctrl -a
```

2. To start the RMC subsystem, enter:

```
rmcctrl -s
```

3. To stop the RMC subsystem, enter:

```
rmcctrl -k
```

4. To delete the RMC subsystem, enter:

```
rmcctrl -d
```

Location

/opt/rsct/bin/rmcctrl

rmcdomainstatus Command

Purpose

Displays the status of the node in management domain and peer domain.

Syntax

```
rmcdomainstatus -s ctrmc [-a ip|IP]
```

Description

When you run the **rmcdomainstatus** command in a node, the output displays the node status in the management domain and peer domain that contains the node. If the output is not displayed, the node is not a member of any peer domain or management domain.

The output of the **rmcdomainstatus** command is displayed in the following format:

```
Domain status
<Token 1 of node status> <Token 2 of node status> <Node ID> <Internal node number> <Node name
| IP address | IP address of the specified MCP> <PD_name>/<PD_status> (n)
```

The following information fields are displayed in the **rmcdomainstatus** command output:

Domain status
Displays the current state of domain. The domain status can be displayed in the following ways:

Peer Domain Status
Displays the current state of peer domain.

Management Domain Status: Managed Nodes
Displays the current state of all the managed nodes that are managed by the node in the management domain.

Management Domain Status: Management Control Points (MCP)
Displays the current state of all the Management Control Points (MCPs) that are managing the node in the management domain.

Note: The output might contain more than one section depending on the current state of the node. That is, if the node is a member of both peer domain and management domain, the output contains two separate sections of information.

Token 1 of node status
Specifies the node status that indicates one of the following conditions:

S
Indicates that text in the output is for current node in the peer domain.

I
In a management domain, this value indicates that the node is in the Up state, which is determined by the Resource Monitoring and Control (RMC) heartbeat mechanism. In a peer domain, this value indicates that the RMC daemon in the specified node is a member of the `rmc_peers` Group Services group and the node is online in the peer domain.

i
In a management domain, this value indicates that the node is in the `Pending Up` state. Communication is established between two RMC daemons but the initial handshake is not completed.

Note: The `i` token is displayed only for management domains.

O
In a management domain, this value indicates that the node is in the Down state, which is determined by the RMC heartbeat mechanism. In a peer domain, this value indicates that the RMC daemon in the specified node is no longer a member of the `rmc_peers` Group Services group.

X
In a management domain, this value indicates that a communication problem is discovered, and the RMC daemon has suspended communication with the RMC daemon that is in the specified node.

Z
Indicates that the RMC daemon has suspended communication with the RMC daemon that is in the specified node because the Up or Down state of the node is changing quickly.

Token 2 of node status
Specifies the node status that indicates one of the following conditions:

S
Indicates that text in the output is for current node in the peer domain.

A
Indicates that the messages are not queued for the specified node.

a
Indicates the same meaning as the A value, except that the specified node is running a version of the RMC daemon that is at a lower level than the local RMC daemon.

R
Indicates that the messages are queued for the specified node.

r
Indicates the same meaning as the R value, except that the specified node is running a version of the RMC daemon that is at a lower level than the local RMC daemon.

Node ID
Specifies the 64-bit node ID that is created when RSCT is installed on the node.

Internal node number
Specifies the internal node number that is used by the RMC daemon.

Node name or IP address
Specifies the name of the node that is identified by the RMC subsystem.

Note: This value is displayed only if the node is a member of a peer domain or a management domain.

IP address of the specified MCP
Specifies the first configured IP address of the specified MCP.

Note: This value is displayed only if the node is an MCP.

PD_name/PD_status (n)
Specifies the peer domain name and peer domain status as received from the managed node when the **-a** flag is not used.

Note: This value is displayed only if the node is an MCP.

The *PD_name* attribute is the name of the peer domain of which the managed node is an online member. The *PD_status* attribute is the status of the peer domain.

If the managed node is offline, the *PD_name*/*PD_status* attributes are set as `!`/`-`, and the *(n)* attribute is not present. If the peer domain status is received from the managed node, the *PD_name* attribute is set as +. The *n* attribute is the number of online nodes in the peer domain of which the specified managed node is a member.

Flags

-s ctrmc
Specifies the RSCT daemon name. For RMC, the RSCT daemon name is `ctrmc`.

-a IP|ip
Lists the IP addresses that are configured on the node. The valid values that can be specified with the **-a** flag are as follows:

IP
Lists all the configured and harvested IP addresses.

ip
Lists the IP addresses that are configured in the `ctrmc.srcntbl` file (for peer domain) and in the `ctrmc.mntbl` or `ctrmc.mcptbl` file (for management domain).

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX, Linux, and other operating systems.

Location
/opt/rsct/bin/rmcdomainstatus

Examples

1. To check the node status in the peer domain and management domain, run the following command:

   ```
   rmcdomainstatus -s ctrmc
   ```

   If the output is not displayed, the node is not a member of a peer domain or a management domain.

   If the node is a member of a peer domain, an output that is similar to the following example is displayed:

   ```
   Peer Domain Status
      I A  0x09898b3065189db6  0002  test1.ppd.pok.ibm.com
      S S  0x07e7287425d0becd  0001  test2.ppd.pok.ibm.com
   ```

   If the node is an MCP, an output that is similar to the following example is displayed:

   ```
   Management Domain Status: Managed Nodes
      I a  0xbf1fb04e5b7d0b06  0001  test1 !/+
      I a  0x3a75dd6c235c428e  0002  test2  masMMtest/+ (1)
      I A  0x07e7287425d0becd  0003  test3  masfive/+ (2)
      I A  0x09898b3065189db6  0004  test4  masfive/+(2)
   ```

   If the node is a managed node, an output that is similar to the following example is displayed:

   ```
   Management Domain Status: Management Control Points
      I A  0xef889c809d9617c7 0001  9.xx.xx.xxx
   ```

2. To display the configured and harvested IP addresses in the current node status, run the following command:

   ```
   rmcdomainstatus -s ctrmc -a IP
   ```

   An output that is similar to the following example is displayed:

   ```
   Peer Domain Status
     I A  0x4313b01f7aae13d9  0002  myrsct1.in.ibm.com
     S S  0xa15313e0cc675d54  0001  myrsct2.in.ibm.com

   Management Domain Status: Management Control Points
     I A  0x128a32b77a5d91cb  0001  10.xx.xx.xx (C)
   ```

rmcifscred Command

Purpose

Removes the CIFS credentials stored in the /etc/cifs_fs/cifscred file for the specified server and user entry.

Syntax

rmcifscred -h *RemoteHost* -u *user*

Description

The rmcifscred command takes a server and user name as input. If this input has credentials listed in /etc/cifs_fs/cifscred, the credentials are removed. Subsequent mounting to the specified server by the specified user requires manually inputting the password.

Flags

Item	Description
-h *RemoteHost*	Specifies the name of the remote host (CIFS server). This can be provided as a host name, an IP address, or as a fully qualified domain name.
-u *user*	Specifies the user name whose credentials for the specified server are to be removed from the `cifscred` file.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Examples

1. To remove the credentials stored in `/etc/cifs_fs/cifscred` for `user1` to mount on `server1`, enter:

```
rmcifscred -h server1 -u user1
```

Location

`/usr/sbin/rmcifscred`

Files

Item	Description
`/etc/cifs_fs/cifscred`	Stores the CIFS credentials.

rmcifsmnt Command

Purpose

Removes a CIFS mount from the `/etc/filesystems` file and unmounts the entry if it is mounted.

Syntax

`rmcifsmnt -f` *MountPoint* [`-B` | `-N`]

Description

The `rmcifsmnt` command removes a CIFS entry from `/etc/filesystems`. If the entry is mounted, the `rmcifsmnt` command then unmounts it.

Flags

Item	Description
-B	Removes the corresponding entry from the `/etc/filesystems` file, and unmounts the file system. This is the default.

Item	Description
`-f` *MountPoint*	Specifies the path name of the CIFS mount.
`-N`	Unmounts the file system, but does not remove the entry from the `/etc/filesystems` file.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Examples

1. To remove the CIFS mount that is mounted over `/mnt` and unmount it, enter:

```
rmcifsmnt -f /mnt
```

Location

`/usr/sbin/rmcifsmnt`

Files

Item	Description
`/etc/filesystems`	Stores the CIFS entry.

rmclass Command

Purpose

Remove a Workload Management class.

Syntax

rmclass [**-d** *Config_Dir*] [**-S** *SuperClass*] *Name*

Description

The **rmclass** command removes the superclass or the subclass identified by the *Name* parameter from the class definition file, the class limits file and the class shares file. The class must already exist. The predefined **Default**, **System**, and **Shared** classes cannot be removed.

In addition, when removing a superclass **Super**, the directory **/etc/wlm/***Config_Dir***/Super** and all the WLM property files it contains (if they exist) are removed. Removing a superclass fails if any user created subclass still exists (subclass other than **Default** and **Shared**).

> **Note:** Only root can remove a superclass. Only root or authorized users whose user ID or group ID matches the user name or group name specified in the attributes **adminuser** and **admingroup** of a superclass can remove a subclass of this superclass.

Normally, **rmclass** deletes the class and its attributes in the relevant WLM property files, and the modifications are applied to the in-core class definitions (active classes) only after an update of WLM using the **wlmcntrl** command.

If an empty string is passed as the configuration name (*Config_dir*) with the **-d** flag, the class is deleted only in the WLM in-core data structures, and no property file is updated. So, if the class is still defined in a WLM configuration, it is recreated after an update or restart of WLM. This flag should be mainly used to remove classes dynamically created in the in-core WLM data structures only by applications using the WLM API, for example, to do some cleanup after application failure.

Note: This command cannot apply to a set of time-based configurations (do not specify a set with the **-d** flag). If the current configuration is a set, the **-d** flag must be given to indicate which regular configuration the command should apply to.

Flags

Item	Description
-d *Config_Dir*	Uses **/etc/wlm/***Config_dir* as alternate directory for the properties files. If this flag is not used, the configuration files in the directory pointed to by **/etc/wlm/current** are used. If an empty string is passed as the configuration name (`-d ""`) the class is deleted only in the WLM in-core data structures and no configuration file is modified.
-S *SuperClass*	Specifies the name of the superclass when removing a subclass. There are two ways of specifying the subclass **Sub** of superclass **Super**: 1. Specify the full name of the subclass as **Super.Sub** and do not use **-S**. 2. Specify the **-S** flag to give the superclass name and use the short name for the subclass: `rmclass options -S Super  Sub`

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
classes	Contains the names and definitions of the classes.
limits	Contains the resource limits.
shares	Contains the resource shares.

rmcluster Command

Purpose

Removes an existing cluster or site configuration.

Syntax

rmcluster [**-n** *clustername*] [**-S** *sitename*][**-v**]

Description

The **rmcluster** command removes the cluster configuration or one of the sites in the cluster. The repository disk and the shared disks of the SAN Volume Controller (SVC) that are associated with the entity that must be removed are released.

When a site is removed from the cluster, the repository and the shared disks that are used by the site are released. Releasing the disks does not cause the site to be removed. When a cluster is removed, all the repository and shared disks are released.

Note: A site cannot remove itself. Sites can only be removed from a node in a different site.

Flags

Item	Description
-n *clustername*	Specifies the name of the cluster to be removed.
-S *sitename*	Specifies the name of the site to be removed.
-v	Specifies the verbose mode.

Examples

1. To remove the cluster configuration, enter the following command:

   ```
   rmcluster -n mycluster
   ```

2. To remove a site named `mysite` from the cluster, enter the following command on a node in a different site:

   ```
   rmcluster -S mysite
   ```

rmcomg Command

Purpose

Removes a communication group that has already been defined from a peer domain.

Syntax

`rmcomg` [`-q`] [`-h`] [`-TV`] *communication_group*

Description

The `rmcomg` command removes the definition of the existing communication group with the name specified by the *communication_group* parameter for the online peer domain. The communication group is used to define heartbeat rings for use by topology services and to define the tunables for each heartbeat ring. The communication group determines which devices are used for heartbeating in the peer domain.

The `rmcomg` command must be run on a node that is currently online in the peer domain where the communication group is defined. More than half of the nodes must be online to remove a communication group from the domain.

The communication group must not be referred to by an interface resource. Use the `chcomg` command to remove references made by interface resources to a communication group.

Flags

-q
Specifies quiet mode. The command does not return an error if the communication group does not exist.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

communication_group
Specifies the name of the defined communication group that is to be removed from the peer domain.

Security

The user of the `rmcomg` command needs write permission for the `IBM.CommunicationGroup` resource class. By default, `root` on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

6
The communication group does not exist.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is defined and online to the peer domain where the communication group is to be removed.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the `-f "-"` or `-F "-"` flag is specified, this command reads one or more node names from standard input.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

In this example, `nodeA` is defined and online to `ApplDomain`. To remove the communication group definition `ComGrp1` for the peer domain `ApplDomain`, run this command on `nodeA`:

```
rmcomg ComGrp1
```

Location

/opt/rsct/bin/rmcomg

rmcondition Command

Purpose

Removes a condition.

Syntax

`rmcondition` [`-f`] [`-q`] [`-h`] [`-TV`] *condition*[`:`*node_name*]

Description

The `rmcondition` command removes the condition specified by the *condition* parameter. The condition must already exist to be removed. When the condition must be removed even if it has linked responses, use the `-f` flag to force the condition and the links with the responses to be removed. If the `-f` flag is not specified and links with responses exist, the condition is not removed. This command does not remove responses.

If a particular condition is needed for system software to work properly, it may be locked. A locked condition cannot be modified or removed until it is unlocked. If the condition you specify on the `rmcondition` command is locked, it will not be removed; instead an error will be generated informing you that the condition is locked. To unlock a condition, you can use the `-U` flag of the `chcondition` command. However, since a condition is typically locked because it is essential for system software to work properly, you should exercise caution before unlocking it.

Flags

-f
Forces the condition to be removed even if it is linked to responses. The links with the responses are removed as well as the condition, but the responses are not removed.

-q
Does not return an error when *condition* does not exist.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

condition
Specifies the name of a condition to be removed.

node_name
Specifies the node where the condition is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

Security

The user needs write permission for the `IBM.Condition` resource class to run `rmcondition`. Permissions are specified in the access control list (ACL) file on the contacted system.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which

the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To remove the condition definition named "FileSystem space used", run this command:

```
rmcondition "FileSystem space used"
```

2. To remove the condition definition named "FileSystem space used" even if the condition is linked with responses, run this command:

```
rmcondition -f "FileSystem space used"
```

This example applies to management domains:

1. In this example, the current node is the management server. To remove the condition definition named "nodeB FileSystem space used" that is defined on managed node nodeB, run this command:

```
rmcondition "FileSystem space used:nodeB"
```

This example applies to peer domains:

1. To remove the condition definition named "nodeA FileSystem space used" that is defined on node nodeA, run this command from any node in the domain:

```
rmcondition "nodeA FileSystem space used:nodeA"
```

Location

/opt/rsct/bin/rmcondition

rmcondresp Command

Purpose

Deletes the link between a condition and one or more responses.

Syntax

To delete the link between a condition and one or more responses:

`rmcondresp` [`-q`] [`-h`] [`-TV`] *condition*[`:`*node_name*] [*response* [*response*...]]

To delete all of the links to one or more responses:

`rmcondresp` [`-q`] `-r` [`-h`] [`-TV`] *response1* [*response2*...][`:`*node_name*]

To lock or unlock the condition/response association:

`rmcondresp` {`-U` | `-L`} [`-h`] [`-TV`] *condition*[`:`*node_name*] *response*

Description

The `rmcondresp` command deletes the link between a condition and one or more responses. A link between a condition and a response is called a *condition/response association*. The response is no longer run when the condition occurs. Use the `-r` flag to specify that the command parameters consist only of responses. This deletes all links to conditions for these responses. If only a condition is specified, links to all responses for that condition are deleted.

If a particular condition/response association is needed for system software to work properly, it may be locked. A locked condition/response association cannot be removed by the `rmcondresp` command. If the condition/response association you specify on the `rmcondresp` command is locked, it will not be removed; instead an error will be generated informing you that this condition/response association is locked. To unlock a condition/response association, you can use the `-U` flag. However, because a condition/response association is typically locked because it is essential for system software to work properly, you should exercise caution before unlocking it.

Flags

-q
Does not return an error when either *condition* or *response* does not exist.

-r
Indicates that all command parameters are responses. There are no conditions specified. This command removes condition/response associations from all conditions that are linked to the specified responses.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

-U
Unlocks a condition/response association so it can be started, stopped, or removed. If a condition/response association is locked, this is typically because it is essential for system software to work properly. For this reason, you should exercise caution before unlocking it. When unlocking a condition/response association using the `-U` flag, no other operation can be performed by this command.

-L
Locks a condition/response association so it cannot be started, stopped, or removed. When locking a condition/response association using the `-L` flag, no other operation can be performed by this command.

Parameters

condition
Specifies the name of the condition linked to the response. The condition is always specified first unless the `-r` flag is used.

response
Specifies the name of a response or more than one response. The links from the specified responses to the specified condition are removed.

node_name
Specifies the node where the condition is defined. If the `-r` flag is used, it is the node where the response is defined. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

Security

The user needs write permission for the `IBM.Association` resource class to run `rmcondresp`. Permissions are specified in the access control list (ACL) file on the contacted system.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To delete the link between the condition "FileSystem space used" and the response "Broadcast event on-shift", run this command:

```
rmcondresp "FileSystem space used" "Broadcast event on-
shift"
```

2. To delete the links between the condition "FileSystem space used" and all of its responses, run this command:

```
rmcondresp "FileSystem space used"
```

3. To delete the links between the condition "FileSystem space used" and the responses "Broadcast event on-shift" and "E-mail root anytime", run this command:

```
rmcondresp "FileSystem space used" "Broadcast event on-shift" "E-mail root
anytime"
```

4. To delete the links between the response "Broadcast event on-shift" and all of the conditions that use it, run this command:

```
rmcondresp -r "Broadcast event on-shift"
```

These examples apply to management domains:

1. To delete the link between the condition "FileSystem space used" on the management server and the response "Broadcast event on-shift", run this command on the management server:

```
rmcondresp "FileSystem space used" "Broadcast event on-
shift"
```

2. To delete the links between the condition "FileSystem space used" on the managed node nodeB and the responses "Broadcast event on-shift" and "E-mail root anytime", run this command on the management server:

```
rmcondresp  "FileSystem space used":nodeB \
"Broadcast event on-shift" "E-mail root
anytime"
```

These examples apply to peer domains:

1. To delete the links between the condition "FileSystem space used" on nodeA in the domain and the responses "Broadcast event on-shift" and "E-mail root anytime", run this command on any node in the domain:

```
rmcondresp "FileSystem space used":nodeA \
"Broadcast event on-shift" "E-mail root
anytime"
```

2. To delete the links between all conditions on nodeA in the domain and the response "Broadcast event on-shift", run this command on any node in the domain:

```
rmcondresp -r "Broadcast event on-
shift":nodeA
```

Location

/opt/rsct/bin/rmcondresp

rmcosi Command

Purpose

Removes a Common Operating System Image (COSI).

Syntax

rmcosi [**-f**] [**-v**] *COSI*

Description

The `rmcosi` command removes a Common Operating System Image (COSI) created with the `mkcosi` command. If the common image to be removed is being used by thin servers, the operation fails unless the force flag (`-f`) is specified. The `-f` flag terminates any thin server sessions with the common image so that the COSI can be removed. This command depends on the `bos.sysmgt.nim.master` fileset being present on the system.

Flags

Item	Description
`-f`	Forces the removal of the common image. If the common image is being used by thin servers, the thin servers will be taken offline so that the common image can be removed.
`-v`	Enables verbose debug output when the `rmcosi` command runs.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

Access Control: You must have root authority to run the `rmcosi` command.

Examples

1. To common image named `cosi1`, enter:

```
rmcosi cosi1
```

Location

`/usr/sbin/rmcosi`

Files

Item	Description
`/etc/niminfo`	Contains variables used by NIM.

rmdel Command

Purpose

Removes a delta from a SCCS file.

Syntax

rmdel -r *SID File ...*

Description

The **rmdel** command removes the delta specified by the *SID* variable from each Source Code Control System (SCCS) file indicated in the *File* parameter. You can remove only the most recently created delta in a branch, or the latest trunk delta if it has no branches. In addition, the SID you specify must not be a version currently being edited for the purpose of making a delta. To remove a delta, you must either own the SCCS file and the directory, or you must have created the delta you want to remove.

If you specify a directory for the *File* parameter, the **rmdel** command performs the requested actions on all SCCS files (those with the **s.** prefix). If you specify a **-** (dash) for the *File* parameter, the **rmdel** command reads standard input and interprets each line as the name of an SCCS file. The **rmdel** command continues to read input until it reaches an end-of-file character.

After a delta has been removed, it is not included in any g-file created by the **get** command. However, the delta table entry remains in the **s.** file with an **R** by the entry to show that the delta has been removed.

Flags

Item	Description
-r *SID*	Removes the specified delta *SID* from the SCCS file. This flag is required.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

To remove delta 1.3 from the **s.test.c** SCCS file, type:

```
rmdel -r 1.3 s.test.c
```

Files

Item	Description
/usr/bin/rmdel	Contains the **rmdel** command.
s.*files*	Files processed by the **rmdel** command.

rmdev Command

Purpose

Removes a device from the system.

Syntax

rmdev { **-l** | **-p** }*Name* [**-d** | **-S**] [**-f** *File*] [**-h**] [**-q**] [**-R**] [**-g**]

Description

Note: The **-l** flag cannot be specified if **-p** is specified. If the **-R** flag is specified along with the **-p** flag, it will be ignored.

The **rmdev** command unconfigures or both unconfigures and undefines the device specified with the device logical name using the **-l** *Name* flag. The default action unconfigures the device but retains its device definition in the Customized Devices object class.

If you specify the **-S** flag, the **rmdev** command sets the device to the Stopped state for devices that support the Stopped state. If you specify the **-d** flag, the **rmdev** command deletes the device definition from the Customized Devices object class (undefines). If you do not specify the **-d** flag, the **rmdev** command sets the device to the Defined state (unconfigures). If you specify the **-R** flag, the **rmdev** command acts on any children of the device as well.

Use the **-p** flag along with the parent device's logical name to unconfigure or delete all of the children devices. The children are unconfigured or deleted in the same recursive fashion as described for the **-R** flag, but the specified device itself is not unconfigured or deleted.

Attention: To protect the Configuration database, the **rmdev** command is not interruptible. Stopping this command before it is complete could result in a corrupted database.

You can use the System Management Interface Tool (SMIT)**smit rmdev** fast path to run this command.

Flags

Item	Description
-d	Removes the device definition from the Customized Devices object class. This flag cannot be used with the **-S** flag.
-f *File*	Reads the necessary flags from the *File* parameter.
-g	Forces the remove operation to run on a locked device.
-h	Displays the command usage message.
-l *Name*	Specifies the logical device, indicated by the *Name* parameter, in the Customized Devices object class. This flag cannot be used with the **-p** flag.
-p *Name*	Specifies the parent logical device (indicated by the *Name* parameter) in the Customized Devices object class, with children that must be removed. This flag may not be used with the **-l** flag.
-q	Suppresses the command output messages from standard output and standard error.
-R	Specifies to unconfigure the device and its children. When used with the **-d** or **-S** flags, the children are undefined or stopped, respectively.
-S	Makes the device unavailable by calling the Stop method if the device has a Stop method. This flag cannot be used with the **-d** flag.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Privilege Control: Only the root user and members of the system group should have execute (x) access to this command.

Auditing Events:

Event	Information
DEV_Stop	Device name
DEV_Unconfigure	Device name
DEV_Remove	Device name

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To unconfigure the `cd0` CD-ROM device while retaining its device definition in the Customized Devices object class, type the following:

   ```
   rmdev  -l cd0
   ```

   The system displays a message similar to the following:

```
cd0 defined
```

2. To remove the `cd0` CD-ROM device definition from the Customized Devices object class, type the following:

```
rmdev  -d  -l cd0
```

The system displays a message similar to the following:

```
cd0 deleted
```

3. To unconfigure the `scsi1` SCSI adapter and all of its children while retaining their device definitions in the Customized Devices object class, type the following:

```
rmdev  -R -l scsi1
```

The system displays a message similar to the following:

```
rmt0 Defined
hdisk1 Defined
scsi1 Defined
```

4. To unconfigure the children of the `scsi1` SCSI adapter, but not the adapter itself, while retaining their device definitions in the Customized Devices object class, type the following:

```
rmdev  -p scsi1
```

The system displays a message similar to the following:

```
rmt0 Defined
hdisk1 Defined
```

5. To unconfigure the children of the `pci1` PCI bus and all other devices under them while retaining their device definitions in the Customized Devices object class, type the following:

```
rmdev  -p pci1
```

The system displays a message similar to the following:

```
rmt0 Defined
hdisk1 Defined
scsi1 Defined
ent0 Defined
```

Files

Item	Description
/usr/sbin/rmdev	Contains the **rmdev** command.

rmdir Command

Purpose

Removes a directory.

Syntax

rmdir [**-p**] *Directory ...*

Description

The **rmdir** command removes the directory, specified by the *Directory* parameter, from the system. The directory must be empty before you can remove it, and you must have write permission in its parent directory. Use the **ls -al** command to check whether the directory is empty. The directory must not be exported for use by the NFS version 4 server.

Note: The **rmdir** command supports the **—** (dash, dash) parameter as a delimiter that indicates the end of the flags.

Flags

Item	Description
-p*Directory*	Removes all directories along the path name specified by the *Directory* parameter. Parent directories must be empty and the user must have write permission in the parent directories before they can be removed.

Exit Status

This command returns the following exit values:

Item	Description
0	Each directory entry specified by a *Directory* parameter was removed successfully.
>0	An error occurred.

Examples

1. To empty and remove a directory, type:

   ```
   rm mydir/* mydir/.*
   rmdir mydir
   ```

   This command removes the contents of the **mydir** file and then removes the empty directory. The **rm** command displays an error message about trying to remove the directories `.` (dot) and `..` (dot, dot), and then the **rmdir** command removes them.

   Note that the **rm mydir/* mydir/.*** command first removes files with names that do not begin with a dot, and then removes those with names that do begin with a dot. You may not realize that the directory contains file names that begin with a dot because the **ls** command does not usually list them unless you use the **-a** flag.

2. To remove the **/home**, **/home/demo**, and **/home/demo/mydir** directories, type:

   ```
   rmdir -p /home/demo/mydir
   ```

   This command removes first the **/mydir** directory and then the **/demo** and **/home** directories, respectively. If a directory is not empty or does not have write permission when it is to be removed, the command terminates.

Files

Item	Description
/usr/bin/rmdir	Contains the **rmdir** command.

rmdom Command

Purpose

Removes the domains from the domain database.

Syntax

rmdom *Name*

Description

The **rmdom** command removes the domain that is identified by the *Name* parameter. The command only removes the existing domains from the domain database. A domain that is referenced by the domain object database cannot be removed until you remove the references to the domain.

When the system is operating in enhanced role-based access control (RBAC) mode, modifications made to the domains database are not used for security considerations until the database has been sent to the kernel security tables by using the **setkst** command.

Parameters

Item	Description
Name	Specifies the name of the domain to be removed.

Security

The **rmdom** command is a privileged command. You must have the following authorization to run the command:

Item	Description
aix.security.domains.remove	Required to remove the domain from the domain database.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

File Accessed

File	Mode
/etc/security/domains	rw

Examples

To remove the `hrdom` domain, type:

```
rmdom hrdom
```

rmf Command

Purpose

Removes folders and the messages they contain.

Syntax

rmf [**+** *Folder*] [**-interactive** | **-nointeractive**]

Description

The **rmf** command deletes the messages within the specified folder and then deletes the folder. By default, the **rmf** command confirms your request before deleting a folder. If the folder contains files that are not messages, the **rmf** command does not delete the files and returns an error.

Attention: The **rmf** command irreversibly deletes messages that do not have other links.

By default, the **rmf** command removes the current folder. When the current folder is removed, **inbox** becomes the current folder. If the **+***Folder* flag is not specified, and the **rmf** command cannot find the current folder, the command requests confirmation before removing the **+inbox** folder.

The **rmf** command does not delete any folder or any messages in a folder to which you have read-only access. The **rmf** command deletes only your private sequences and your current message information from the profile.

The **rmf** command does not delete folders recursively. You cannot remove subfolders by requesting the removal of a parent folder. If you remove a subfolder, the parent of that folder becomes the current folder.

Flags

Item	Description
+*Folder*	Specifies the folder to be removed.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.
-interactive	Requests confirmation before removing the folder. If the **+***Folder* flag is not specified, this is the default.
-nointeractive	Removes the folder and its messages without requesting confirmation. This is the default.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Current-Folder:`	Sets the default current folder.
`Path:`	Specifies the user's MH directory.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

Attention: The **rmf** command irreversibly deletes messages that do not have other links.

1. To remove the current folder called `status`, enter:

```
rmf
```

The system responds with a message similar to the following:

```
Remove folder "status"?
```

If you do want the folder removed, enter `yes`. The system responds with a message similar to the following:

```
[+inbox now current]
```

2. To remove the `meetings` folder noninteractively, enter:

```
rmf +meetings
```

Files

Item	Description
$HOME/.mh_profile	Defines the MH user profile.
/usr/bin/rmf	Contains the **rmf** command.

rmfilt Command

Purpose

Removes a filter rule from the filter table.

Syntax

rmfilt -v 4|**6 -n fid** | **all** [**-f**]

Description

Use the **rmfilt** command to remove filter rules from the filter rule table. Actions by this command will not effect the IP Security subsystem until the **mkfilt** command is executed. IPsec filter rules for this command can be configured by using the `genfilt` command, or IPsec smit (IP version 4 or IP version 6) in the Virtual Private Network submenu.

The **rmfilt** command removes a filter rules from the filter rule table. Only manual filter rules can be removed.

Flags

Item	Description
-f	Force to remove auto-generated filter rules. **-f** flag works with **-n all** to remove all the filter rules (user-defined and auto-generated filter rules) except rule number 1 for IP version 4.
-n	The ID of the filter rule you want to remove from the filter rule table. For IP version 4, the value of **1** is invalid for this flag, that is a reserved filter rule. If **all** is specified, all the user defined filter rules will be removed until the **-f** flag is specified.
-v	IP version of the filter rule you want to remove. Value **4** specifies IP version 4. Value **6** specifies IP version 6.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

rmfs Command

Purpose

Removes a file system.

Syntax

rmfs [**-r** | **-i**] *FileSystem*

Description

The **rmfs** command removes a file system. If the file system is a journaled file system (JFS or JFS2), the **rmfs** command removes both the logical volume on which the file system resides and the associated stanza in the **/etc/filesystems** file. If the file system is not a JFS or JFS2 file system, the command removes only the associated stanza in the **/etc/filesystems** file. The *FileSystem* parameter specifies the file system to be removed.

You could also use the System Management Interface Tool (SMIT) **smit rmfs** fast path to run this command.

Flags

Item	Description
-r	Removes the mount point of the file system.
-i	Displays warning and prompts the user before removing file system.

Exit Status

This command returns the following exit values:

Item	Description
0	File system is successfully removed.
>0	File system is not successfully removed.

Security

Access Control: Only the root user or a member of the **system** group can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To remove the **/test** file system, enter:

```
rmfs /test
```

This removes the **/test** file system, its entry in the **/etc/filesystems** file, and the underlying logical volume.

Files

Item	Description
/etc/rmfs	Contains the **rmfs** command.
/etc/filesystems	Lists the known file systems and defines their characteristics.

rmgroup Command

Purpose

Removes a group.

Syntax

rmgroup [**-p**] [**-R** *load_module*] *Name*

Description

The **rmgroup** command removes a group specified by the *Name* parameter. This command deletes all the group attributes as well. To remove a group, the group name must already exist. Users who are group members are not removed from the system.

If the group is the primary group for any user, you cannot remove it unless you redefine the user's primary group with the **chuser** command. The **chuser** command alters the **/etc/passwd** file. Only the root user or a user with GroupAdmin authorization can remove an administrative group or a group with administrative users as members.

For groups that were created with an alternate Identification and Authentication (I&A) mechanism, the **-R** flag can be used to specify the I&A load module used. Load modules are defined in the **/usr/lib/security/methods.cfg** file.

You could also use the System Management Interface Tool (SMIT) **smit rmgroup** fast path to run this command.

Flag

Item	Description
-p	Removes the group keystore.
-R *load_module*	Specifies the loadable I&A module used to remove a group.

Exit Status

This command returns the following exit values:

Item	Description
0	The command executes successfully and all requested changes are made.
>0	An error occurred. The printed error message gives further details about the type of failure.

Security

Access Control: This command should grant execute (x) access only to the root user and members of the security group. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the root user with the **setuid** (SUID) bit set.

Files Accessed:

Mode	File
r	**/etc/passwd**
rw	**/etc/group**
rw	**/etc/security/group**

Auditing Events:

Event	Information
GROUP_Remove	group

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Limitations

Removing a group may not be supported by all loadable I&A modules. If the loadable I&A module does not support removing a group, an error is reported.

Examples

1. To remove the finance group, type:

```
rmgroup finance
```

2. To remove the LDAP I&A loadable module group monsters, type:

```
rmgroup -R LDAP monsters
```

Files

Item	Description
/usr/sbin/rmgroup	Contains the **rmgroup** command.
/etc/group	Contains the basic attributes of groups.
/etc/security/group	Contains the extended attributes of groups.

rmiscsi Command

Purpose

Removes iSCSI target data.

Syntax

`rmiscsi -l` *AdapterName* [`-g` *group*] [`-t` *TargetName*] [`-n` *PortNumber*] [`-i` *IPaddress*]

Description

The `rmiscsi` command removes iSCSI target data to ODM. There are two categories of data stored in ODM. The first is for statically configured iSCSI targets, which require that all the relevant iSCSI target information (such as target name, IP address, and port number) are specified in order for AIX to discover them. The second category of iSCSI target data is for iSCSI target devices that can be configured automatically, but require authentication from the host (such as passwords). These two categories of iSCSI target data are associated with the `static` and `auto` groups, respectively, specified by the `-g` flag.

Flags

Item	Description
`-g` *group*	Specifies which group this iSCSI target is associated with. There two valid groups are `static` and `auto`. The `static` group is for iSCSI targets that cannot be automatically discovered from this host; all relevant iSCSI target information for them (such as target name, IP address, and port number) must be specified. The `auto` group is for iSCSI targets that are automatically discovered, but require authentication information such as passwords.
`-i` *IPaddress*	Specifies the IP address of the iSCSI target.
`-l` *AdapterName*	Specifies the adapter name for the iSCSI TCP/IP Offload Engine (TOE) adapter that is attached to this iSCSI target. It can also specify the iSCSI protocol device for the iSCSI software solution device.
`-n` *PortNumber*	Specifies the port number on which the iSCSI target is accessed. The default port number is `3260`.
`-t` *TargetName*	Specifies the iSCSI target name (for example, `iqn.sn9216.iscsi-hw1`).

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

The `rmiscsi` command is executable only by root.

Examples

1. To remove one statically configured iSCSI target, enter:

```
rmiscsi -l ics0 -g static -t iqn.sn1234.iscsi_hw1 -i 10.2.1.4 -n 3260
```

2. To remove all iSCSI targets for the iSCSI TOE adapter `ics0`, enter:

```
rmiscsi -l ics0
```

Location

`/usr/sbin/rmiscsi`

Files

Item	Description
`src/bos/usr/sbin/iscsia`	Contains the common source files from which the iSCSI commands are built.

rmitab Command

Purpose

Removes records in the **/etc/inittab** record. You can specify a record to remove by using the *Identifier* parameter. The *Identifier* parameter specifies a field of one to fourteen characters used to uniquely identify an object. If the *Identifier* field is not unique, the command is unsuccessful.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To remove the tty entry for tty2 , enter:

```
rmitab "tty002"
```

rmkeyserv Command

Purpose

Stops the **keyserv** daemon and comments the entry in the **/etc/rc.nfs** file.

Syntax

/usr/sbin/rmkeyserv [**-I** | **-B** | **-N**]

Description

The **rmkeyserv** command comments the entry for the **keyserv** daemon in the **/etc/rc.nfs** file. The **rmkeyserv** daemon stops the **keyserv** daemon by using the **stopsrc** command.

You could also use the System Management Interface Tool (SMIT) **smit rmkeyserv** fast path to run this command.

Flags

Item	Description
-I	Comments the entry for the **keyserv** daemon in the **/etc/rc.nfs** file.
-B	Comments the entry for the **keyserv** daemon in the **/etc/rc.nfs** file and stops the **keyserv** daemon. This flag is the default.

Item	Description
-N	Stops the **keyserv** daemon using the **stopsrc** command. This flag does not change the **/etc/rc.nfs** file.

Examples

To comment the entry in the **/etc/rc.nfs** file that starts the **keyserv** daemon, enter:

```
rmkeyserv -I
```

This command will not stop the currently executing daemon.

Files

Item	Description
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.

rmlpcmd Command

Purpose

Removes one or more least-privilege (LP) resources from the resource monitoring and control (RMC) subsystem.

Syntax

To remove one or more LP resources:

- From the local node:

  `rmlpcmd` [`-h`] [`-TV`] *resource_name1* [, *resource_name2* , ...]
- From all nodes in a domain:

  `rmlpcmd -a` [`-h`] [`-TV`] *resource_name1* [, *resource_name2* , ...]
- From a subset of nodes in a domain:

  `rmlpcmd -n` *host1* [,*host2*,...] [`-h`] [`-TV`] *resource_name1* [, *resource_name2* , ...]

Description

The `rmlpcmd` command removes one or more LP resources from the RMC subsystem. An LP resource is a `root` command or script to which users are granted access based on permissions in the LP access control lists (ACLs). You can use the `rmlpcmd` command to remove LP resources from particular nodes or all nodes in a domain. If you want to remove locked LP resources, you must first use the `chlpcmd` command to unset the resource's `Lock` attribute.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the `-a` flag. If you want this command to run on a subset of nodes in a domain, use the `-n` flag. Otherwise, this command runs on the local node.

Flags

-a

Removes one or more LP resources from all nodes in the domain. The `CT_MANAGEMENT_SCOPE` environment variable's setting determines the cluster scope. If `CT_MANAGEMENT_SCOPE` is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists

2. The peer domain, if it exists
3. Local scope

The `rmlpcmd` command runs once for the first valid scope that the LP resource manager finds. For example, suppose a management domain and a peer domain exist and the `CT_MANAGEMENT_SCOPE` environment variable is not set. In this case, `rmlpcmd −a` runs in the management domain. To run `rmlpcmd −a` in the peer domain, you must set `CT_MANAGEMENT_SCOPE` to 2.

-n *host1*[,*host2*,...]
Specifies one or more nodes in the domain from which the LP resource is to be removed. By default, the LP resource is removed from the local node. The −n flag is valid only in a management or peer domain. If the CT_MANAGEMENT_SCOPE variable is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The `rmlpcmd` command runs once for the first valid scope that the LP resource manager finds.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error.

-V
Writes the command's verbose messages to standard output.

Parameters

***resource_name1*[,*resource_name2*,...]**
Specifies one or more LP resources to be removed.

Security

To run the `rmlpcmd` command, you need read and write permission in the Class ACL of the `IBM.LPCommands` resource class. Permissions are specified in the LP ACLs on the contacted system. See the `lpacl` file for general information about LP ACLs and the *RSCT Administration Guide* for information about modifying them.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
The resource was not found.

Environment Variables

CT_CONTACT
Determines the system that is used for the session with the RMC daemon. When `CT_CONTACT` is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If `CT_CONTACT` is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the LP resources that are processed.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to process the LP resource. The management scope determines the set of possible target nodes where the resource can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. When the `-V` flag is specified, this command's verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To remove an LP resource named LP1, enter:

```
rmlpcmd LP1
```

2. To remove LP resources LP1 and LP2, enter:

```
rmlpcmd LP1 LP2
```

Location

/opt/rsct/bin/rmlpcmd
Contains the `rmlpcmd` command

rmlv Command

Purpose

Removes logical volumes from a volume group.

Syntax

rmlv [**-B**] [**-f**] [**-p** *Physical Volume*] *LogicalVolume ...*

Description

Attention: This command destroys all data in the specified logical volumes.

The **rmlv** command removes a logical volume. The LogicalVolume parameter can be a logical volume name or logical volume ID. The logical volume first must be closed. If the *volume group* is varied on in concurrent mode, the logical volume must be closed on all the concurrent nodes on which *volume group* is varied on. For example, if the logical volume contains a file system, it must be unmounted. However, removing the logical volume does not notify the operating system that the file system residing on it have been destroyed. The command **rmfs** updates the **/etc/filesystems** file.

Note:

1. To use this command, you must either have root user authority or be a member of the **system** group.
2. You cannot use the **rmlv** command on a snapshot volume group or a volume group that has a snapshot volume group.
3. You cannot use the **rmlv** command on an active firmware assisted dump logical volume.
4. In AIX 7.2 Technology Level 1, or later, after the partition is freed by running the `rmlv` command, the space reclamation process runs for the freed partition.

You could also use the System Management Interface Tool (SMIT) **smit rmlv** fast path to run this command.

Flags

Item	Description
-B	Issues a **chlvcopy -B -s** for the parent logical volume if the logical volume was created using the **-l** flag. If it is a regular logical volume then the **-B** flag is ignored.
-f	Removes the logical volumes without requesting confirmation.
-p *PhysicalVolume*	Removes only the logical partition on the *PhysicalVolume*. The logical volume is not removed unless there are no other physical partitions allocated.  **Attention:** If the logical volume spans multiple physical volumes, the removal of only logical partitions on the *PhysicalVolume* can jeopardize the integrity of the entire logical volume.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

Attention: The command used in this example destroys all data in the logical volumes.

To remove logical volume `lv05` without requiring user confirmation, enter the following command:

```
rmlv -f lv05
```

The logical volume is removed from the volume group.

Files

Item	Description
/usr/sbin/rmlv	Directory where the **rmlv** command resides.
/tmp	Directory where the temporary files are stored while the command is running.
/etc/filesystems	Lists the known file systems and defines their characteristics.

rmlvcopy Command

Purpose

Removes copies from a logical volume.

Syntax

rmlvcopy[**-p** *mirrorpool*] *LogicalVolume Copies* [*PhysicalVolume* ...]

Description

The **rmlvcopy** command removes copies from each logical partition in the *LogicalVolume*. Copies are the physical partitions which, in addition to the original physical partition, make up a logical partition. You can have up to two copies in a logical volume. The *Copies* parameter determines the maximum number of physical partitions that remain. The *LogicalVolume* parameter can be a logical volume name or logical volume ID. The *PhysicalVolume* parameter can be the physical volume name or the physical volume ID. If the *PhysicalVolume* parameter is used, then only copies from that physical volume will be removed.

You could also use the System Management Interface Tool (SMIT) **smit rmlvcopy** fast path to run this command.

Note:

1. To use this command, you must either have `root` user authority or be a member of the **system** group.
2. If LVM has not recognized that a disk has failed it is possible that LVM will remove a different mirror. Therefore if you know that a disk has failed and LVM does not show those disks as missing you should specify the failed disks on the command line or you should use **replacepv** to replace the disk or **reducevg** to remove the disk.
3. The **rmlvcopy** command is not allowed on a snapshot volume group.
4. The **rmlvcopy** command is allowed on a volume group that has a snapshot volume group only if the physical volume names are specified on the command line and the specified physical volumes belong to the snapshot volume group.
5. Running the **rmlvcopy** command on an active firmware-assisted dump logical volume temporarily changes the dump device to the **/dev/sysdumpnull** file. After the successful removal of the logical volume copy, the **rmlvcopy** command calls the **sysdumpdev -P** command to set the firmware-assisted dump logical volume to the original dump logical volume.
6. In AIX 7.2 Technology Level 1, or later, after the partition is freed by running the `rmlvcopy` command, the space reclamation process runs for the freed partition.

Flags

-p mirrorpool
Removes a copy from the specified mirror pool. To remove more than one copy, provide multiple [**-p** *mirrorpool*] flags.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To reduce the number of copies of each logical partition belonging to logical volume `lv0112`, enter:

```
rmlvcopy lv0112 2
```

Each logical partition in the logical volume now has at most two physical partitions.

Files

Item	Description
/usr/sbin/rmlvcopy	Contains the **rmlvcopy** command.
/tmp/*	Directory where the temporary files are stored while the command is running.

rmm Command

Purpose

Removes messages from active status.

Syntax

rmm [**+** *Folder*] [*Messages*]

Description

The **rmm** command removes messages from active status by renaming them. To rename a message, the system prefaces the current message number with a , (comma). Inactive files are unavailable to the Message Handler (MH) package. However, system commands can still manipulate inactive files.

Note: The **rmm** command does not change the current message.

Inactive messages should be deleted periodically. An entry can be placed in your **crontab** file to automatically delete all files beginning with a comma.

Flags

Item	Description
+*Folder*	Specifies the folder containing the messages to rename.

Item	Description
Messages	Specifies the messages to rename. You can specify several messages, a range of messages, or a single message. Use the following references to specify a message: ***Number*** Number of the message ***Sequence*** A group of messages specified by the user. Recognized values include: **all** All messages in a folder **cur or . (dot)** Current message. This is the default. **first** First message in a folder **last** Last message in a folder **next** Message following the current message **prev** Message preceding the current message
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To remove the current message in the current folder, enter:

```
rmm
```

2. To remove messages 2 through 5 from the `sales` folder, enter:

```
rmm  +sales 2-5
```

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile**:

Item	Description
`Current-Folder:`	Sets the default current folder.
`Path:`	Specifies the *UserMhDirectory*.
`rmmproc:`	Specifies the program used to remove messages from a folder.

Files

Item	Description
$HOME/.mh_profile	Contains the MH user profile.
/usr/bin/rmm	Contains the **rmm** command.

rmnamsv Command

Purpose

Unconfigures TCP/IP-based name service on a host.

Syntax

rmnamsv [**-f** | **-F** *FileName*]

Description

The **rmnamsv** high-level command unconfigures a TCP/IP-based name service on a host. You can unconfigure name service for a host functioning as a client.

To unconfigure name service for a client, the **rmnamsv** command calls the **namerslv** low-level command to unconfigure entries in the **/etc/resolv.conf** file or to rename the **/etc/resolv.conf** file to a default or user-specified file name.

You could also use the System Management Interface Tool (SMIT) **smit rmnamerslv** fast path to run this command.

Flags

Item	Description
-F *FileName*	Renames the system configuration database to the file name specified by *FileName*.
-f	Specifies that the default file name (**/etc/resolv.conf.sv**) should be used to rename the **/etc/resolv.conf** file.

Files

Item	Description
/etc/resolv.conf	Contains the default system configuration database.

rmnfs Command

Purpose

Changes the configuration of the system to stop running NFS daemons.

Syntax

/usr/sbin/rmnfs [**-I** | **-N** | **-B**]

Description

The **rmnfs** command changes the current configuration of the system so that the **/etc/rc.nfs** file is not executed on system restart. In addition, you can direct the command to stop NFS daemons that are currently running.

Flags

Item	Description
-B	Removes the entry in the **inittab** file and stops NFS daemons that are currently executing. This flag is the default.
-I	Removes the entry in the **inittab** file that starts NFS daemons on system restart.
-N	Stops immediately NFS daemons and does not change the **inittab** file.

Examples

To stop all of the NFS daemons immediately, enter:

```
rmnfs -N
```

This command will not change the **inittab** file.

rmnfsexp Command

Purpose

Unexports a directory from NFS clients.

Syntax

/usr/sbin/rmnfsexp **-d** *Directory* [**-V** *Exported Version*] [**-f** *Exports_file*] [**-I** | **-B** | **-N**] [**-F**]

Description

The **rmnfsexp** command removes an entry from the exports list for NFS clients. This command starts the **exportfs** command to unexport the specified directory. If an entry exists in the **/etc/exports** file, that entry is removed.

Flags

Item	Description
-d *Directory*	Specifies the directory to be unexported.
-f *Exports_File*	Specifies the full path name of the exports file to use if other than the **/etc/exports** file.
-I	Directs the command to remove the entry from the **/etc/exports** file without executing the **exportfs** command.
-B	Removes the entry in the **/etc/exports** file for the directory specified, and executes the **exportfs** command to remove the export.
-N	Unexports the directory immediately by invoking the **exportfs** command. The **/etc/exports** file is not modified with this flag.

Item	Description
-V *Exported Version*	Specifies the version to be used for unexporting the directory. The valid version numbers are 2, 3 and 4.
-F	Forces to unexport the directory.

Examples

1. To unexport a directory immediately, enter the following command:

   ```
   rmnfsexp -d /usr -N
   ```

   In this example, the **/usr** directory is unexported immediately.
2. To unexport a directory immediately and after every system restart, enter the following command:

   ```
   rmnfsexp -d /home/guest -B
   ```
3. To unexport a directory immediately from an exports file other than the **/etc/exports** file, enter the following command:

   ```
   rmnfsexp -d /usr -f /etc/exports.other -N
   ```
4. To unexport the **/common/documents** directory that is exported as version 3, enter the following command:

   ```
   rmnfsexp -d /common/documents -V 3
   ```

Files

Item	Description
/etc/xtab	Lists the currently exported directories.

html

rmnfsmnt Command

Purpose

Removes an NFS mount.

Syntax

/usr/sbin/rmnfsmnt -f *PathName* [**-I** | **-B** | **-N**]

Description

The **rmnfsmnt** command removes the appropriate entry from the **/etc/filesystems** file and unmounts the file system specified. When used with the **-N** flag, the **rmnfsmnt** command unmounts the file system and does not modify the **/etc/filesystems** file.

Flags

Item	Description
-B	Removes the entry in the **/etc/filesystems** file and unmounts the directory. If no entry exists in the **/etc/filesystems** file, the flag makes no changes to the file. If the file system is not currently mounted, the flag does not attempt to unmount it. This flag is the default.
-f *PathName*	Specifies the path name of the NFS-mounted file system.
-I	Removes the entry specified by the path name from the **/etc/filesystems** file.
-N	Unmounts the specified directory and does not modify the **/etc/filesystems** file.

Examples

1. To unmount a file system, enter:

   ```
   rmnfsmnt -f /usr/man -N
   ```

   In this example, the `/usr/man` file system is unmounted.
2. To remove a mount for a file, enter:

   ```
   rmnfsmnt -f /usr/local/man -B
   ```

   In this example, the mount for the `/usr/local/man` file is removed.

File

Item	Description
/etc/filesystems	Lists the remote file systems to mount during the system restart.

rmnfsproxy Command

Purpose

Removes a previously configured and mounted instance of a proxy-enabled Cachefs.

Syntax

/usr/sbin/rmnfsproxy *Cachefs_mount_point*

Description

The specified Cachefs mount is unmounted. The corresponding NFS client mount is also unmounted. Finally, all cached information created in the local file system is removed.

Note: If the Cachefs instance is NFS-exported, the instance must first be unexported before running `rmnfsproxy`.

Parameters

Item	Description
Cachefs_mount_point	Specifies where the proxy-enabled Cachefs instance to be removed was mounted.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Examples

1. To remove a previously configured `/proj1_cached` Cachefs instance, enter:

```
rmnfsproxy /proj1_cached
```

Location

`/usr/sbin/rmnfsproxy`

rmnotify Command

Purpose

Removes a notify method definition from the Notify object class.

Syntax

rmnotify -n *NotifyName*

Description

The **rmnotify** command removes a notify method definition from the notify object class.

Flags

Item	Description
-n *NotifyName*	Specifies the notify method definition to be removed. The **rmnotify** command is unsuccessful if the *NotifyName* name does not already exist in the Notify object class.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/etc/objrepos/SRCnotify	Specifies the SRC Notify Method object class.

rmpath Command

Purpose

Removes from the system a path to an MPIO capable device.

Syntax

rmpath [**-l** *Name*] [**-p** *Parent*] [**-w** *Connection*] [**-i** *PathID*]

rmpath [**-l** *Name*] [**-p** *Parent*] [**-w** *Connection*] [**-d**] [**-g**]

rmpath **-h**

Description

The **rmpath** command unconfigures, and possibly undefines, one or more paths associated with the specified target device (**-l** *Name*). The set of paths that are removed are determined by the combination of the **-l** *Name*, **-p** *Parent*, and **-w** *Connection* flags. If the command will result in all paths associated with the device being unconfigured or undefined, the command will exit with an error and without unconfiguring or undefining any path. In this situation, **rmdev** command must be used instead to unconfigure or undefine the target device itself.

The default action unconfigures each specified path, but does not completely remove it from the system. If the **-d** flag is specified, the **rmpath** command unconfigures (if necessary) and removes, or deletes, the path definition(s) from the system.

When the **rmpath** command finishes, it displays a status message. When unconfiguring paths, it is possible for this command to be able to unconfigure some paths and not others (e.g., paths that are in the process of doing I/O cannot be unconfigured).

The **rmpath** command provides status messages about the results of operation. Messages in one of the following formats will be generated:

path [defined | deleted]
This message is displayed when a single path was successfully unconfigures or undefined. If the path is successfully configured the message `path available` displays. If the path is not successfully configured and there is no explicit error code returned by the method, the message `path defined` displays.

paths [defined | deleted]
This message is displayed if multiple paths were identified and all paths were successfully unconfigured or undefined. If the **-d** flag is not specified, the message would be `paths defined`. If the **-d** flag is specified, the message would be `paths deleted`.

some paths [defined | deleted]
This message is display if multiple paths were identified, but only some of them were successfully unconfigured or undefined. If the **-d** flag is not specified, the message would be `some paths defined`. If the '**-d** flag is specified, the message would be `some paths deleted`.

no paths processed
This message is generated if no paths were found matching the selection criteria.

Flags

Item	Description
-d	Indicates that the specified paths are to be deleted from the system.
-g	Forces the remove path operation to run on a locked device.

Item	Description
-h	Displays the command usage message.
-i *PathID*	Indicates the path ID associated with the path to be removed and is used to uniquely identify a path.
-l *Name*	Specifies the logical device name of the target device whose path is to be removed. The paths to be removed are qualified via the **-p** and **-w** flags.
-p *Parent*	Indicates the logical device name of the parent device to use in qualifying the paths to be removed. Since all paths to a device cannot be removed by this command, either this flag, the **-w** flag, or both must be specified.
-w *Connection*	Indicates the connection information to use in qualifying the paths to be removed. Since all paths to a device cannot be removed by this command, either this flag, the **-p** flag, or both must be specified.

Security

Privilege Control: Only the root user and members of the system group have execute access to this command.

Auditing Events:

Event	Information
DEV_Change	rmpath,Unconfigure,<unconfigure method arguments>
DEV_Change	rmpath,Undefine,<undefine method arguments>

Examples

1. To unconfigure the path from **scsi0** to **hdisk1** at connection **5,0**, type:

   ```
   rmpath -l hdisk1 -p scsi0 -w "5,0"
   ```

   The message generated would be similar to:

   ```
   path defined
   ```

2. To unconfigure all paths from **scsi0** to **hdisk1**, type:

   ```
   rmpath -l hdisk1 -p scsi0
   ```

   If all paths were successfully unconfigured, the message generated would be similar to:

   ```
   paths defined
   ```

   However, if only some of the paths were successfully unconfigured, the message would be similar to:

   ```
   some paths defined
   ```

3. To undefine the path definition between **scsi0** and **hdisk1** at connection **5,0**, type:

   ```
   rmpath -d -l hdisk1 -p scsi0 -w "5,0"
   ```

   The message generated would be similar to the following:

```
path deleted
```

4. To unconfigure all paths from **scsi0** to **hdisk1**, type:

```
rmpath -d -l hdisk1 -p scsi0
```

The message generated would be similar to:

```
paths deleted
```

Files

Item	Description
/usr/sbin/rmpath	Contains the **rmpath** command.

rmprtsv Command

Purpose

Unconfigures a print service on a client or server machine.

Syntax

rmprtsv { **-c** | **-s** } [**-T** | **-U** | **-A**] [**-h "***HostName ...***"** | **-H** *FileName*] [**-q "***QEntry ...***"**] [**-q** *QEntry* **-v "***DeviceName ...***"**]

Description

The **rmprtsv** high-level command unconfigures a print service on a client or server machine.

To unconfigure print service for a client, the **rmprtsv** command calls the **rmque** and **rmquedev** commands to disable the client spool queue and to remove the appropriate entries in the **/etc/qconfig** file.

To unconfigure print service for a server, the **rmprtsv** command performs the following procedure:

1. Calls the **stopsrc** command to deactivate the **lpd** and **qdaemon** servers.
2. Calls the **ruser** low-level command to unconfigure remote users on the print server.
3. Calls the **rmque** and **rmquedev** commands to unconfigure the spooler and its device queues, and delete the appropriate entries in the server's **/usr/lib/lpd/qconfig** file.

Flags

Item	Description
-A	Removes specified entries from the **/etc/qconfig** file but does not fully unconfigure print service.
-c	Unconfigures print service for a client machine. Use the **-q** flag with the **-c** flag.
-H *FileName*	Specifies the name of a file containing a list of host names to be left configured for print service.
-h "*HostName...***"**	Specifies a list of remote host names not allowed to use the print server. Note that the queuing system does not support multibyte host names.
-q "*QEntry...***"**	Specifies a list of entries to remove from the **/etc/qconfig** file.

Item	Description
-s	Unconfigures print service for a server machine. The **-h**, **-H**, and **-q** flags should be used with the **-s** flag.
-T	Stops print service but does not fully unconfigure print service.
-U	Removes specified remote users on the print server but does not fully unconfigure print service.
-v "*DeviceName...***"**	Specifies a list of the names of the device stanzas in the **qconfig** file. Must be used with the **-q** *QEntry* flag.

Files

Item	Description
/etc/qconfig	Contains configuration information for the printer queueing system.

rmps Command

Purpose

Removes an inactive paging space.

Syntax

rmps[**-t** *ps_helper*] *PagingSpace*

Description

The **rmps** command removes an inactive paging space. The *PagingSpace* parameter specifies the name of the paging space that must be removed. This paging space is the name of the logical volume on which the paging space is present.

For an NFS paging space, the *PagingSpace* parameter specifies the name of the paging space to be removed. The device and its definition, which corresponds to this paging space, is removed from the system. Nothing is changed on the NFS server where the file that is used for paging is present.

If the **-t** flag is specified, the argument is assumed to be a third-party helper executable. If the helper executable is present in the `/sbin/helpers/pagespace` path, the executable is created by passing the **-r** flag to specify the **rmps** command. The `/etc/swapspaces` directory is modified so that the helper executable returns zero.

The helper executable is used to remove the paging space. If the named helper does not exist in the `/sbin/helpers/pagespace` directory, the **rmps** command displays a usage error. The helper executable exits with a value 0 when successful and a non-zero value when it fails.

Active pages can be removed by first deactivating them with the **swapoff** command.

Flags

Item	Description
-t	Specifies to use the helper program under `/sbin/helpers/pagespace` directory. **ps_helper** Name of the helper program for a third-party device.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To remove PS01 paging space, run the following command:

   ```
   rmps PS01
   ```

   This removes the PS01 paging space.
2. To remove PS01 paging space by using the helper program `foo`, run the following command:

   ```
   rmps –t foo PS01
   ```

   This removes the PS01 paging space.

Files

Item	Description
`/etc/swapspaces`	Specifies the paging space devices and their attributes.

rmqos Command

Purpose

Changes the configuration of the system to remove QoS support.

Syntax

/usr/sbin/rmqos [**-I** | **-N** | **-B**]

Description

The **rmqos** command changes the current configuration of the system to remove Quality of Service (QoS) support.

Flags

Item	Description
-B	Removes the entry in the **inittab** file that enables QoS at system startup and stops the QoS daemons. This flag is the default.
-I	Removes the entry in the **inittab** file that enables QoS at system startup but does not affect the currently running QoS subsystem.
-N	Disables QoS support immediately but does not change the **inittab** file.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
inittab	Controls the initialization process of the system.
/etc/rc.qos	Contains the startup script for the QoS daemons.

rmque Command

Purpose

Removes a printer queue from the system.

Syntax

rmque **-q** *Name*

Description

The **rmque** command removes a queue from the system configuration by deleting the queue stanza named by the **-q** flag from the **/etc/qconfig** file. All queue devices must be deleted using the **rmquedev** command before entering this command.

You could also use the System Management Interface Tool (SMIT) **smit rmque** fast path to run this command.

Recommendation: To edit the **/etc/qconfig** file, use the **chque**, **mkque**, **rmque**, **chquedev**, **mkquedev**, and **rmquedev** commands or SMIT. Further, it is recommended to run these commands during slow or off-peak time.

If manual editing of the **/etc/qconfig** file is necessary, you can first issue the **enq -G** command to bring the queuing system and the **qdaemon** to a halt after all jobs are processed. Then you can edit the **/etc/qconfig** file and restart the **qdaemon** with the new configuration.

Flags

Item	Description
-q *Name*	Specifies the name of the queue to be removed.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To remove printer queue `lp0`, enter:

```
rmque -q lp0
```

Files

Item	Description
/usr/bin/rmque	Contains the **rmque** command.

Item	Description
/etc/qconfig	Contains the configuration file.

rmquedev Command

Purpose

Removes a printer or plotter queue device from the system.

Syntax

rmquedev **-d** *Name* **-q** *Name*

Description

The **rmquedev** command removes a printer or plotter queue device from the system configuration by deleting the device stanza named by the **-d** flag from the **/etc/qconfig** file. It also modifies the `Device=DeviceName1,DeviceName2,DeviceName3` line of the queue stanza, deleting the entry for the device `Name`.

You could also use the System Management Interface Tool (SMIT) **smit rmquedev** fast path to run this command.

Recommendation: To edit the **/etc/qconfig** file, use the **chque**, **mkque**, **rmque**, **chquedev**, **mkquedev**, and **rmquedev** commands or SMIT. Further, it is recommended to run these commands during slow or off-peak time.

If manual editing of the **/etc/qconfig** file is necessary, you can first issue the **enq -G** command to bring the queuing system and the **qdaemon** to a halt after all jobs are processed. Then you can edit the **/etc/qconfig** file and restart the **qdaemon** with the new configuration.

Flags

Item	Description
-d *Name*	Specifies the *Name* of the device stanza to be deleted from the **qconfig** file.
-q *Name*	Specifies the *Name* of the device to be modified in the preceding queue stanza.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To delete the `loc` device stanza from the **/etc/qconfig** file and modify the "`DEVICE =`" stanza in the preceding queue stanza `lpq`, enter:

```
rmquedev -q lpq -d loc
```

Files

Item	Description
/usr/bin/rmquedev	Contains the **rmquedev** command.

Item	Description
/etc/qconfig	Configuration file.

rmramdisk Command

Purpose

Removes RAM disks created by the **mkramdisk** command.

Syntax

rmramdisk *ram_disk_name*

Description

The **rmramdisk** command removes the specified RAM disk and the device special files that were created for that RAM disk. RAM disks are also removed when the system is rebooted. Device special files can only be removed via the`rmramdisk` command.

Parameters

Item	Description
ram_disk_name	Name of the specific RAM disk to be removed from memory. If not specified, an error is returned. The names of the RAM disks are in the form of **rramdisk***x* where *x* is the logical RAM disk number (0 through 63).

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

```
# ls -l /dev/*ramdisk2
brw-------   1 root     system       38,  0 Aug 01 05:52 /dev/ramdisk2
crw-------   1 root     system       38,  0 Aug 01 05:52 /dev/rramdisk2
```

To remove the ramdisk2, enter:

```
# rmramdisk ramdisk2

# ls -l /dev/*ramdisk2
ls: 0653-341 The file /dev/*ramdisk2 does not exist.
```

Files

Item	Description
/usr/sbin/rmramdisk	Contains the **rmramdisk** command.

rmresponse Command

Purpose

Removes a response.

Syntax

`rmresponse` [`-f`] [`-q`] [`-h`] [`-TV`] *response*[`:`*node_name*]

Description

The `rmresponse` command removes the response specified by the *response* parameter. The response must already exist in order to be removed. When the response must be removed even if it is linked with conditions, specify the `-f` flag. This forces the response and the links with the conditions to be removed. If the `-f` flag is not specified and links with conditions exist, the response is not removed. This command does not remove conditions.

If a particular response is needed for system software to work properly, it may be locked. A locked response cannot be modified or removed until it is unlocked. If the response you specify on the `rmresponse` command is locked, it will not be removed; instead an error will be generated informing you that the response is locked. To unlock a response, you can use the `-U` flag of the `chresponse` command. However, since a response is typically locked because it is essential for system software to work properly, you should exercise caution before unlocking it.

Flags

-f
Forces the response to be removed even if it is linked with conditions. The links with the conditions are removed as well as the response, but the conditions are not removed.

-q
Does not return an error when *response* does not exist.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

response
Specifies the name of a defined response to be removed.

node_name
Specifies the node in a cluster where the response is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

Security

The user needs write permission for the `IBM.EventResponse` resource class to run `rmresponse`. Permissions are specified in the access control list (ACL) file on the contacted system.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To remove the response definition named "Broadcast event on-shift", run this command:

```
rmresponse  "Broadcast event on-shift"
```

2. To remove the response definition named "Broadcast event on-shift" even if the response is linked with conditions, run this command:

```
rmresponse -f "Broadcast event on-shift"
```

This example applies to management domains:

1. In this example, the current node is the management server. To remove the response definition named "Broadcast event on-shift" on managed node nodeB, run this command:

```
rmresponse  "Broadcast event on-shift":nodeB
```

This example applies to peer domains:

1. To remove the response definition named "Broadcast event on-shift" defined on node nodeA, run this command from any node in the domain:

```
rmresponse  "Broadcast event on-shift":nodeA
```

Location

/opt/rsct/bin/rmresponse

rmrole Command

Purpose

Removes a role.

Syntax

rmrole [**-R** *load_module*] *Name*

Description

The **rmrole** command removes the role identified by the *Name* parameter from the **/etc/security/roles** file. The role name must already exist.

You can use the System Management Interface Tool (SMIT) to run the **rmrole** command.

If the system is configured to use databases from multiple domains, the **rmrole** command finds the first match from the database domains in the order that it was specified by the **secorder** attribute of the roles stanza in the **/etc/nscontrol.conf** file. Meanwhile, the **rmrole** command removes the role entry from the domain. If any matching roles from the rest of the domains exist, they are not affected. Use the **-R** flag to remove a role from a specific domain.

When the system is operating in enhanced role based access control (RBAC) mode, roles removed from the role database still exist in the kernel security tables (KST) until the KST is updated with the **setkst** command.

Flags

Item	Description
-R *load_module*	Specifies the loadable module to use for role deletion.

Security

The **rmrole** command is a privileged command. You must have the **aix.security.role.remove** authorization to run the command:

Item	Description
aix.security.role.remove	Required to run the command.

Files Accessed:

Mode	File
rw	/etc/security/roles
r	/etc/security/user.roles

Auditing Events:

Event	Information
ROLE_Remove	role

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To remove the `ManageObjects` role, use the following command:

```
rmrole ManageObjects
```

2. To remove the `ManageRoles` role from LDAP, use the following command:

```
rmrole -R LDAP ManageRoles
```

Files

Item	Description
/etc/security/roles	Contains the attributes of roles.
/etc/security/user.roles	Contains the role attribute of users.

rmrpdomain Command

Purpose

Removes a peer domain that has already been defined.

Syntax

`rmrpdomain` [`-f`] [`-q`] [`-h`] [`-TV`] *peer_domain*

Description

The `rmrpdomain` command removes the peer domain definition that is specified by the *peer_domain* parameter. The peer domain that is to be removed must already be defined. This command must be run on a node that is defined in the peer domain. When `rmrpdomain` is run on a node that is online to the peer domain, it removes the peer domain definition on all nodes defined to the peer domain that are reachable from that node. If a node defined to the peer domain is not reachable, that node's local peer domain definition is not removed. To remove the local peer domain definition when the peer domain is not online or when the node is not online to the peer domain, run the `rmrpdomain` command on that node and specify the `-f` flag.

The most efficient way to remove a peer domain definition is to make sure the peer domain is online. Then, from a node that is online to the peer domain, run the `rmrpdomain` command. If there are nodes that are not reachable from the node on which the `rmrpdomain` command was run, on each of those nodes, run the `rmrpdomain` command using the `-f` flag. This can be done at a later time if the node itself is not operational.

The `-f` flag must also be used to override a subsystem's rejection of the peer domain removal. A subsystem may reject the request if a peer domain resource is busy, for example. Specifying the `-f` flag in this situation indicates to the subsystems that the peer domain definition must be removed.

The `rmrpdomain` command does not require configuration quorum. Therefore, this command is still successful if it is issued to a minority subcluster. Later, the majority subcluster may become active. If so, the domain is still removed.

If a Cluster-Aware AIX (CAA) cluster is configured and this peer domain is representing it, the **rmrpdomain** command removes the underlying CAA cluster as well.

Flags

-f
Forces the peer domain to be removed. The force flag is required to remove a peer domain definition:

- from the local node when the node is not online to the peer domain.
- when a subsystem may reject the request, as when resources are allocated, for example.

-q
Specifies quiet mode. The command does not return an error if the peer domain does not exist.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

peer_domain
Specifies the name of the defined peer domain that is to be removed.

Security

The user of the `rmrpdomain` command needs write permission to the `IBM.PeerDomain` resource class on each node that is to be defined to the peer domain. By default, `root` on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

6
The peer domain definition does not exist.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

The node on which this command is run must be defined to the peer domain and should be able to reach all of the nodes that are defined to the peer domain. The node's local peer domain definition will not be removed if the node is not reachable.

Implementation Specifics

This command is part of the **rsct.basic.rte** fileset for AIX®.

Standard Input

When the `-f "-"` or `-F "-"` flag is specified, this command reads one or more node names from standard input.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To remove the peer domain definition of ApplDomain where nodeA, nodeB, and nodeC are defined and online to *ApplDomain*, and all are reachable to each other, run this command on nodeA, nodeB, or nodeC:

```
rmrpdomain ApplDomain
```

2. To remove the local peer domain definition of ApplDomain on nodeD when nodeD is not online to the peer domain, the peer domain is offline, or the peer domain does not exist, run this command on nodeD:

```
rmrpdomain -f ApplDomain
```

3. To remove the peer domain definition of ApplDomain where nodeA, nodeB, and nodeC are defined and online to ApplDomain, all are reachable to each other, and to prevent a subsystem from rejecting the request, run this command on nodeA, nodeB, or nodeC:

```
rmrpdomain -f ApplDomain
```

Location

/opt/rsct/bin/rmrpdomain

Files

The /etc/services file is modified.

rmrpnode Command

Purpose

Removes one or more nodes from a peer domain definition.

Syntax

rmrpnode [-f] [-q] [-h] [-TV] *node_name1* [*node_name2* ...]

rmrpnode -F { *file_name* | "-" } [-f] [-q] [-h] [-TV]

Description

The rmrpnode command removes one or more nodes from the online peer domain where the command is run. The command must be run on a node that is online to the peer domain in which the nodes are to be removed. The nodes that are to be removed must be offline to the peer domain and must be reachable from the node where the command is run. To take nodes offline, use the stoprpnode command.

If a Cluster-Aware AIX (CAA) cluster is configured and this peer domain is representing it, the **rmrpnode** command removes the nodes from the underlying CAA cluster as well.

Specifying the **-f** flag forces the specified nodes to be removed from the peer domain. When the last tiebreaker node is removed using **rmrpnode -f**, only the remaining quorum nodes (as opposed to all nodes) are converted to being tiebreaker nodes.

If the **-f** flag is not specified when this command is run:

- more than half of the quorum nodes must be online to remove one or more nodes from the domain
- an error is returned if the peer domain has no remaining tiebreaker nodes as a result

See the *Administering RSCT* for more information about quorum nodes and tiebreaker nodes.

Flags

-f
Forces the specified nodes to be removed from the peer domain.

When the last tiebreaker node is removed using this flag, only the remaining quorum nodes (as opposed to all nodes) are converted to being tiebreaker nodes.

See the *Administering RSCT* for more information about quorum nodes and tiebreaker nodes.

-q
Specifies quiet mode. The command does not return an error if the specified nodes are not in the peer domain.

-F { *file_name* | "–" }
Reads a list of node names from *file_name*. Each line of the file is scanned for one node name. The pound sign (#) indicates that the remainder of the line (or the entire line if the # is in column 1) is a comment.

Use `-F "-"` to specify `STDIN` as the input file.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

***node_name1* [*node_name2* ...]**
Specifies the peer domain node names of the nodes to be removed from the peer domain definition. You can remove one or more nodes using the `rmrpnode` command. You must specify the node names in exactly the same format as they were specified with the `addrpnode` command or the `mkrpdomain` command. To list the peer domain node names, run the `lsrpnode` command.

Security

The user of the `rmrpnode` command needs write permission for the `IBM.PeerNode` resource class on each node that is to be removed from the peer domain. By default, `root` on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

6
The node does not exist in the peer domain.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is online in the peer domain in which the nodes are to be removed. The nodes to be removed must also be offline to the peer domain.

Implementation Specifics

This command is part of the **rsct.basic.rte** fileset for AIX®.

Standard Input

When the `-F "-"` flag is specified, this command reads one or more node names from standard input.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

To remove the peer domain definitions of nodes `nodeB` and `nodeC` from the peer domain `ApplDomain`, when `nodeA` is defined and online to `ApplDomain`, and `nodeB` and `nodeC` are reachable from `nodeA`, run this command from `nodeA`:

```
rmrpnode nodeB nodeC
```

Location

/opt/rsct/bin/rmrpnode

rmrset Command

Purpose

Remove an rset from the system registry.

Syntax

```
rmrset rsetname
```

Description

The **rmrset** command removes an rset or exclusive rset (xrset) from the system registry. When used to delete an xrset, the `rmrset` command changes the state of the corresponding CPUs on the system to general use mode. Deleting an xrset requires root privilege.

Parameters

Item	Description
rsetname	The name of the rset to be removed from the system registry. The name consists of a *namespace* and an *rsname* separated by a "/" (slash). Both the *namespace* and *rsname* may contain up to 255 characters. See the **rs_registername()** service for additional information about character set limits of rset names.

Security

The user must have `root` authority, or CAP_NUMA_ATTACH capability and write access permission to the specified rset.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To remove **test/cpus0to7** from system registry, type:

```
rmrset test/cpus0to7
```

Files

Item	Description
/usr/bin/rmrset	Contains the **rmrset** command.

rmrsrc Command

Purpose

Removes a defined resource.

Syntax

To remove one or more resource.

- entered on the command line:

  `rmrsrc -s` "*selection_string*" [`-a | -N` { *node_file* | `"-"` }] [`-h`] [`-TV`] *resource_class*

  `rmrsrc -r` "*resource_handle*" [`-h`] [`-TV`]

- predefined in an input file:

  `rmrsrc -f` *resource_data_input_file* `-s` "*selection_string*" [`-a | -N` { *node_file* | `"-"` }] [`-h`] [`-TV`] *resource_class*

  `rmrsrc -f` *resource_data_input_file* `-r` "*resource_handle*" [`-h`] [`-TV`]

To display the names and datatypes of the command arguments:

`rmrsrc -l [-h] resource_class`

Description

The `rmrsrc` command removes — or "undefines" — the specified resource instance (or instances). The `rmrsrc` command makes a request to the resource monitoring and control (RMC) subsystem to undefine a specific resource instance. The resource manager of the resource removes the resource.

The first format of this command requires a resource class name parameter and a selection string specified using the `-s` flag. All resources in the specified resource class that match the specified selection string are removed. If the selection string identifies more than one resource to be removed, it is the same as running this command once for each resource that matches the selection string.

The second format of this command allows the actual resource handle linked with a specific resource to be specified as the parameter. It is expected that this form of the command would be more likely used from within a script.

Instead of specifying multiple node names in *selection_string*, you can use the `-N` *node_file* flag to indicate that the node names are in a file. Use `-N "-"` to read the node names from standard input.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM `nodegrp` command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference.*

Flags

-a
Specifies that this command applies to all nodes in the cluster. The cluster scope is determined by the `CT_MANAGEMENT_SCOPE` environment variable. If it is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management and peer domain exist, `rmrsrc -a` with `CT_MANAGEMENT_SCOPE` not set will apply to the management domain. In this case, to apply to the peer domain, set `CT_MANAGEMENT_SCOPE` to 2.

-f *resource_data_input_file*
Specifies the name of the file that contains resource argument information.

-l
Lists the command arguments and datatypes. Some resource managers accept additional arguments that are passed to the remove request. Use this flag to list any defined command arguments and the datatypes of the command argument values.

-N { *node_file* | "-" }
Specifies that node names are read from a file or from standard input. Use `-N` *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (*#*) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use `-N "-"` to read the node names from standard input.

The CT_MANAGEMENT_SCOPE environment variable determines the scope of the cluster. If CT_MANAGEMENT_SCOPE is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and CT_MANAGEMENT_SCOPE is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-r "*resource_handle*"
Specifies a resource handle. The resource handle must be specified using the format: "0x*nnnn* 0x*nnnn* 0x*nnnnnnnn* 0x*nnnnnnnn* 0x*nnnnnnnn* 0x*nnnnnnnn*", where *n* is any valid hexadecimal digit. The resource handle uniquely identifies a particular resource instance that should be removed.

-s "*selection_string*"
Specifies a selection string. All selection strings must be enclosed within either double or single quotation marks. If the selection string contains double quotation marks, enclose the entire selection string in single quotation marks. For example:

```
-s 'Name == "testing"'
```

```
-s 'Name ?= "test"'
```

Only persistent attributes can be listed in a selection string. For information on how to specify selection strings, see the *RSCT: Administration Guide* .

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

resource_class
Specifies the resource class name. The resource instances for this resource class that match the selection string criteria are removed.

Security

The user needs write permission for the *resource_class* specified in `rmrsrc` to run `rmrsrc`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for information about the ACL file and how to modify it.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
No resources were found that match the selection string.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output.

The command output and all verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To remove the resource with the Name c175n05 from resource class IBM.Host, enter:

```
rmrsrc -s 'Name == "c175n05"' IBM.Host
```

2. To remove the resource linked with resource handle: "0x4017 0x0001 0x00000000 0x0069684c 0x0d52332b3 0xf3f54b45", enter:

```
rmrsrc -r "0x4017 0x0001 0x00000000 0x0069684c 0x0d52332b3 0xf3f54b45"
```

3. To remove the resources named `Test1` from `IBM.Foo` for certain nodes in the cluster, using the `/tmp/common/node_file` file:

```
# common node file
#
```

```
node1.ibm.com     main node
node2.ibm.com     main node
node4.ibm.com     backup node
node6.ibm.com     backup node
#
```

as input, enter:

```
rmrsrc -s 'Name == "Test1"' -N /tmp/common/node_file IBM.Foo
```

Location

/opt/rsct/bin/rmrsrc

rmsecattr Command

Purpose

Removes the definition of the security attributes for a command, a device, a privileged file, or a domain-assigned object in the database.

Syntax

rmsecattr [**-R** *load_module*] { **-c** | **-d** | **-f** | **-o** } *Name*

Description

The **rmsecattr** command removes the security attributes for a command, a device, a file entry, or a domain-assigned object that is identified by the *Name* parameter from the appropriate database. The command interprets the *Name* parameter as a command, device, file entry, or domain-assigned object based on whether the **-c** (command), **-d** (device), **-f** (privileged file), or **-o** (domain-assigned object) flag is specified. If the **-c** flag is specified, the *Name* parameter must include the full path to the command and the command must at that time have an entry in the **/etc/security/privcmds** privileged command database.

If you specify the **-d** flag, the *Name* parameter must include the full path to the device and the device must at that time have an entry in the **/etc/security/privdevs** privileged device database.

If you specify the **-f** flag, the *Name* parameter must include the full path to the file and the file must have an entry in the **/etc/security/privfiles** privileged file database.

If you specify the **-o** flag, the *Name* parameter must include the full path if the object type is file or device and it must have an entry in the **/etc/security/domobjs** domain-assigned object database.

Important: The **rmsecattr** command removes only the definition of its security attributes; it does not remove the actual command, device, or file.

If the system is configured to use databases from multiple domains, the **rmsecattr** command finds the first match from the database domains in the order that was specified by the **secorder** attribute of the corresponding database stanza in the **/etc/nscontrol.conf** file. Meanwhile, the **rmsecattr** command removes that command or device entry from the domain. If any matching entries from the rest of the domains exist, they are not affected. Use the **-R** flag to remove an entry from a specific domain.

Modifications made by this command are not used for the security considerations until the databases are sent to the kernel security tables using the **setkst** command.

Flags

Item	Description
-c	Specifies, when used with the *Name* parameter, the full paths to one or more commands on the system that have entries in the privileged command database.
-d	Specifies, when used with the *Name* parameter, the full paths to one or more devices on the system that have entries in the privileged device database.
-f	Specifies, when used with the *Name* parameter, the full path to a privileged file on the system.
-o	Specifies, when used with the *Name* parameter, an object as specified in the domain-assigned object database.
-R *load_module*	Specifies the loadable module to use for the deletion of the *Name* entry.

Parameters

Item	Description
Name	The object to modify. The *Name* parameter is interpreted according to the **-c**, **-d**, **-f**, or **-o** flags that you specified.

Security

The **rmsecattr** command is a privileged command. It is owned by the root user and the security group, with mode set to 755. You must have at least one of the following authorizations to run the command:

Item	Description
aix.security.cmd.remove	Required to remove the security attributes of a command with the **-c** flag.
aix.security.device.remove	Required to remove the security attributes of a device with the **-d** flag.
aix.security.dobject.remove	Required to remove the security attributes of a domain-assigned object with the **-o** flag.
aix.security.file.remove	Required to remove the security attributes of a file with the **-f** flag.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

File Accessed

File	Mode
/etc/security/domobjs	rw
/etc/security/privcmds	rw
/etc/security/privdevs	rw
/etc/security/privfiles	rw

Examples

1. To remove the `/usr/sbin/mytest` command from the privileged command database, type:

```
rmsecattr -c /usr/sbin/mytest
```

2. To remove the `/dev/mydev` device from the privileged device database, type:

```
rmsecattr -d /dev/mydev
```

3. To remove the `/dev/mydev` device from the privileged device database in LDAP, type:

```
rmsecattr -R LDAP -d /dev/mydev
```

4. To remove the **/etc/testconf** file from the privileged file database, type:

```
rmsecattr -f /etc/testconf
```

5. To remove the network interface `en0` from the domained object database, type:

```
rmsecattr -o objectype=netint en0
```

rmsensor Command

Purpose

Removes a sensor or a microsensor from the resource monitoring and control (RMC) subsystem.

Syntax

`rmsensor` [**-m**] [`-a` | `-n` *host1*[**,** *host2*...] | **-N** { *node_file* | "-" }] [`-h`] [`-v` | `-V`] *sensor_name1* [*sensor_name2*...]

Description

The **rmsensor** command removes one or more sensors from the **IBM.Sensor** resource class or one or more microsensors from the **IBM.MicroSensor** resource class in the RMC subsystem. Use the **-m** flag to remove a microsensor.

If the sensor or microsensor is being monitored, monitoring will be stopped, but the event response resource manager (ERRM) resources defined for monitoring are not removed. To remove the ERRM resources, use the **rmcondition**, **rmresponse**, or **rmcondresp** command against the monitoring resources that were used for this sensor or microsensor.

The **rmsensor** command runs on any node. If you want **rmsensor** to run on all of the nodes in a domain, use the **-a** flag. If you want **rmsensor** to run on a subset of nodes in a domain, use the **-n** flag. Instead of specifying multiple node names using the **-n** flag, you can use the **-N** *node_file* flag to indicate that the node names are in a file. Use **-N "–"** to read the node names from standard input.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

Flags

-a

Removes sensors that match the specified name on all nodes in the domain.
The CT_MANAGEMENT_SCOPE environment variable determines the cluster scope. If CT_MANAGEMENT_SCOPE is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is

valid for the command. The command will run once for the first valid scope found. For example, if both a management domain and a peer domain exist, `rmsensor -a` with CT_MANAGEMENT_SCOPE not set will run in the management domain. In this case, to run in the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-m
Specifies that the resources to be removed are microsensor resources.

-h
Writes the command's usage statement to standard output.

-n *host1*[,*host2*...]
Specifies the node from which the sensor should be removed. By default, the sensor is removed from the local node. This flag is only appropriate in a management domain or a peer domain.

-N {*node_file* | "-"}
Specifies a file or standard input listing the nodes on which the sensor must be removed. This flag is only appropriate in a Cluster Systems Management (CSM) or a peer domain cluster.

-v | **-V**
Writes the command's verbose messages to standard output.

Parameters

***sensor_name1* [*sensor_name2*...]**
Specifies one or more names of sensors to remove.

Security

To remove sensors using this command, you need write permission for the **IBM.Sensor** resource class. To remove microsensors using this command, you need write permission for the **IBM.MicroSensor** resource class. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0
The command has run successfully.

1
An incorrect combination of flags and parameters has been entered.

6
No sensor resources were found.

n
Based on other errors that can be returned by the RMC subsystem.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If this environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled.

The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the rsct.core fileset for AIX®.

Examples

1. To remove the sensor `sensor1`, enter:

```
rmsensor sensor1
```

2. To remove the sensor called **sensor1** from the nodes that are listed in the **/u/joe/common_nodes** file, enter:

```
rmsensor -N /u/joe/common_nodes sensor1
```

where **/u/joe/common_nodes** contains:

```
# common node file
#
node1.myhost.com    main node
node2.myhost.com    backup node
```

3. To remove the microsensor called **IBM.usensor1**, enter:

```
rmsensor -m IBM.usensor1
```

Location

/opt/rsct/bin/rmsensor

rmserver Command

Purpose

Removes a subserver definition from the Subserver Type object class.

Syntax

rmserver -t *Type*

Description

The **rmserver** command removes an existing subserver definition from the Subserver Type object class.

Flags

Item	Description
-t *Type*	Specifies the subserver name that uniquely identifies the existing subserver to be removed. The **rmserver** command is unsuccessful if the *Type* name is not known in the Subserver Type object class.

Security

Auditing Events: If the auditing subsystem has been properly configured and is enabled, the **rmserver** command will generate the following audit record (event) every time the command is executed:

Event	Information
SRC_Delserver	Lists in an audit log the name of the subserver definition that was deleted.

See html

rmsmbcred Command

Purpose

Removes the Server Message Block (SMB) client file system credentials that are stored in the `/etc/smbcred` file for the specified server and user entry.

Syntax

```
rmsmbcred -s server_name -u user_name
```

Description

You must specify the server name and the username in the **rmsmbcred** command. If the specified server-user entry is found in the credentials that are listed in the `/etc/smbcred` file, the corresponding credential entry is removed. After the credential entry is removed, if you want to mount the SMB client file system on the specified server for the specified user, you must manually specify the password in the specified server.

Flags

-s *server_name*
Specifies the name of the remote host, which is an SMB server. The *server_name* parameter can be provided as a hostname, an IP address, or a fully qualified domain name.

-u *user_name*
Specifies the username for which credentials must be removed from the `/etc/smbcred` file.

Exit status

0
The command completed successfully.

>0
An error occurred.

Example

To remove the credential that is stored in `/etc/smbcred` file for `user1` to mount the SMB client file system on the `xxx.in.ibm.com` server, enter the following command:

```
rmsmbcred -s xxx.in.ibm.com -u user1
```

Location

`/usr/sbin/rmsmbcred`

Files

/etc/smbcred
Stores the credentials of the SMB client file system.

rmsock Command

Purpose

Removes a socket that does not have a file descriptor.

Syntax

rmsock *Address TypeofAddress*

Description

The **rmsock** command removes a socket that does not have a file descriptor. It accepts a socket, tcpcb, inpcb, ripcb, or rawcb address and converts it to a socket address. All opened files in every process are then checked to find a match to the socket. If a match is not found, an abort action is performed on that socket regardless of the existence of the socket **linger** option. The port number held by the socket is released. If a match is found, its file descriptor and status of the owner process are displayed to the user. The results are passed to **syslogd** and recorded in the **/var/adm/ras/rmsock.log** file.

If the socket to be removed is not held by any active processes, but there are processes in the exiting state, `rmsock` will not remove the socket specified because the socket could be held by the processes in the exiting state. Any socket that is held by the exiting processes will be cleaned up when those processes exit completely.

Examples

1. To remove a socket from its socket address, type:

   ```
   rmsock 70054edc socket
   ```

   You do not need to specify the type of the socket. It can be a tcpcb, udp, raw, or routing socket.
2. To remove a socket from its inpcb address, type:

   ```
   rmsock 70054edc inpcb
   ```
3. To remove a socket from its tcpcb address, type:

   ```
   rmsock 70054ecc tcpcb
   ```

Files

Item	Description
/usr/sbin	Directory where the **rmsock** command resides.
/var/adm/ras/rmsock.log	Contains the **rmsock.log** file.

rmss Command

Purpose

Simulates a system with various sizes of memory for performance testing of applications.

Syntax

rmss **-c** *MemSize*

rmss **-r**

rmss **-p**

rmss [**-d** *MemSize*] [**-f** *MemSize*] [**-n** *NumIterations*] [**-o** *OutputFile*] [**-s** *MemSize*] *Command*

Description

The **rmss** command simulates a system with various sizes of real memory, without having to extract and replace memory boards. By running an application at several memory sizes and collecting performance statistics, you can determine the memory needed to run an application with acceptable performance. The **rmss** command can be invoked for either of two purposes:

- To change the memory size and then exit, using the **-c** , **-p** , and **-r** flags. This lets you experiment freely with a given memory size.
- To function as a driver program, using the **-s** , **-f** , **-d** , **-n** , and **-o** flags. In this mode, the **rmss** command executes a specified command multiple times over a range of memory sizes, and displays important statistics describing command performance at each memory size. The command can be an executable or shell script file, with or without command line arguments.

The **rmss** command is incompatible with the DR subsystem. If a DR event occurs during the **rmss** command execution, the system can hang. Since the memory removal function of the **rmss** command can be replaced by DR memory removal with the **drmgr** command, the information text of the **rmss** command must be amended with this warning:

Attention: The **rmss** command is incompatible with the AIX DLPAR component, and its usage may result in a hung system. The **drmgr** command provides a safe memory removal function in a DLPAR environment.

Attention: When **rmss** is used on a multiple memory pool system, it may fail with:

```
Failure: VMM unable to free enough frames for stealing.
Choose a larger memory size or retry with less system activity.
```

Or a similar message. This failure can occur when **rmss** has stolen all the frames from a memory pool, and is unable to steal frames from other pools. A workaround is to decrease memory by increments.

The number and size of memory pools on a system can be retrieved with the command:

```
echo "mempool *" | kdb
```

The **-c**, **-p**, and **-r** flags are mutually exclusive. The **-c** flag changes the memory size; the **-p** flag displays the current memory size; and the **-r** flag resets the memory size to the real memory size of the machine.

The **-s**, **-f**, **-d**, **-n**, and **-o** flags are used in combination when the **rmss** command is invoked as a driver program to execute and measure the performance of a command (where a command is an executable or a shell script file) over a range of memory sizes. When invoked this way, the **rmss** command displays performance statistics, such as the response time of the command and the number of page-ins that occurred while the command ran, for each memory size. These statistics, which are also written to a file, are described in this example.

The **-s** and **-f** flags specify the starting and ending points of the range, while the **-d** flag specifies the increment or decrement between memory sizes within the range. The **-n** flag is used to specify the number of times to run the command at each memory size, and the **-o** flag is used to specify the name of an output file into which to write the **rmss** report. The *Command* parameter specifies the command to be run and measured at each memory size.

Note:

1. The **rmss** command reports “usable” real memory. On machines where there is bad memory or where the system is using the memory, **rmss** reports the amount of real memory as the amount of physical real memory minus the memory that is bad or in use by the system. For example, using the **rmss -r** flag might report:

   ```
   Simulated Memory Size changed to 79.9062MB
   ```

   This could be a result of some pages being marked bad or a result of a device that is reserving some pages for its own use (and thus not available to the user).
2. The **rmss** command may underestimate the number of page-ins that are required to run an application if the application, combined with background processes such as daemons, accesses a lot of different files (including directory files). The number of different files that must be accessed to cause such results is approximately 250 files per 8MB of simulated memory size. The following table gives the approximate number of different files that, when accessed at the given simulated memory size, may result in the **rmss** command underestimating page-in requirements.

Simulated Memory Size (MB)	Access to Different Files
8	250
16	500
24	750
32	1000
48	1500
64	2000
128	4000
256	8000

You can use the **filemon** command to determine the number of files accessed while your command runs, if you suspect that it may be accessing many different files.

Flags

Item	Description
-c *MemSize*	Changes the simulated memory size to the *MemSize* value, which is an integer or decimal fraction in units of megabytes. The *MemSize* variable must be between 8MB and the real memory size of the machine. There is no default for the **-c** flag. **Note:** It is difficult to change the simulated memory size to less than 8MB, because of the size of inherent system structures such as the kernel.

Item	Description
-d *MemSize*	Specifies the increment or decrement between memory sizes to be simulated. The *MemSize* value is an integer or decimal fraction in units of megabytes. If the **-d** flag is omitted, the increment or decrement will be 8MB.
-f *MemSize*	Specifies the final memory size. You should finish testing the simulated system by executing the command being tested at a simulated memory size given by the *MemSize* variable, which is an integer or decimal fraction in units of megabytes. The *MemSize* variable must be between 4MB and the real memory size of the machine. If the **-f** flag is omitted, the final memory size will be 8MB. **Note:** It is difficult to finish at a simulated memory size of less than 8MB because of the size of inherent system structures such as the kernel.
-n *NumIterations*	Specifies the number of times to run and measure the command, at each memory size. There is no default for the **-n** flag. If the **-n** flag is omitted, during **rmss** command initialization, the **rmss** command will determine how many iterations of the command being tested are necessary to accumulate a total run time of 10 seconds, and then run the command that many times at each memory size. **Note:** The **rmss** command always executes the command once at each memory size prior to the executions that are measured. This prepares the simulation for the actual test.
-o *OutputFile*	Specifies the file into which to write the **rmss** report. If the **-o** flag is omitted, then the **rmss** report is written to the file **rmss.out**. In addition, the **rmss** report is always written to standard output.
-p	Displays the current simulated memory size.
-r	Resets the simulated memory size to the real memory size of the machine.
-s *MemSize*	Specifies the starting memory size. Start by executing the command at a simulated memory size specified by the *MemSize* variable, which is an integer or decimal fraction in units of megabytes. The *MemSize* variable must be between 4MB and the real memory size of the machine. If the **-s** flag is omitted, the starting memory size will be the real memory size of the machine. **Note:** It is difficult to start at a simulated memory size of less than 8MB, because of the size of inherent system structures such as the kernel.
Command	Specifies the command to be run and measured at each memory size. The *Command* parameter may be an executable or shell script file, with or without command line arguments. There is no default command.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To change the memory size to 13.5MB, enter:

```
rmss -c 13.5
```

2. To print the current memory size, enter:

```
rmss -p
```

3. To reset the memory size to the real memory size of the machine, enter:

```
rmss -r
```

4. To investigate the performance of the command `cc -O foo.c` on memory sizes 32, 24, 16, and 8MB; run and measure the command once at each memory size; and then write the report to the `cc.rmss.out` file, enter:

```
rmss -s 32 -f 8 -d 8 -n 1 -o cc.rmss.out cc -O foo.c
```

5. To investigate the performance of the sequence of commands in the `foo.sh` shell script file on memory sizes starting at the real memory size of the machine and ending at 8MB, by increments of 8MB; let the **rmss** command determine the number of iterations to run and measure the `foo.sh` at file each memory size; and then write the **rmss** report to the `rmss.out` file (with all defaults used in this invocation of the **rmss** command), enter the following:

```
rmss foo.sh
```

6. To investigate the performance of the executable `bar` on memory sizes from 8MB to 16MB, by increments of 0.5MB; run and measure `bar` twice at each memory size; and write the report to the `bar.rmss.out` file, enter:

```
rmss -s 8 -f 16 -d .5 -n 2 -o bar.rmss.out bar
```

7. When any combination of the **-s**, **-f**, **-d**, **-n**, and **-o** flags is used, the **rmss** command runs as a driver program, which executes a command multiple times over a range of memory sizes, and displays statistics describing the command's performance at each memory size.

   An example of the report printed out by the **rmss** command follows:

```
Hostname:  xray.austin.ibm.com
Real memory size:   48.00 Mb
Time of day:  Wed Aug  8 13:07:33 1990
Command:  cc -O foo.c
Simulated memory size initialized to  24.00 Mb.
Number of iterations per memory size = 1 warmup + 1 measured = 2.
Memory size  Avg. Pageins  Avg. Response Time   Avg. Pagein Rate
(megabytes)                      (sec.)           (pageins/sec.)
   ---------------------------------------------------------------
24.00             0.0              113.7                0.0
22.00             5.0              114.8                0.0
20.00             0.0              113.7                0.0
18.00             3.0              114.3                0.0
16.00             0.0              114.6                0.0
14.00             139.0            116.1                1.2
12.00             816.0            126.9                6.4
10.00             1246.0           135.7                9.2
8.00              2218.0           162.9                13.6
```

   This report was generated by the following command:

```
rmss -s 24 -f 8 -d 2 -n 1 cc -O foo.c
```

The top part of the report gives general information, including the machine that the **rmss** command was running on, the real memory size of that machine, the time and date, and the command that was being measured. The next two lines give informational messages that describe the initialization of the **rmss** command. Here, the **rmss** command displays that it has initialized the simulated memory size to 24MB, which was the starting memory size given with the **-s** flag. Also, the **rmss** command prints out the number of iterations that the command will be run at each memory size. The command is to be run twice at each memory size: once to warmup, and once when its performance is measured. The number of iterations was specified by the **-n** flag.

The lower part of the report provides the following for each memory size the command was run at:

- The memory size, along with the average number of page-ins that occurred while the command was run
- The average response time of the command
- The average page-in rate that occurred when the command was run.

  Note: The average page-ins and average page-in rate values include all page-ins that occurred while the command was run, not just those initiated by the command.

Files

Item	Description
/usr/bin/rmss	Contains the **rmss** command.

rmssys Command

Purpose

Removes a subsystem definition from the subsystem object class.

Syntax

rmssys -s *Subsystem*

Description

The **rmssys** command removes an existing subsystem definition from the subsystem object class. It also removes any subservers and notify method definitions that exist for the subsystem being removed.

Flags

Item	Description
-s *Subsystem*	Specifies the name that uniquely identifies the subsystem to be removed. The **rmssys** command is unsuccessful if the subsystem name is not known in the subsystem object class. The **rmssys** command removes any subserver definitions from the Subserver Type object class that are defined for this subsystem, as well as any notify method definitions from the Notify object class that are defined for this subsystem.

Security

For details about selecting and grouping audit events, and configuring audit event data collection, see "Setting Up Auditing" in *Security* topic.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration object class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration object class.
/etc/objrepos/SRCnotify	Specifies the SRC Notify Method object class.
/dev/SRC	Specifies the **AF_UNIX** socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.

rmt Command

Purpose

Allows remote access to magnetic tape devices.

Syntax

rmt

Description

The **rmt** command allows remote access to magnetic tape devices. The remote dump and restore programs use the **rmt** command as a remote magnetic tape protocol module. The **rmt** command is normally started with a **rexec** or **rcmd** subroutine.

The **rmt** command accepts requests specific to the manipulation of magnetic tapes, performs the commands, and then responds with a status indication. All responses are in ASCII and in one of two forms. Successful commands receive responses of `Axxx`, where `xxx` is an ASCII representation of a decimal number. Unsuccessful commands receive responses of `Eyyy error-message`, where `yyy` is one of the possible error numbers described in the **errno.h** file and `error-message` is the corresponding error string as printed from a call to the **perror** subroutine. The protocol is comprised of the following subcommands.

Subcommands

Item	Description
O*DeviceMode*	Opens the device specified by the *Device* parameter using the mode indicated by the *Mode* parameter. The value of the *Device* parameter is a full path name, and that of the *Mode* parameter is an ASCII representation of a decimal number suitable for passing to the **open** subroutine. An open device is closed before a new open operation is performed.
C*Device*	Closes the open device. The device specified with the *Device* parameter is ignored.
L*WhenceOffset*	Performs an **lseek** operation using the specified parameters. The **lseek** subroutine returns the response value.

Item	Description
W*Count*	Writes data onto the open device. From the connection, the **rmt** command reads the number of bytes specified by the *Count* parameter, ending if a premature end-of-file is encountered. The **write** subroutine returns the response value.
R*Count*	Reads, from the open device, the number of bytes of data specified by the *Count* parameter. The **rmt** command then performs the requested read operation and responds with `A`*zzz*, where *zzz* is the number of bytes read if the operation was successful. The data read is then sent. Otherwise, an error in the standard format is returned.
I*OperationCount*	Performs an **STIOCTOP** ioctl subroutine using the specified parameters. The parameters are interpreted as the ASCII representations of the decimal values to place in the `mt_op` and `mt_count` fields of the structure used in the ioctl subroutine. The return value is the value of the *Count* parameter when the operation is successful.

Any other subcommand causes the **rmt** command to exit.

> **Note:** For the **R** and **W** subcommands, if the *Count* parameter specifies more bytes than the connection can handle, the data will be truncated to a size that can be handled.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Files

Item	Description
/usr/sbin/rmt	Contains the **rmt** command.
/usr/include/sys/errno.h	Describes the possible error numbers.

rmtcpip Command

Purpose

Removes the TCP/IP configuration for a host machine.

Syntax

rmtcpip

Description

The **rmtcpip** command removes TCP/IP configuration on a host machine. The basic functions of this command is:

- Removes the network interface configurations
- Restores **/etc/rc.tcpip** to the initial installed state

- Restores **/etc/hosts** to the initial installed state
- Removes the **/etc/resolv.conf** file
- Removes the default and static routes
- Sets the hostname to localhost
- Sets the hostid to 127.0.0.1
- Resets configuration database to the initial installed state

Note:

1. Any daemon which is commented out by default in **/etc/rc.tcpip**, but running at the time this command is issued, is stopped.
2. Your version of the **/etc/hosts** file is saved as **/etc/hosts.save** prior to the **/etc/hosts** file being restored to the originally installed state.
3. Your version of the **/etc/resolv.conf** file is saved as **/etc/resolv.conf.save** prior to the removal of the **/etc/resolv.conf** file.

Security

This command can only be run by root.

rmts Command

Purpose

Removes a thin server.

Syntax

rmts [**-f**] [**-v**] *ThinServer*

Description

The `rmts` command removes a thin server specified by *ThinServer* and created with the `mkts` command. If the thin server is running, the `rmts` command does not remove the thin server. Instead, it prints a message indicating that the thin server could not be removed. In this case, use the `-f` flag to terminate the thin server's session with a common image.

Flags

Item	Description
`-f`	Forces the removal of the thin server if the thin server is up and running.
`-v`	Enables verbose debug output when the `rmts` command runs.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

Access Control: You must have root authority to run the `rmts` command.

Examples

1. To remove a thin server named `lobo`, enter:

```
rmts lobo
```

Location

`/usr/sbin/rmts`

Files

Item	Description
`/etc/niminfo`	Contains variables used by NIM.

rmtun Command

Purpose

Deactivates operational tunnel(s) and optionally removes tunnel definition(s).

Syntax

rmtun -v 4|**6 -t** *tid_list* | **all** [**-d**]

Description

Use the **rmtun** command to deactivate an active tunnel(s) and optionally remove tunnel definition(s). It also will remove the auto-generated filter rules created for the tunnel by the **gentun** command when the tunnel definition is removed from the tunnel database.

Flags

Item	Description
all	Deactivates and optionally removes all the tunnel(s).
tid_list	The list of the tunnel(s) you want to deactivate. The tunnel IDs can be separated by "," or "-". You can use "-" to specify a range of IDs. For example, 1,3,5-7 specified there are five tunnel IDs in the list, 1, 3, 5, 6 and 7.
-d	Specifies that the tunnels are to be removed from the tunnel database. This is an optional flag.
-t	The list of the tunnel(s) you want to deactivate. If **-d** is specified, all the tunnel definitions in the list will also be removed from the tunnel database.
-v	The IP version of the tunnel. For the IP version 4 tunnel, use the value of **4**. For the IP version 6 tunnel, use the value of **6**.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

rmusil Command

Purpose

Removes an existing user-specified installation location (USIL) instance.

Syntax

rmusil **-R** *RelocatePath* **-r**

Description

The **rmusil** command removes an existing USIL instance.

Flags

Item	Description
-r	Removes the Software Vital Product Data (SWVPD) of an USIL instance.
-R *RelocatePath*	The path to an existing USIL location.

Note: The **rmusil** command only removes the USIL reference in the SWVPD. No files are removed in the USIL installation path.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/usr/sbin/rmusil	Contains the **rmusil** command.

rmuser Command

Purpose

Removes a user account.

Syntax

rmuser [**-R** *load_module*] [**-c**] [**-p**] *Name*

Description

The **rmuser** command removes the user account that is identified by the *Name* parameter. This command removes a user account's attributes without removing the user's home directory and files. The user name must exist. If you specify the **-c** flag, the **rmuser** command checks whether the user is logged in or has running processes before removing the user account. If the user is logged in or has running processes, the **rmuser** command fails. If you specify the **-p** flag, the **rmuser** command also removes passwords and other user authentication information from the /etc/security/passwd file.

For user accounts that are created with an alternate Identification and Authentication (I&A) mechanism, use the **-R** flag with the appropriate load module to remove that user. The load modules are defined in the **/usr/lib/security/methods.cfg** file.

Only the root user or users with UserAdmin authorization can remove administrative users. Administrative users are those users with **admin=true** set in the /**etc/security/user** file.

You can also use the System Management Interface Tool (SMIT) **smit rmuser** fast path to run this command.

Flags

Item	Description
-c	Verifies that the user is not logged in and does not have running processes before removing the user account.
-p	Removes user password information from the **/etc/security/passwd** file and removes the user keystore.
-R *load_module*	Specifies the loadable I&A module that is used to remove the user account.

Parameter

Item	Description
Name	Specifies a user account.

Exit Status

This command returns the following exit values:

Item	Description
0	The command ran successfully and all requested changes are made.
>0	An error occurred. The printed error message gives further details about the type of failure.

Security

Access Control: This command should grant execute (x) access only to the root user and members of the security group. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the root user with the **setuid** (SUID) bit set.

Files Accessed:

Mode	File
rw	**/etc/passwd**
rw	**/etc/security/passwd**
rw	**/etc/security/user**

Mode	File
rw	**/etc/security/user.roles**
rw	**/etc/security/limits**
rw	**/etc/security/environ**
rw	**/etc/security/audit/config**
rw	**/etc/group**
rw	**/etc/security/group**

Auditing Events:

Event	Information
USER_Remove	user

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To remove the user account `davis` and its attributes from the local system, enter:

```
rmuser davis
```

2. To remove the user account `davis` and all its attributes, including passwords and other user authentication information in the **/etc/security/passwd** file, type:

```
rmuser -p davis
```

3. To remove the user account `davis`, who was created with the LDAP load module, type:

```
rmuser -R LDAP davis
```

Files

Item	Description
/usr/sbin/rmuser	Contains the **rmuser** command.
/etc/security/passwd	Contains password information.
/etc/security/user	Contains the extended attributes of user accounts.
/etc/security/environ	Contains environment attributes of user accounts.
/etc/group	Contains the basic attributes of groups.

rmvfs Command

Purpose

Removes entries in the **/etc/vfs** file. The *VfsName* parameter is the name of a virtual file system. The **rmvfs** command takes one argument, the name of the virtual file system type to be removed from the file. If this *VfsName* entry exists, it is removed from the file.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To remove the `newvfs` entry, enter:

```
rmvfs newvfs
```

Files

Item	Description
/etc/vfs	Contains descriptions of virtual file system types.

rmvirprt Command

Purpose

Removes a virtual printer.

Syntax

rmvirprt -q *PrinterQueueName* **-d** *QueueDeviceName*

Description

The **rmvirprt** command removes the virtual printer assigned to the *PrinterQueueName* and *QueueDeviceName* variable value. The **rmvirprt** command also removes the System Management Interface Tool (SMIT) Object Database Manager (ODM) objects associated with the specified queue and queue device.

You can also use the System Management Interface Tool (SMIT) **smit rmvirprt** fast path to run this command.

Note: When the command **rmvirprt** is run from the command line, it does not remove the queue or queue device, nor does it check for any jobs running or queued on the specified queue and queue device. However, if SMIT is used to run this command interactively, the corresponding queue, queue device, and, optionally, printer device, are removed along with the virtual printer, if there are no jobs running or queued.

Flags

Item	Description
-d *QueueDeviceName*	Specifies the name of the queue device to which the virtual printer is assigned.
-q *PrinterQueueName*	Specifies the name of the print queue to which the virtual printer is assigned.

Examples

To remove the attribute values for the `mypro` virtual printer associated with the `proq` print queue, type:

```
rmvirprt  -d mypro  -q proq
```

Files

Item	Description
/etc/qconfig	Contains the configuration file.
/usr/sbin/rmvirprt	Contains the **rmvirprt** command.
/var/spool/lpd/pio/@local/custom/*	Contains the customized virtual printer attribute files.
/var/spool/lpd/pio/@local/ddi/*	Contains the digested virtual printer attribute files.

rmwpar Command

Purpose

Removes a workload partition.

Syntax

/usr/sbin/rmwpar [-F] [-p] [-s] [-v] *WparName*

Description

The **rmwpar** command deletes the specified workload partition from the system that includes the following tasks:

- Removing the workload partition's configuration data from the system's workload partition database
- Deleting the workload partition's file systems (if you do not specify the -p flag)
- Removing the workload partition's Workload Manager (WLM) profile

Without the -F flag, the **rmwpar** command stops the first time any part of the operation fails. If you specify the -F flag, the **rmwpar** command removes as much as possible. If the specified workload partition is active, the **rmwpar** command fails unless you specify the -s flag or the -F flag.

Flags

Item	Description
-F	Specifies that the **rmwpar** command must override or ignore most failures. It can be used to force the removal of broken workload partitions. This flag implies the -s flag.
-p	Removes a preservation removal that is assigned for the workload partition. The configured local file systems that are the logical volumes or subdirectories within the pre-existing logical volumes are not emptied or removed. This flag is for system workload partitions only. This flag cannot be used with rootvg workload partitions. File systems that are preserved by using this flag can be used with the following command to create a new workload partition that is attached to the m: `mkwpar -p`

Item	Description
-s	Stops the workload partition. This flag is equivalent to calling the **stopwpar** command before the **rmwpar** command. Use this flag to shut down and delete a workload partition in 1 step. If the **rmwpar** command was run with the -F flag specified, the **stopwpar** command can be run with the -F flag specified. If the **rmwpar** command is run on an active workload partition without the -s flag or the -F flag that is specified, the **rmwpar** command fails.
-v	Verbose mode.

Security

Access Control: Only the root user can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To remove the workload partition called "roy", enter:

```
rmwpar roy
```

2. To stop and remove the workload partition called "roy", preserving data on its file system, enter:

```
rmwpar -p -s roy
```

rmyp Command

Purpose

Removes the configuration for NIS.

Syntax

/usr/sbin/rmyp { **-s** | **-c** }

Description

The **rmyp** command removes everything from the system that is used to make NIS work. For example, the **rmyp** command removes all of the NIS maps and all of the entries in the **/etc/rc.nfs** file for the NIS daemons.

You could also use the System Management Interface Tool (SMIT) **smit rmyp** fast path to run this command. You can use the **System management interface tool (SMIT**) to run this command. To use SMIT, enter:

```
smit rmyp
```

Flags

Item	Description
-s	Removes the server configuration from the system.
-c	Removes the client configuration from the system.

rndc Command

Purpose

Name server control utility.

Syntax

rndc [**-b** *source-address*] [**-c** *config-file*] [**-k** *key-file*] [**-s** *server*] [**-p** *port*] [**-V**] [**-y** *key_id*] *command*

Description

The **rndc** command controls the operation of a name server. It supersedes the **ndc** utility that was provided in old BIND releases. If you run the **rndc** command with no command line options or arguments, it prints a short summary of the supported commands and the available options and their arguments.

The **rndc** command communicates with the name server over a TCP connection, sending commands authenticated with digital signatures. In the current versions of the **rndc** command and the **named** daemon, the only supported authentication algorithm is HMAC-MD5, which uses a shared secret on each end of the connection. This provides TSIG-style authentication for the command request and the name server's response. All commands sent over the channel must be signed by a **key_id** known to the server.

The **rndc** command reads a configuration file to determine how to contact the name server and decide what algorithm and key it must use.

Flags

Item	Description
-b *source-address*	Uses the *source-address* value as the source address for the connection to the server. Multiple instances are permitted to allow setting of both the IPv4 and IPv6 source addresses.
-c *config-file*	Uses the *config-file* value as the configuration file instead of the default, **/etc/rndc.conf**.
-k *key-file*	Uses the *key-file* value as the key file instead of the default, **/etc/rndc.key**. The key in **/etc/rndc.key** is used to authenticate commands sent to the server if the *config-file* argument does not exist.
-s *server*	Specifies the name or address of the server which matches a server statement in the configuration file for the **rndc** command. If you do not specify the *server* value, the host named by the default-server clause in the option statement of the configuration file is used.
-p *port*	Sends commands to TCP port instead of BIND 9's default control channel port, 953.
-V	Enables verbose logging.

Item	Description
-y *keyid*	Uses the *keyid* key from the configuration file. The *keyid* value must be known by the **named** daemon with the same algorithm and secret string in order for control message validation to succeed. If you do not specify the *keyid* value, the **rndc** command first looks for a key clause in the server statement of the server being used, or if no server statement is present for that host, then the default-key clause of the options statement. **Note:** The configuration file contains shared secrets which are used to send authenticated control commands to name servers. It cannot have general read or write access.

For the complete set of commands supported by the **rndc** command, see the BIND 9 Administrator Reference Manual or run the **rndc** command without arguments to see its help message.

Limitations

The **rndc** command only works with the **named9** daemon. The shared-secret for a *key_id* cannot be provided without using the configuration file.

rndc-confgen Command

rndc-confgen [**-a**] [**-b** *keysize*] [**-c** *keyfile*] [**-h**] [**-k** *keyname*] [**-p** *port*] [**-r** *randomfile*] [**-s** *address*] [**-t** *chrootdir*] [**-u** *user*]

Purpose

Generates configuration files for the **rndc** command.

Syntax

Description

The **rndc-confgen** command generates configuration files for the **rndc** command. You can use this command as a convenient alternative to writing the **rndc.conf** file, the corresponding controls, and key statements in **named.conf** by hand. You can run the **rndc-confgen** command with the **-a** flag to set up a **rndc.key** file. Doing this avoids the need for a **rndc.conf** file and a controls statement.

Flags

Item	Description
-a	Performs automatic **rndc** configuration. This creates a file **rndc.key** in **/etc** (or whatever **sysconfdir** was specified as when BIND was built) that is read by both the **rndc** command and the **named** daemon on startup. The **rndc.key** file defines a default command channel and authentication key allowing the **rndc** command to communicate with the **named** daemon on the local host with no further configuration.
-b *keysize*	Specifies the size of the authentication key in bits. Must be between 1 and 512 bits. The default is 128.
-c *keyfile*	Used with the **-a** flag to specify an alternate location for **rndc.key**.
-h	Prints a short summary of the options and arguments of the **rndc-confgen** command.
-k *keyname*	Specifies the key name of the **rndc** authentication key. This must be a valid domain name. The default is **rndc-key**.

Item	Description
-p *port*	Specifies the command channel port where the **named** daemon listens for connections from **rndc**. The default is 953.
-r *randomfile*	Specifies a source of random data for generating the authorization. If the operating system does not provide a **/dev/random** or equivalent device, the default source of randomness is keyboard input. The *randomfile* argument specifies the name of a character device or file containing random data to be used instead of the default. The `keyboard` value indicates that keyboard input must be used.
-s *address*	Specifies the IP address where the **named** daemon listens for command channel connections from **rndc**. The default is the loopback address 127.0.0.1.
-t *chrootdir*	Used with the **-a** flag to specify a directory where the **named** daemon runs chrooted. An additional copy of the **rndc.key** will be written relative to this directory so that it will be found by the chrooted **named**.
-u *user*	Used with the **-a** flag to set the owner of the **rndc.key** file generated. If the **-t** flag is also specified, only the file in the chroot area has its owner changed.

Examples

1. To use the **rndc** command with no manual configuration, enter the following command:

```
rndc-confgen -a
```

2. To print a sample **rndc.conf** file and have corresponding controls and key statements to be manually inserted into the **named.conf** file, enter the following command:

```
rndc-confgen
```

roffbib Command

Purpose

Prints a bibliographic database.

Syntax

roffbib [**-m** *Macro*] [**-x**] [*FormatFlags*] [*Database*...]

Description

The **roffbib** command prints out all records that are in a bibliographic database format rather than in a format for footnotes or endnotes. Generally, the command is used as a filter for the **troff** command, in particular, the **-e**, **-h**, **-n**, **-o**, **-r**, **-s**, and **-T** flags.

If abstracts or comments are entered following the **%X** key field, they are formatted into paragraphs for an annotated bibliography. Several **%X** fields can be given if several annotation paragraphs are desired.

Parameters

Item	Description
FormatFlags	Accepts most of the **nroff** command flags, especially the **-e**, **-h**, **-n**, **-o**, **-r**, **-s**, and **-T** flags.
Database	Stores a bibliographic database of all records.

Flags

Item	Description
-m *Macro*	Specifies a file that contains a user-defined set of macros. There should be a space between the **-m** flag and the macro. This set of macros replaces the ones defined in the **/usr/share/lib/tmac/tmac.bib** file. Users can rewrite macros to create customized formats.
-x	Suppresses the printing of abstracts or comments that are entered following the **%X** field key.

Examples

Following is an example of the **roffbib** command used in conjunction with the html

rolelist Command

Purpose

Displays role information for a user or process.

Syntax

rolelist [**-a**] [**-e** | **-u** *username* | **-p** *PID*]

Description

The **rolelist** command provides role and authorization information to the invoker about their current roles or the roles assigned to them. If no flags or arguments are specified, the **rolelist** command displays the list of roles assigned to the invoker on the real user ID with the text description of each role if one is provided in the roles database. Specifying the **-e** flag outputs information about the current effective active role set for the session. If the invoker is not currently in a role session and specifies the **-e** flag, no output is displayed. Specifying the **-a** flag displays the authorizations associated with the roles instead of the text description.

The **rolelist** command also allows a privileged user to list the role information for another user or for a process. Specifying a user name with the **-u** flag allows a privileged user to list the roles assigned to another user. The active role set of a given user cannot be determined because the user can have multiple active role sessions. Therefore, if the **-u** flag is specified, the **-e** flag is not allowed. Specifying a process ID with the **-p** flag allows a privileged user to display the roles associated with a process. The command fails immediately if invoked by a non-privileged user when the **-u** or **-p** flag is specified.

The authorization information displayed by the **rolelist** command is retrieved from the kernel security tables. The information can differ with the current state of the roles database if it is modified after the kernel security tables are updated.

Flags

Item	Description
-a	Displays the authorizations assigned to each role instead of the role description.
-e	Displays information about the effective active role set of the session.
-u *username*	Displays role information for the specified user.
-p *PID*	Displays role information of the specified process.

Security

All users can run the **rolelist** command. To query the role information of another user or a process, the following authorizations are required.

Item	Description
aix.security.role.list	Required to invoke the command on another user.
aix.security.proc.role.list	Required to list the roles associated with a process.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files Accessed

Files	Mode
/etc/security/user.roles	r
/etc/security/roles	r

Examples

1. To display the list of roles that assigned to you and their text descriptions, use the following command:

   ```
   rolelist
   ```

   Information similar to the following example is displayed:

   ```
   UserAdmin        User Administrator
   RoleAdmin        Role Administrator
   FSAdmin          File System Administrator
   ```

2. To display the authorizations associated with the assigned roles, use the following command:

   ```
   rolelist -a
   ```

   Information similar to the following example is displayed:

   ```
   UserAdmin        aix.security.user
   RoleAdmin        aix.security.role
   FSAdmin          aix.security.fs
   ```

3. As a privileged user, use the following command to display the roles assigned to a specific user :

   ```
   rolelist -u user1
   ```

   Information similar to the following example is displayed:

   ```
   SysInfo          System Information Retrieval
   ```

roleqry Command

Purpose

Queries the usage of roles over a time period.

Syntax

roleqry {**-c** [**-s**] | **-q** [**-F** <trailListfile>] [**-t** <time_period_in_days>] } user

Description

The **roleqry** command queries information about the roles used by a user over a specified time frame.

When the **-c** flag is specified, the user is configured for the auditing of role information and authorization information. A **rbacqry** class is added to the /etc/security/audit/config file with events for auditing authorizations and roles. If the user is already being audited, a user entry present in the configuration file, then the **rbacqry** class is added to the user. Otherwise the username is added to the /etc/security/audit/config with the **rbacqry** class parameter. If the **-s** flag is specified, the user is enabled for audit. If the audit subsystem is already turned on, then it is restarted. If the audit system is already turned off, then the audit subsystem is started.

When the **-q** flag is specified, the audit data is queried for role information. When the **-t** flag is specified, the usage of roles from the date to the current system date is queried and obtained. Without **-t** falg, role usage over the period from which auditing was enabled for that user is obtained. The command displays the entire set of roles used during the time frame.

Note: The **roleqry** commands make use of the auditing feature in AIX. Auditing has to be turned on, audit configuration for the user setup and the audit data collected during the specified time frame for the **roleqry** command to work as expected.

Flags

Item	Description
-c	Use this flag to configure the user for auditing of role usage.
-s	Use this flag to start auditing subsystem if it is turned off. Shutdown and restart auditing subsystem if it is already turned on.
-q	Use this flag to query audit data for role usage over a time period.
-F	Use this flag to read the names of the audit trails to obtain audit information from the *trailListFile*. The names of audit trail files should be one name per line of text. If the **-F** flag is not specified, the system “audit/trail file is taken by default as the file to obtain audit information from.
-t	Use this flag to specify the number of days from the current date to get the authorization usage.

Exit Status

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: This command should grant execute (x) access to the root user.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files:

- /etc/security/roles
- audit/trail

Examples

1. To query roles used by Bob run the following command:

```
roleqry -q Bob
```

2. To query roles used by Simon for the past 20 days run the following command:

```
roleqry -q -t 20 Simon
```

rolerpt Command

Purpose

Reports the security capabilities of roles.

Syntax

rolerpt [**-R** <load_module>] [**-C**] [**-c** | **-f**] { "ALL" | role1, role2, ... | **-a** }

rolerpt [**-R** <load_module>] [**-C**] [**-u**] { "ALL" | role1, role2, ... }

Description

The **rolerpt** command reports capability information of roles such as privileged commands, privileged files, and user information.

Either of **–c**, **-f,** or **–u** flags can be specified. When the **-c** flag is specified, the privileged commands present in the `/etc/security/privcmds` database that can be run by virtue of the roles is listed. When the **–f** flag is specified, the list of privileged files present in the `/etc/security/privfiles` database that can be accessed by users that are assigned to the roles is displayed.

When the **–u** flag is specified, the list of users with roles are displayed based on the Loadable Authentication Model (LAM) 's that is configured in the `/etc/nscontrol.conf` database. The **–u** flag can be used only by a root user or a privileged user that is authorized for the **`rolerpt`** command. Only root user or the authorized user with **aix.security.role.list** authorization can view reports that display capabilities for roles that are not held by them.

When no flag is specified, all the capability information such as commands, privileged files, and user information for the role is displayed.

The **–a** flag specifies the capabilities of the active roles. The **–u** flag cannot be used with the **–a** flag. The root user or the authorized user can specify the **ALL** keyword to display capabilities for all the roles on the system.

The **`rolerpt`** command accepts inputs such as **-a** flag to specify the active roles, the **ALL** keyword, or a comma-separated list of role names. When no role name is specified, all the capability information such as commands, privileged files, and user information that is associated with the roles of the invoker is displayed.

Flags

Item	Description
-a	Specifies that report on only capabilities of active roles is to be obtained.
-c	Specifies that a report of privileged commands executable by the roles is to be obtained.

Item	Description
-C	Displays the role attributes in colon-separated records, as displayed in the following example: `#role:attribute1:attribute2: ...` `role1:value1:value2: ...` `role2:value1:value2: ...`
-f	Specifies that a report of privileged file information accessible to the roles is to be obtained.
-R	Specifies the loadable module to obtain the report of roles capabilities from.
-u	Specifies that a report of authorized user information that is assigned to the roles is to be obtained.

Exit status

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: This command must grant execute (x) access to all users. The **–u** flag can be used by the root user or authorized users with **aix.security.role.list** authorization or **aix.security.user.list** authorization. Only root or the authorized user with **aix.security.role.list** authorization can specify the **ALL** keyword and view reports of capabilities of roles that are not held by them.

Attention RBAC users and Trusted AIX users: This command does privileged operations. Only privileged users can run privileged operations For more information about authorizations and privileges, review the Privileged Command Database topic. For a list of privileges and the authorizations that are associated with this command, review the **lssecattr** command or the **getcmdattr** subcommand.

Files

- /etc/security/roles
- /etc/security/authorizations
- /etc/security/privcmds
- /etc/security/privfiles

Examples

1. To report the commands that are associated with the role ManageAllUsers, run the following command:

   ```
   rolerpt –c ManageAllUsers
   ```

2. To report capabilities of active roles that are, the authorization, command, and privileged file information run the following command:

   ```
    rolerpt –a
   ```

3. To report all capabilities of role ManageAllUsers in a colon separated format, run the following command:

   ```
   rolerpt –C ManageAllUsers
   Information similar to the following appears:
   ```

```
#role:commands:privfiles:users
   ManageAllUsers:/usr/bin/lsuser,/usr/bin/mkuser:/var/adm/sulog:Bob,Simon
```

rollback Command

Purpose

Reverts a JFS2 file system to a point-in-time snapshot.

Syntax

To rollback to an external snapshot

```
rollback [-s ] [ -v ] [-c] snappedFS snapshotObject
```

To rollback to an internal snapshot

```
rollback [ -v ] -n snapshotName snappedFS
```

Description

The `rollback` command is an interface to revert a JFS2 file system to a point-in-time snapshot. The *snappedFS* parameter must be unmounted before the `rollback` command is run and remains inaccessible for the duration of the command. Any snapshots that are taken after the specified snapshot (*snapshotObject* for external or *snapshotName* for internal) are removed. The associated logical volumes are also removed for external snapshots.

If the `rollback` command is interrupted for any reason, the *snappedFS* parameter remains inaccessible until the command is restarted and completes. A restarted `rollback` must target the same *snapshotObject* or *snapshotName* as the initial command.

Flags

Item	Description
-c	If specified, `rollback` continues even if read or write errors are observed when restoring the *snappedFS* from the snapshot. If you do not specify the **-c** flag, an error message is issued and the rollback stops. Run the **fsck** command in this case.
-n *snapshotName*	Specifies the name of the internal snapshot to use for the rollback.
-s	If specified, any logical volumes associated with snapshots removed by `rollback` will be preserved. The snapshots are still deleted.
-v	This is the verbose option and causes a count of restored blocks to be printed as the rollback progresses.

Parameters

Item	Description
snappedFS	The JFS2 system to roll back.
snapshotObject	The logical volume of the external snapshot to revert to.

Examples

To roll back the `/home/janet/sb` file system to the external snapshot on logical volume `/dev/snapsb`, enter:

```
rollback /home/janet/sb /dev/snapsb
```

Location

Item	Description
`/usr/sbin/rollback`	Contains the `rollback` command.

route Command

Purpose

Manually manipulates the routing tables.

Syntax

route [**-f**] [**-n**] [**-q**] [**-C**] [**-v**] *Command* [*Family*] [[**-net** | **-host**] *Destination* [**-prefixlen** *n*] [**-netmask** [*Address*]] *Gateway*] [*Arguments*] [**-i**] [**-@** *WparName*]

Description

The **route** command allows you to make manual entries into the network routing tables. The **route** command distinguishes between routes to hosts and routes to networks by interpreting the network address of the *Destination* variable, which can be specified either by symbolic name or numeric address. The **route** command resolves all symbolic names into addresses, using either the **/etc/hosts** file or the network name server.

Routes to a particular host are distinguished from those to a network by interpreting the Internet address associated with the destination. The optional phs **-net** and **-host** force the destination to be interpreted as a network or a host, respectively. If the destination has a local address part of INADDR_ANY or if the destination is the symbolic name of a network, then the route is assumed to be to a network; otherwise, it is presumed to be a route to a host.

For example, `128.32` is interpreted as `-host 128.0.0.32`; `128.32.130` is interpreted as `-host 128.32.0.130`; `-net 128.32` is interpreted as `128.32.0.0`; and `-net 128.32.130` is interpreted as `128.32.130.0`.

If the route is by way of an interface rather than through a gateway, the **-interface** argument should be specified. The specified gateway is the address of the host on the common network, indicating the interface to be used for transmission.

The **-netmask** argument must be followed by an address parameter (to be interpreted as a network mask). One can override the implicit network mask generated in the **-inet** case by making sure this option follows the *Destination* parameter.

All symbolic names specified for a destination or gateway are looked up first as a host name, using the **gethostbyname** subroutine. If this fails, the **getnetbyname** subroutine is then used to interpret the name as a network name.

> **Note:** Route uses a routing socket and the new message types RTM_ADD, RTM_DELETE, and RTM_CHANGE. As such, only the root user may modify the routing tables.

If the **flush** or **-f** command is specified, route will "flush," or clear, the routing tables of all gateway entries. One can choose to flush only those routes whose destinations are of a given address family, by specifying an optional ph describing which address family.

The **netstat -r** command displays the current routing information contained in the routing tables.

Note: You must use the same set of commands (**route**, **smitty**, **ifconfig** and **chdev** commands) for creating or deleting the routing table. If you create the routing table by using the **smitty** or **chdev** command, and delete it by using the **route** command, the route entry is not deleted from Object Data Manager (ODM), and if the system gets restarted, the routing table gets builds from ODM, due to which you can see the same route entry again.

Flags

Item	Description
-f	Purges all entries in the routing table that are not associated with network interfaces.
-i	Enables workload-partition-specific routing for the workload partition (WPAR). By default, outgoing network traffic from a WPAR is routed as if it were being sent from the global environment: • Traffic between addresses that are hosted on the same global system is sent through the loopback interface. • Routing table entries that are configured in the global system, including the default route, are used to transmit workload partition traffic. If you enable WPAR-specific routing by specifying the **-i** flag, the WPAR creates and uses its own routing table for the outgoing traffic. Routing entries are created automatically for each of the network addresses of the WPAR to accommodate broadcast, loopback, and subnet routes.
-n	Displays host and network names numerically, rather than symbolically, when reporting results of a flush or of any action in verbose mode.
-q	Specifies quiet mode and suppresses all output.
-C	Specifies preference for **ioctl** calls over routing messages for adding and removing routes.
-v	Specifies verbose mode and prints additional details.
-net	Indicates that the *Destination* parameter should be interpreted as a network.
-netmask	Specifies the network mask to the destination address. Make sure this option follows the *Destination* parameter.
-host	Indicates that the *Destination* parameter should be interpreted as a host.
-prefixlen *n*	Specifies the length of a destination prefix (the number of bits in the netmask).
-@*WparName*	Displays the network statistics that are associated with the WPAR that is, (@*WparName* flag). If the @*WparName* flag is not specified, the network statistics for all the WPARs are displayed.

The route default is a host (a single computer on the network). When neither the **-net** parameter nor the **-host** parameter is specified, but the network portion of the address is specified, the route is assumed to be to a network. The host portion of the address is 0 (zero).

Parameters

Item	Description
Arguments	Specifies one or more of the following arguments. Where *n* is specified as a variable to an argument, the value of the *n* variable is a positive integer. **-active_dgd** Enables Active Dead Gateway Detection on the route. **-cloning** Clones a new route. **-genmask** Extracts the length of TSEL, which is used for the generation of cloned routes. **-interface** Manipulates interface routing entries. **-rtt *n*** Specifies round-trip time. **-rttvar *n*** Specifies round-trip time variance.

Item	Description
	-sendpipe *n* Specifies send-window size. **-recvpipe *n*** Specifies receive-window size. **-allowgroup *gid*** Specifies a group ID that is allowed to use the route. The group ID will be added to a list of allowed groups or deleted from a list of denied groups. **-denygroup *gid*** Specifies a group ID that is not allowed to use the route. The group ID will be added to a list of denied groups or deleted from a list of allowed groups. **-stopsearch** Stops searching if a routing table lookup matches the route, but it is not allowed to use the route due to group routing restrictions. **-mtu *n*** Specifies maximum transmission unit for this route. Will override interface mtu for TCP applications as long as it does not exceed maximum mtu for the interface. This flag has no affect on mtu for applications using UDP. **-hopcount *n*** Specifies maximum number of gateways in the route. **-policy *n*** Specifies the policy to be used for Multipath Routing. *n* is number between 1 and 5 where these numbers mean the following: 1. Weighted Round-Robin 2. Random 3. Weighted Random 4. Lowest Utilization 5. Hash-based If the policy is not explicitly set and multipath routing is used, then the global **no** command option called **mpr_policy** determines the policy that will be used. The default policy is Weighted Round Robin which behaves just like Round-Robin when the weights are all 1. Although the Default policy is Weighted Round-Robin, when the policy is not set, then the network option **mpr_policy** takes precedence. On the other hand, if the policy is explicitly set to WRR then this setting overrides the **mpr_policy** setting. For more information about these policies, see the **no** command. **-weight *n*** Specifies the weight of the route that will be used for the Weighted policies with the Multipath Routing feature.

Item	Description
	-expire *n* Specifies expiration metrics used by routing protocol **-ssthresh *n*** Specifies outbound gateway buffer limit. **-lock** Specifies a meta-modifier that can individually lock a metric modifier. The **-lock** meta-modifier must precede each modifier to be locked. **-lockrest** Specifies a meta-modifier that can lock all subsequent metrics. **-if *ifname*** Specifies the interface (en0, tr0 ...) to associate with this route so that packets will be sent using this interface when this route is chosen. **-xresolve** Emits a message on use (for external lookup). **-iface** Specifies that the destination is directly reachable. **-static** Specifies the manually added route. **-nostatic** Specifies the pretend route that is added by the kernel or daemon. **-reject** Emits an ICMP unreachable when matched. **-blackhole** Silently discards packets during updates. **-proto1** Sets protocol specific routing flag number 1. **-proto2** Sets protocol specific routing flag number 2.
Command	Specifies one of six possibilities: **add** Adds a route. **flush or -f** Removes all routes. **delete** Deletes a specific route. **change** Changes aspects of a route (such as its gateway). **monitor** Reports any changes to the routing information base, routing lockup misses, or suspected network partitionings. **get** Lookup and display the route for a destination. **set** Set the policy and weight attributes of a route.
Family	Specifies the address family. The **-inet** address family is the default. The **-inet6** family specifies that all subsequent addresses are in the inet6 family.

Item	Description
Destination	Identifies the host or network to which you are directing the route. The *Destination* parameter can be specified either by symbolic name or numeric address.
Gateway	Identifies the gateway to which packets are addressed. The *Gateway* parameter can be specified either by symbolic name or numeric address.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To establish a route so that a computer on one network can send a message to a computer on a different network, type:

   ```
   route add 192.100.201.7 192.100.13.7
   ```

   The `192.100.201.7` address is that of the receiving computer (the *Destination* parameter). The `192.100.13.7` address is that of the routing computer (the *Gateway* parameter).
2. To establish a route so you can send a message to any user on a specific network, type:

   ```
   route add -net 192.100.201.0 192.100.13.7
   ```

   The `192.100.201.0` address is that of the receiving network (the *Destination* parameter). The `192.100.13.7` address is that of the routing network (the *Gateway* parameter).
3. To establish a default gateway, type:

   ```
   route add 0 192.100.13.7
   ```

   The value `0` or the default ph for the *Destination* parameter means that any packets sent to destinations not previously defined and not on a directly connected network go through the default gateway. The `192.100.13.7` address is that of the gateway chosen to be the default.
4. To clear the host gateway table, type:

   ```
   route -f
   ```
5. To add a route specifying weight and policy information, type:

   ```
   route add  192.158.2.2 192.158.2.5 -weight 5 -policy 4
   ```
6. To set the weight and policy attributes of a preexisting route, type:

   ```
   route set 192.158.2.2 192.158.2.5 -weight 3 -policy
   ```

routed Daemon

Purpose

Manages network routing tables.

Syntax

Note: Use SRC commands to control the **routed** daemon from the command line. Use the **gated** daemon, which supports all TCP/IP gateway protocols, the **routed** daemon only implements the Routing

Information Protocol (RIP). Do not use the **routed** daemon when Exterior Gateway Protocol (EGP), Simple Network Management Protocol (SNMP), or Distributed Computer Network Local-Network Protocol routing is needed. Use the /etc/gateways file for information about these distant and external gateways.

The **/etc/gateways** file contains information about routes through distant and external gateways to hosts and networks that should be advertised through RIP. These routes can be either static routes to specific destinations or default routes for use when a static route to a destination is unknown. The format of the **/etc/gateways** file is:

{ *net* | *host* } *name1 gateway name2 metric* { **passive** | **active** | **external** }

When a gateway specified in the **/etc/gateways** file supplies RIP routing information, it should be marked as active. Active gateways are treated like network interfaces. That is, RIP routing information is distributed to the active gateway. If no RIP routing information is received from the gateway for a period of time, the **routed** daemon deletes the associated route from the routing tables.

A gateway that does not exchange RIP routing information should be marked as passive. Passive gateways are maintained in the routing tables indefinitely. Information about passive gateways is included in any RIP routing information transmitted.

An external gateway is identified to inform the **routed** daemon that another routing process will install such a route and that the **routed** daemon should not install alternative routes to that destination. External gateways are not maintained in the routing tables and information about them is not included in any RIP routing information transmitted.

Note: Routes through external gateways must be to networks only.

The **routed** daemon can also perform name resolution when routing to different networks. For example, the following command adds a route to the network called `netname` through the gateway called `host1`. The `host1` gateway is one hop count away.

```
route add net netname host1 1
```

To perform network name resolution, the **routed** daemon uses the **/etc/networks** file to get information on the network addresses and their corresponding names. To perform host name resolution, the **routed** daemon must take additional steps before the routing is complete. First the daemon checks for the existence of the **/etc/resolv.conf** file. This file indicates whether the host is running under a domain name server, and if so, gives the IP address of the host machine running the **named** daemon.

If the **/etc/resolv.conf** file does not exist, the **routed** daemon uses the **/etc/hosts** file to find the host for which it is routing.

The **routed** daemon should be controlled using the System Resource Controller (SRC) or the System Management Interface Tool (SMIT). Entering the **routed** daemon at the command line is not recommended.

Manipulating the routed Daemon with the System Resource Controller

The **routed** daemon is a subsystem controlled by the System Resource Controller (SRC). The **routed** daemon is a member of the SRC **tcpip** system group. This daemon is disabled by default and can be manipulated by the following SRC commands:

Item	Description
startsrc	Starts a subsystem, group of subsystems, or subserver.
stopsrc	Stops a subsystem, group of subsystems, or subserver.
tracesoff	Disables tracing of a subsystem, group of subsystems, or subserver.
lssrc	Gets the status of a subsystem, group of subsystems, or subserver.

Signals

The following signals have the specified effect when sent to the **routed** process using the **kill** command:

Item	Description
SIGINT	Restarts the **routed** daemon and flushes the routing table.
SIGHUP, **SIGTERM**, or **SIGQUIT**	Broadcasts RIP packets with hop counts set to infinity. These signals disable the local host as a router. After a second **SIGHUP**, **SIGTERM**, or **SIGQUIT** signal, the **routed** daemon terminates.
SIGUSR1	Turns packet tracing on or, if packet tracing is already on, steps up the tracing one level. The first level traces transactions only. The second level traces transactions plus packets. The third level traces the packet history, reporting packet changes. The fourth level traces packet contents. This command increments the level of tracing through four levels.
SIGUSR2	Turns packet tracing off.

Flags

Item	Description
-d	Enables additional debugging information, such as bad packets received, to be logged.
-g	Runs the routing daemon on a gateway host. The **-g** flag is used on internetwork routers to offer a route to the default destination.
-q	Prevents the **routed** daemon from supplying routing information regardless of whether it is functioning as an internetwork router. The **-q** flag indicates "quiet". Do not use the **-q** flag and the **-s** flag together.
-s	Supplies routing information regardless of whether it is functioning as an internetwork router. The **-s** flag indicates "supply". Do not use the **-q** flag and the **-s** flag together.
-t	Writes all packets sent or received to standard output or to the file specified in the *LogFile* parameter. The **routed** daemon remains under control of the controlling terminal that started it. Therefore, an interrupt from the controlling terminal keyboard stops the **routed** process.

Examples

1. To start the **routed** daemon manually, type:

   ```
   startsrc -s routed -a "-s"
   ```

   Note: The **routed** daemon is not started by default at each system startup. Use the **rc.tcpip** file format and a System Resource Controller (SRC) command to start the **routed** daemon. You can also start the **routed** daemon using the System Management Interface Tool (SMIT).

   The **-s** flag causes the **routed** daemon to return routing information regardless of whether the **routed** daemon is an internetwork router.
2. To stop the **routed** daemon, type the following:

   ```
   stopsrc -s routed
   ```
3. To get a short-status report from the **routed** daemon, type the following:

   ```
   lssrc -s routed
   ```

   This command returns the name of the daemon, the process ID of the daemon, and the state of the daemon (active or inactive).
4. To enable tracing for **routed** daemon, type the following:

```
traceson -s routed
```

This command enables socket-level debugging. Use the html

rpc.pcnfsd Daemon

Purpose

Handles service requests from PC-NFS (Personal Computers Network File System) clients.

Syntax

/usr/sbin/rpc.pcnfsd

Description

The **rpc.pcnfsd** daemon handles requests from PC-NFS clients for authentication services on remote machines. These services include authentication for mounting and for print spooling. The PC-NFS program allows personal computers running DOS to be networked with machines running NFS. The **rpc.pcnfsd** daemon supports Versions 1 and 2 of the **pcnfsd** protocol.

When a PC-NFS client makes a request, the **inetd** daemon starts the **rpc.pcnfsd** daemon (if the **inetd.conf** file contains the appropriate entry). The **rpc.pcnfsd** daemon reads the **umask**specifications. A record of logins is appended to the **exportfs** command and the **enq** command. The daemon adopts the identity of the personal computer user to execute the print request command. Because constructing and executing the command involves user ID privileges, the **rpc.pcnfsd** daemon must be run as a root process.

All print requests from clients include the name of the printer to be used. The printer name is represented by queue and device definitions in the **/etc/qconfig** file. Additionally, the **rpc.pcnfsd** daemon provides a method for defining PC-NFS virtual printers recognized only by **rpc.pcnfsd** clients. Each PC-NFS virtual printer is defined in the **/etc/pcnfsd.conf** file with a line similar to the following:

```
printer Name AliasFor Command
```

In this format, `Name` specifies the name of the printer to be defined, and `AliasFor` is the name of the existing printer that will do the work. For example, a request to show the queue for `Name` translates into a queue command on the `AliasFor` printer. To define a printer `Name` with no existing printer, use a single - (minus sign) in place of the `AliasFor` parameter. The `Command` parameter specifies a command run when a file is printed on the `Name` printer. This command is executed by the Bourne shell, using the **-c** option. For complex operations, replace the `Command` parameter with an executable shell script.

The following list of tokens and substitution values can be used in the *Command* parameter:

Token	Substitution Value
$FILE	The full path name of the print data file. After the command has executed, the file is unlinked.
$USER	The user name of the user logged-in to the client.
$HOST	The host name of the client system.

Examples

The following example **/etc/pcnfsd.conf** file configures a virtual printer on the first line and a null device for testing on the second line:

```
printer rotated lw /bin/enscript -2r $FILE
printer test - /usr/bin/cp $FILE /usr/tmp/$HOST-$USER
```

The first line stipulates that if a client system prints a job on the `rotated` printer, the `enscript` utility is called to preprocess the `$FILE` file. The `-2r` option causes the file to be printed in two-column, rotated format on the default PostScript printer. If a client requests a list of the print queue for the `rotated` printer, the **rpc.pcnfsd** daemon translates this request into a request for a similar listing for the `lw` printer.

The second line establishes a printer test. Files sent to the `test` printer are copied into the **/usr/tmp** directory. Requests to the `test` printer to list the queue, check the status, or perform similar printer operations, are rejected because - (minus sign) is specified in place of the *AliasFor* parameter.

Files

Item	Description
/etc/pcnfsd.conf	Contains the **rpc.pcnfsd** daemon configuration file.
/var/spool/pcnfs	Contains the default print-spooling directory.

rpcgen Command

Purpose

Generates C code to implement an RPC protocol.

Syntax

To Generate Four Types of Output Files for a File

/usr/bin/rpcgen *InputFile*

To Generate a Specific Output File for a File

rpcgen { **-c** | **-h** | **-l** | **-m** } [**-o** *OutputFile*] [*InputFile*]

To Generate a Server-Side File for TCP or UDP

rpcgen { **-s** *Transport ...* } [**-o** *OutputFile*] [*InputFile*]

Description

The **rpcgen** command generates C code to implement a Remote Procedure Call (RPC) protocol. The input to the **rpcgen** command is a language similar to C language known as RPC Language.

The first syntax structure is the most commonly used form for the **rpcgen** command where it takes an input file and generates four output files. For example, if the *InputFile* parameter is named **proto.x**, then the **rpcgen** command generates the following:

Item	Description
proto.h	Header file
proto_xdr.c	XDR routines
proto_svc.c	Server-side stubs
proto_clnt.c	Client-side stubs

Use the other syntax structures when you want to generate a particular output file rather than all four output files.

The **cpp**command, a C preprocessor, is run on all input files before they are actually interpreted by the **rpcgen** command. Therefore, all the **cpp** directives are legal within an **rpcgen** input file. For each type of output file, the **rpcgen** command defines a special **cpp** symbol for use by the **rpcgen** programmer:

Item	Description
RPC_HDR	Defined when compiling into header files
RPC_XDR	Defined when compiling into XDR routines
RPC_SVC	Defined when compiling into server-side stubs
RPC_CLNT	Defined when compiling into client-side stubs

In addition, the **rpcgen** command does some preprocessing of its own. Any line beginning with a **%** (percent sign) passes directly into the output file, uninterpreted by the **rpcgen** command.

To create your own XDR routines, leave the data types undefined. For every data type that is undefined, the **rpcgen** command assumes that a routine exists by prepending **xdr_** to the name of the undefined type.

Notes:

1. Nesting is not supported. As a work-around, structures can be declared at top-level with their names used inside other structures in order to achieve the same effect.
2. Name clashes can occur when using program definitions since the apparent scoping does not really apply. Most of these can be avoided by giving unique names for programs, versions, procedures, and types.
3. To program to the TIRPC interfaces, and allow the use of multi-threaded RPC applications use the **tirpcgen** command. It will also be necessary to define the preprocessor variable **_AIX_TIRPC** in the Makefile as well as the **libtli.a** (**-ltli**) specification. **tirpcgen** is a temporary name for a new **rpcgen** command that will replace **rpcgen** in a future version the operating system.

Flags

Item	Description
-c	Compiles into XDR routines.
-h	Compiles into C-data definitions (a header file).
-l	Compiles into client-side stubs.
-m	Compiles into server-side stubs, but does not generate a main routine. This option is useful for doing call-back routines and for writing a main routine to do initialization.
-o *OutputFile*	Specifies the name of the output file. If none is specified, standard output is used.
-s *Transport*	Compiles into server-side stubs, using given transport. The supported transports are udp and `tcp`. This flag can be run more than once to compile a server that serves multiple transports.

rpcinfo Command

Purpose

Reports the status of Remote Procedure Call (RPC) servers.

Syntax

To Display a List of Statistics

/usr/bin/rpcinfo [**-m** | **-s**] [*Host*]

To Display a List of Registered RPC Programs

/usr/bin/rpcinfo -p [*Host*]

To Report Transport

/usr/bin/rpcinfo -T *transport Host Prognum* [*Versnum*]

To Display a List of Entries

/usr/bin/rpcinfo -l [**-T** *transport*] *Host Prognum Versnum*

To Report Program Status using UDP

/usr/bin/rpcinfo [**-n** *PortNum*] **-u** *Host Prognum* [*Versnum*]

To Report Program Status using TCP

/usr/bin/rpcinfo [**-n** *PortNum*] **-t** *Host Prognum* [*Versnum*]

To Report Program Status

/usr/bin/rpcinfo -a *ServAddress* **-T** *transport Host Prognum* [*Versnum*]

To Display All Hosts Running a Specified Program Version

/usr/bin/rpcinfo [**-b**] [**-T** *transport*] *Prognum Versnum*

To Delete Registration of a Service

/usr/bin/rpcinfo [**-a -d**] [**-T** *transport*] *Prognum Versnum*

Description

The **rpcinfo** command makes an RPC call to an RPC server and reports the status of the server. For instance, this command reports whether the server is ready and waiting or not available.

The program parameter can be either a name or a number. If you specify a version, the **rpcinfo** command attempts to call that version of the specified program. Otherwise, the **rpcinfo** command attempts to find all the registered version numbers for the program you specify by calling version 0 (zero) and then attempts to call each registered version. (Version 0 is presumed not to exist. If it does exist, the **rpcinfo** command attempts to obtain this information by calling an extremely high version number instead.)

Flags

Item	Description
-a	Specifies the complete IP address and port number of the host.
-b	Makes an RPC broadcast to procedure 0 of the specified prognum and versnum and reports all hosts that respond. If *transport* is specified, it broadcasts its request only on the specified *transpor*t. If broadcasting is not supported by any *transport*, an error message is printed. Using broadcasting (**-b** flag) should be limited because of the possible adverse effect on other systems.
-d	Deletes registration for the RPC service of the specified prognum and versnum. If transport is used, unregister the service only on that transport, otherwise unregister the service on all the transports where it was registered. This option can be exercised only by the root user.
-l	Displays a list of entries with the specified prognum and versnum on the specified host. Entries are returned for all transports in the same protocol family as those used to contact the remote **portmap** daemon.
-m	Displays a table of portmap operations statistics on the specified host. The table contains statistics for each version of portmap (Versions 2, 3, and 4), the number of times each procedure was requested and successfully serviced, the number and type of remote call requests that were made, and information about RPC address lookups that were handled. This information is used for monitoring RPC activities on the host.

Item	Description
-n *Portnum*	Use the *Portnum* parameter as the port number for the **-t** and **-u** options instead of the port number given by the portmap. Using the **-n** options avoids a call to the remote portmap to find out the address of the service. This option is made obsolete by the **-a** option.
-p	Probes the **portmap** service on the host using Version 2 of the portmap protocol and displays a list of all registered RPC programs. If a host is not specified, it defaults to the local host.
-s	Displays a concise list of all registered RPC programs on the host. If host is not specified, the default is the local host.
-t	Makes an RPC call to procedure 0 of prognum on the specified host using TCP, and reports whether a response was received. This option is made obsolete when using the **-T** option as shown in the third syntax.
-T	Specifies the transport where the service is required.
-u	Makes an RPC call to procedure 0 of prognum on the specified host using UDP, and reports whether a response was received. This option is made obsolete when using the **-T** option as shown in the third syntax.

Examples

1. To show all of the RPC services registered on a local machine, enter:

   ```
   rpcinfo -p
   ```

2. To show all of the RPC services registered on a specific machine, enter:

   ```
   rpcinfo -p zelda
   ```

   In this example, the **rpcinfo** command shows all RPC services registered on a machine named `zelda`.

3. To show all machines on the local network that are running a certain version of a specific server, enter:

   ```
   rpcinfo -b ypserv 2
   ```

   In this example, the **rpcinfo** command shows a list of all machines that are running version 2 of the **ypserv** daemon.

4. To delete the registration of a service, enter:

   ```
   rpcinfo -d sprayd 1
   ```

   In this example, the **rpcinfo** command deletes version 1 of the **sprayd** daemon.

5. To check whether the host with IP address 127.0.0.1, program 100003, and version 3 is listening on port 2049 over the TCP, enter:

   ```
   rpcinfo -a 127.0.0.1.8.1 -T tcp 100003 3
   ```

Files

Item	Description
/etc/services	Contains an entry for each service available through the Internet network.

rpvstat Command

Purpose

Displays RPV client device statistics.

Syntax

```
rpvstat -h
rpvstat [-d] [-t] [-i Interval [-c Count] [-d]] [rpvclient_name...]
rpvstat -N [-t] [-i Interval [-c Count] [-d]]
rpvstat -m [-d] [-t] [rpvclient_name...]
rpvstat -R [-r][rpvclient_name...]
rpvstat -r [-R] [rpvclient_name...]
rpvstat -A [-t] [-i Interval [-d] [-c Count] ] [rpvclient_name...] |
rpvstat -C [-t] [-i Interval [-d] [-c Count] ] [rpvclient_name...]
rpvstat -G [-t] [-i Interval [-d] [-c Count] ] [rpvclient_name...]
rpvstat -g [-t] [-i Interval [-d] [-c Count] ] [rpvclient_name...]
```

Description

The **rpvstat** command displays statistical information available from the RPV client device that includes the following details:

- RPV client name
- Connection status
- Total number of completed reads
- Total number of KBs read
- Total number of read errors
- Total number of pending reads
- Total number of pending KBs to read
- Total number of completed writes
- Total number of KBs written
- Total number of write errors
- Total number of pending writes
- Total number of pending KBs to write
- Statistics for asynchronous I/O
- Statistics for asynchronous I/O cache

The read and write errors are displayed together. These counters indicate the number of I/O errors returned to the application.

The **rpvstat** command can optionally display its I/O-related statistics on a per-network basis. A network summary option of the command displays the following additional information:

- Network throughput in kilobytes per second. The throughput is calculated per interval time specified by the user while in monitoring mode.
- The highest recorded values for the pending statistics. These historical high water mark numbers are:
  - Maximum number of pending reads per network

  - Maximum number of pending kilobytes to read per network
  - Maximum number of pending writes per network
  - Maximum number of pending kilobytes to write per network
- Number of retried I/O operations (both read and write operations). This count records the number of I/O retries that have occurred on this network or device. This can be used as an indicator for a marginal or failing network. These statistics are reported on a separate display.

You can also display the statistics for asynchronous mirroring. The **rpvstat** command prints overall asynchronous statistics using the **-A** option. To display statistics per device, you need to specify the list of devices. You can display the asynchronous I/O cache information using **-C** option.

Flags

Table 16. Flags

Flag	Description
-A	Displays the following statistical information for one or more asynchronous I/O operations: • Asynchronous device name • Asynchronous status: The status is printed as a single character. – A - Device is fully configured for asynchronous I/O and can accept asynchronous I/O requests. – I - Asynchronous configuration is incomplete. – U - The device is not configured with asynchronous configuration. Hence it is acting as a synchronous device. All statistics will be printed as zero. – X - Device status cannot be retrieved. All the remaining statistics will be printed as zero. • Total number of asynchronous remote writes completed. The writes are mirrored and complete. • Total asynchronous remote writes completed in kilobyte. The writes are mirrored and complete. • Total number of asynchronous writes pending to mirror. The writes are in the cache. These writes are complete as per LVM is concerned but not yet mirrored. • Total asynchronous writes pending to mirror in kilobyte. The writes are in the cache. These writes are complete as per LVM is concerned but not yet mirrored. • Total number of writes whose response pending. These writes are in the pending queue and not yet written to cache. • Total asynchronous writes response pending in kilobyte. These writes are in the pending queue and not yet written to cache.

Table 16. Flags (continued)	
Flag	**Description**
-C	Displays the following statistical information for asynchronous I/O cache. The VG name is extracted from the ODM. • Volume group name • Asynchronous status: The status is printed as a single character. – A - Device is fully configured for asynchronous I/O and can accept asynchronous I/O requests. – I - Asynchronous configuration is incomplete. – U - The device is not configured with asynchronous configuration. Hence it is acting as a synchronous device. All statistics will be printed as zero. – X - Device status can't be retrieved. All the remaining statistics will be printed as zero • Total asynchronous write operations • Maximum cache utilization in percent • Number of pending asynchronous write requests waiting for the cache flush after cache hits high water mark. • Percentage of writes waiting for the cache flush after cache hits high water mark limit. • Maximum time waited after cache hits high water mark in seconds. • Current free space that is available in cache in kilobytes.
-c *Count*	Displays information at the indicated interval for *Count* times. The value of the *Count* parameter must be an integer greater than zero and less than or equal to 999999. If the *Interval* parameter is specified, but the *Count* parameter is not, then it re-displays indefinitely.
-d	Displays applicable monitored statistics as delta amounts from prior value.
-G	Displays the following statistical information about the asynchronous I/O group: • Volume group name, which is obtained from the Object Data Manager (ODM) • Number of committed volume groups • Average volume group commit time • Total committed asynchronous I/O data in KB • Asynchronous I/O data that is committed per second in KB • Number of completed volume groups • Average volume group completion time • Total completed asynchronous I/O data in KB • Asynchronous I/O data that is completed per second in KB • Number of volume groups read from cache disk • Total asynchronous I/O data that is read from cache disk in KB • Average volume group read time • Asynchronous I/O data that is read per second in KB • Number of times cache disk is detected as full • Total waiting time for cache memory to be available • Total asynchronous I/O write data in transit in KB

Table 16. Flags (continued)	
Flag	**Description**
-g	Displays a summary of statistical information about the asynchronous I/O group that includes the following data: • Volume group name, which is obtained from the ODM • Average volume group formation time • Average volume group commit time • Average volume group completion time • Average volume group read time • Number of times cache disk is detected as full
-h	Displays command syntax and usage.
-i *Interval*	Automatically displays status in every *Interval* seconds. The value of the *Interval* parameter must be an integer greater than zero and less than or equal to 3600. If the *Interval* parameter is not specified, then the status information is displayed once.
-m	Displays historical maximum pending values (high water mark values) and accumulated retry count.
-N	Displays summary statistics by mirroring network, including throughput rate for each network.
-n	Displays statistics for individual mirroring networks.
-R	Resets counters in the RPV clients (requires root privilege).
-r	Resets counters for the asynchronous I/O cache information. You can specify the **-R** and **-r** options together to reset all counters. Requires root access.
-t	Includes date and time in display.

Table 17. Operands	
Field	**Value**
rpvclient_name	Name of one or more RPV clients for which to display information. If no RPV client names are specified, then information for all RPV clients is displayed.

Note:

- In monitor mode (**-i**) if the **-d** option is also specified, then some statistics (completed reads, completed writes, completed kilobyte read, completed kilobyte written, and errors) are represented as delta amounts from their previously displayed values. These statistics are prefixed with a plus sign (+) on the second and succeeding displays. A delta value is not displayed under certain circumstances, such as when an error is detected in the previous iteration, or a configuration change is made between iterations.
- When a list of RPV client devices is not explicitly listed on the command line, the list of all available RPV Clients is generated at command initiation. In monitor mode, this list of RPV clients to display is not refreshed on each display loop. This means any additional RPV clients added or deleted are not recognized until the command is started again.
- The **-i** interval is the time, in seconds, between each successive gathering and display of RPV statistics in monitor mode. This interval is not a precise measure of the elapsed time between each successive updated display. The **rpvstat** command obtains some of the information it displays by calling system services and has no control over the amount of time these services take to complete their processing. Larger numbers of RPVs will result in the **rpvstat** command taking longer to gather information and

will elongate the time between successive displays in monitor mode, sometimes taking much longer than the **-i** interval between displays.

- The count of reads and writes is accumulated on a per buffer basis. This means that if an application I/O passes a vector of buffers in a single read or write call, then instead of counting that read or write as a single I/O, it is counted as the number of buffers in the vector.
- The count of completed and pending I/O kilobytes is truncated. Any fractional amount of a KB is dropped in the output display.
- The cx field in the output displays one of the following connection status:

Table 18. cx output

Field	Description
A number	This number is the count of active network connections between the RPV Client and its RPV Server.
Y	Indicates the connection represented by the IP address is available and functioning.
N	Indicates the connection represented by the IP address is not available.
X	Indicates the required information could not be retrieved from the device driver. Reasons for this status can be that the device driver is not loaded, the device is not in the available state, or the device has been deleted.

Exit Status

This command returns the following exit values:

Table 19. Exist status

Field	Description
0	No errors.
>0	An error occurred.

Examples

1. To display statistical information for all RPV clients, enter the following command:

   ```
   rpvstat
   ```

2. To display statistical information for RPV client hdisk14, enter the following command:

   ```
   rpvstat hdisk14
   ```

3. To reset the statistical counters in RPV client hdisk23, enter the following command:

   ```
   rpvstat -R hdisk23
   ```

4. To display statistical information for RPV client hdisk14 and repeat the display every 30 seconds for 12 times, enter the following command:

   ```
   rpvstat hdisk14 -i 30 -c 12
   ```

5. To display statistical information for all RPV clients and include detailed information by mirroring network, enter the following command:

   ```
   rpvstat -n
   ```

6. To display statistical information for all mirroring networks, enter the following command:

   ```
   rpvstat -N
   ```

7. To display statistical information on maximum pending values for all RPV clients, enter the following command:

```
rpvstat -m
```

8. To display statistical information about asynchronous I/O groups, enter the following command:

```
rpvstat -G
```

9. To display summary of statistical information about asynchronous I/O groups, enter the following command:

```
rpvstat -g
```

Files

The `/usr/sbin/rpvstat` path contains the **rpvstat** command.

rpvutil Command

Purpose

Configures a mirror pool in the remote physical volume (RPV) client of Geographic Logical Volume Manager (GLVM).

Syntax

```
rpvutil -h [tunable_name]
rpvutil [-v vg_name] {-a | -o tunable_name[=value]}
rpvutil -d tunable_name
```

Description

Starting with AIX version 7.2.5, you can use the **rpvutil** command to perform the following operations:

- Set the maximum expected delay before the application receives the I/O acknowledgments for a mirror pool in a volume group.
- Compress the I/O data packet before it is sent from the RPV client to the RPV server or for monitoring the RPV network.
- Monitor the RPV data network.

Flags

-a
Displays the current values of all tunable parameters of the GLVM RPV client.

-d *tunable_name*
Resets the specified tunable parameter to its default value.

-h [*tunable_name*]
Displays help information for the **rpvutil** command. Optionally, if you specify a tunable parameter with this flag, the command displays help information for the specified tunable parameter.

-o *tunable_name[=value]*
If you do not specify any value for the tunable parameter, the **-o** *tunable_name* flag displays the current values of the specified tunable parameter. You can specify the following tunable parameters for the **rpvutil -o** command:

compression=1|0
To use the compression tunable parameter, consider the following prerequisites:

- Ensure that the RPV client and the RPV server are running AIX version 7.2.5, or later, with all the latest RPV device drivers.
- Ensure that both the RPV server and the RPV client are IBM Power Systems servers with NX842 acceleration units.
- Ensure that the compression tunable parameter is enabled on both the RPV server and RPV client so that the I/O data packets are compressed when the workload is failed over between the RPV client and the RPV server.

When you set the **compression** tunable parameter to 1, the **rpvutil** command compresses the I/O data packet before it is sent from the RPV client to the RPV server by using the special acceleration units for cryptography and compression (NX842) that are available in IBM Power Systems servers. If the I/O data packet is compressed successfully, a flag is set in the data packet that is sent to indicate that the data packet is compressed. If the I/O data packet that is received at the RPV server has the compression flag enabled, the RPV server decompresses the I/O data packet. If the NX842 acceleration unit is not available in the RPV server, the RPV server attempts a software decompression operation instead of the hardware decompression operation. By default, this option is set to 0 (disabled).

io_grp_latency=*timeout_value*
Indicates the maximum expected delay, in milliseconds, before receiving the I/O acknowledgment for a mirror pool that is configured in asynchronous mode. You must specify the volume group that is associated to the mirror pool by using the **-v** flag. The default delay value is 10 ms. You can specify lower values to improve I/O performance but CPU consumption might increase.

>|

nw_sessions=*number_of_sessions*
Specifies the number of parallel RPV sessions (sender and receiver threads) per network. This tunable parameter is used to increase the number of parallel RPV sessions per GLVM network, which helps in sending the data in parallel and to improve the data transfer rate and use full bandwidth between the sites. A single RPV session on a RPV client consists of a sender thread for transfer of the data, and a receiver thread to receive the acknowledgment. Range of the **nw_sessions** is 1 - 99. You can change RPV value only when all the RPV devices on the AIX operating system are in defined state.

Note: When you change the number of sessions value to a higher value, you might create many threads. You must choose number of sessions value based on the workload.

|<

-v *vg_name*
Specifies a volume group name for which the tunable parameters must be added, modified, or displayed.

Examples

- To set a timeout value of 5 ms for the gmvg1 volume group, enter the following command:

```
rpvutil -v gmvg1 -o io_grp_latency=5
```

An output that is similar to the following example is displayed:

```
RPVC timeout: 10. ODM Value: 10
Updated timeout[5] to ODM VG: gmvg1
Updated timeout[5] to rpvc. VG: gmvg1
```

- To check the value of the compression option, enter the following command:

```
rpvutil -o compression
```

An output that is similar to the following example is displayed:

```
compression[odm,driver]=ENABLED,ENABLED
```

- To compress the I/O data packets before sending the I/O data packets from the RPV client to the RPV server, enter the following command:

  ```
  rpvutil -o compression=1
  ```

  The command returns 0 after successful completion.
- To set the network sessions, enter the following command:

  ```
  rpvutil -o nw_sessions=2
  ```

  An output that is similar to the following example is displayed:

```
Setting nw_sessions to 2
```

- To display the current value of tunable, enter the following command:

  ```
  rpvutil -o nw_sessions
  ```

  An output that is similar to the following example is displayed:

```
nw_sessions = 1
```

Files

The `/usr/lib/methods` directory contains the **rpvutil** command.

rrestore Command

Purpose

Copies previously backed up file systems from a remote machine's device to the local machine.

Syntax

rrestore [**-b***Number*] [**-h**] [**-i**] [**-m**] [**-s***Number*] [**-t**] [**-v**] [**-y**] [**-x**] [**-r**] [**-R**] **-f***Machine***:***Device* [*FileSystem ...*] [*File ...*]

Description

The **rrestore** command restores Version 3 by i-node backups from a remote machine's device to a file system on the local machine. The **rrestore** command creates a server on the remote machine to the backup medium.

The **rrestore** command only accepts backup formats created when a file system is backed up by i-node.

Note: A user must have root authority to execute this command.

Flags

Item	Description
-b*Number*	Specifies the number of blocks to read in a single input operation. If you do not specify this flag, the **rrestore** command selects a default value appropriate for the physical device you have selected. Larger values of the *Number* variable result in larger physical transfers from tape devices.

Item	Description
-f*Machine***:***Device*	Specifies the input device on the remote machine. Specify the *Device* variable as a file name (such as the **/dev/rmt0** file) to get input from the named device. For more information on using tape devices see the **rmt**special file.
-h	Restores only the actual directory named by the *File* parameter, not the files contained in that directory. This option is ignored when either the **-r** or **-R** flag is specified.
-i	Starts the interactive mode. This flag allows you to restore selected files from the directory represented by the *File* parameter. The subcommands for the **-i** flag are: **ls [*Directory*]** Displays directory names within the specified *Directory* parameter with a / (slash) after the name, and displays files to be restored with an * (asterisk) before the name. If the **-v** flag is used, the i-node number of each file and directory is also displayed. If the *Directory1* parameter is not specified, the current directory is used. **cd *Directory*** Changes the current directory to the *Directory* parameter. **pwd** Displays the full path name of the current directory. **add [*File*]** Specifies the *File* parameter to restore. If the *File* parameter is a directory, that directory and all its files are restored (unless the **-h** flag is used). Files to be restored are displayed with an * (asterisk) before the name by the **ls** subcommand. If the *File* parameter is not specified, the current directory is used. **delete [*File*]** Specifies the *File* parameter to ignore in restore. If the *File* parameter is a directory, the directory and all its files are not restored (unless the **-h** flag is used). If the *File* parameter is not specified, the current directory is used. **extract** Restores all files displayed with an * (asterisk) before the name by the **ls** subcommand. **setmodes** Sets owner, modes, and times for the files being restored rather than using this information as it resides on the backup medium. **verbose** Displays the i-node numbers of all restored files with the **ls** subcommand. Information about each file is also displayed as it is restored. The next invocation of the **verbose** subcommand turns **verbose** off. **help** Displays a summary of the subcommands. **quit** Stops execution of the **rrestore** command immediately, even if all files requested have not been restored.
-m	Restores files by i-node number rather than by path name.

Item	Description
-r	Restores an entire file system. **Attention:** If you do not follow this procedure carefully, you can ruin an entire file system. If you are restoring a full (level 0) backup, run the **mkfs**command to create an empty file system before doing the restore. To restore an incremental backup at level 2, for example, run the **mkfs** command, restore the appropriate level 0 backup, restore the level 1 backup, and finally restore the level 2 backup. As an added safety precaution, run the **fsck** command after you restore each backup level.
-R	Causes the **rrestore** command to request a specific volume in a multivolume set of backup medium when restoring an entire file system. The **-R** flag provides the ability to interrupt and resume the **rrestore** command.
-s*Number*	Specifies which backup to restore from a multibackup medium. Numbering starts with 1.
-t	Displays the table of contents for the backed up files. The **rrestore** command displays the file name. The names are relative to the root (**/**) directory of the file system backed up. The only exception is the root (**/**) directory itself.
-v	Reports the progress of the restoration as it proceeds.
-x	Restores individually named files. If no names are given, all files on that medium are restored. The names must be in the same form as the names shown by the **-t** flag.
-y	Prevents the **rrestore** command from asking whether it should stop the restore if a tape error is encountered. The **rrestore** command attempts to skip over bad blocks.
-?	Displays the usage message.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list files from a remote tape device, enter:

   ```
   rrestore  -fmachine1:/dev/rmt0  -t
   ```

   This command reads information from the `/dev/rmt0` device on remote `machine1`. The file names are shown.

2. To restore files, enter:

   ```
   rrestore  -x  -fmachine1:/dev/rmt0 /home/mike/file1
   ```

   This command extracts the `/home/mike/file1` file from the backup medium on the `/dev/rmt0` device on remote `machine1`.
3. To restore all the files in a directory, enter:

   ```
   rrestore -fhost:/dev/rmt0 -x /home/mike
   ```

   This command restores the directory `/home/mike` and all the files it contains.
4. To restore a directory, but not the files in the directory, enter:

   ```
   rrestore -fhost:/dev/rmt0 -x -h /home/mike
   ```
5. To restore all the files in a directory from a specific backup on a multibackup medium, enter:

   ```
   rrestore -s3 -fhost:/dev/rmt0.1 -x /home/mike
   ```

   This command restores the `/home/mike` directory and all the files it contains from the third backup on the backup medium.

Files

Item	Description
/dev/rfd0	Specifies the default restore device.
/usr/sbin/rrestore	Contains the **rrestore** command.

Rsh command

Purpose

Starts the restricted version of the Bourne shell.

Syntax

Rsh [**-i**] [{ **+** | **-** }{ [**-a**] [**-e**] [**-f**] [**-h**] [**-k**] [**-n**] [**-t** *timeout*] [**-u**] [**-v**] [**-x**] }] [**-c** *String* | **-s** | *File* [**Parameter**]]

Note: Preceding a flag with a **+** (plus sign) rather than a **-** (minus sign) turns it off.

Description

The **Rsh** command starts a restricted version of the Bourne shell, which is useful for installations that require a more controlled shell environment. You can create user environments with a limited set of privileges and capabilities.

Flags

The Bourne shell interprets the following flags only when the shell is started at the command line.

Note: Unless you specify either the **-c** or **-s** flag, the shell assumes that the next parameter is a command file (shell script). It passes anything else on the command line to that command file.

Item	Description
-a	Marks for export all variables to which an assignment is performed. If the assignment precedes a command name, the export attribute is effective only for that command's execution environment, except when the assignment precedes one of the special built-in commands. In this case, the export attribute persists after the built-in command is completed. If the assignment does not precede a command name, or if the assignment is a result of the operation of the **getopts** or **read** command, the export attribute persists until the variable is unset.
-c *String*	Runs commands that are read from the *String* variable. Sets the value of special parameter 0 from the value of the *String* variable and the positional parameters ($1, $2, and so on) in sequence from the remaining parameter operands. The shell does not read additional commands from standard input when you specify this flag.
-e	Exits immediately if all of the following conditions exist for a command: • It exits with a return value greater than 0. • It is not part of the compound list of a while, until, or if command. • It is not being tested by using AND or OR lists. • It is not a pipeline that is preceded by the ! (exclamation point) reserved word.
-f	Disables file name substitution.
-h	Locates and remembers the commands that are called within functions as the functions are defined. (Usually these commands are located when the function is run; see the hash command.)
-i	Makes the shell interactive, even if input and output are not from a workstation. In this case, the shell ignores the TERMINATE signal, so that the **kill 0** command does not stop an interactive shell, and traps an INTERRUPT signal, so you can interrupt the function of the wait command. In all cases, the shell ignores the QUIT signal.
-k	Places all keyword parameters in the environment for a command, not just those preceding the command name.
-n	Reads commands but does not run them. The **-n** flag can be used to check for shell-script syntax errors. An interactive shell might ignore this option.
-s	Reads commands from standard input. Any remaining parameters that are specified are passed as positional parameters to the new shell. Shell output is written to standard error, except for the output of built-in commands.
-t *timeout*	Exits after the timeout seconds if there is no reply from the server.
-u	Treats an unset variable as an error and immediately exits when it performs variable substitution. An interactive shell does not exit.
-v	Displays shell input lines as they are read.
-x	Displays commands and their arguments before they are run.

Note: Using a **+** (plus sign) rather than a **-** (minus sign) unsets flags. The *$-* special variable contains the current set of flags.

Files

Item	Description
/usr/bin/bsh	Specifies the path name to the Bourne shell.
/usr/bin/Rsh	Specifies the path name to the restricted Bourne shell, a subset of the Bourne shell.

Item	Description
/tmp/sh*	Contains temporary files that are created when a shell is opened.

rsh or remsh Command

Purpose

Executes the specified command at the remote host or logs in to the remote host.

Syntax

{ **rsh** | **remsh** } *RemoteHost* [**-n] [-l *User*] [-f | -F] [-k *realm*] [-S] [-u] [*Command*]**

Description

The **/usr/bin/rsh** command executes the command specified by the *Command* parameter at the remote host specified by the *RemoteHost* parameter; if the *Command* parameter is not specified, the **rsh** command logs into the remote host specified by the *RemoteHost* parameter. The **rsh** command sends standard input from the local command line to the remote command and receives standard output and standard error from the remote command.

Note: Because any input to the remote command must be specified on the local command line, you cannot use the **rsh** command to execute an interactive command on a remote host. If you need to execute an interactive command on a remote host, use either the **rlogin** command or the **rsh** command without specifying the *Command* parameter. If you do not specify the *Command* parameter, the **rsh** command executes the **rlogin** command instead.

Access Files

If you do not specify the **-l** flag, the local user name is used at the remote host. If **-l** *User* is entered, the specified user name is used at the remote host.

Using Standard Authentication

The remote host allows access only if at least one of the following conditions is satisfied:

- The local user ID is not the root user, and the name of the local host is listed as an equivalent host in the remote **/etc/hosts.equiv** file.
- If either the local user ID is the root user or the check of **/etc/hosts.equiv** is unsuccessful, the remote user's home directory must contain a **$HOME/.rhosts** file that lists the local host and user name.

Although you can set any permissions for the **$HOME/.rhosts** file, it is recommended that the permissions of the **.rhosts** file be set to 600 (read and write by owner only).

In addition to the preceding conditions, the **rsh** command also allows access to the remote host if the remote user account does not have a password defined. However, for security reasons, use of a password on all user accounts is recommended.

For Kerberos 5 Authentication

The remote host allows access only if all of the following conditions are satisfied:

- The local user has current DCE credentials.
- The local and remote systems are configured for Kerberos 5 authentication (On some remote systems, this method is not necessary. It is necessary that a daemon is listening to the klogin port).
- The remote system accepts the DCE credentials as sufficient for access to the remote account. See the **kvalid_user** function for more information.

Remote Command Execution

When the remote command is run, pressing the Interrupt, Terminate, or Quit key sequences sends the corresponding signal to the remote process. However, pressing the Stop key sequence stops only the local process. Usually, when the remote command terminates, the local **rsh** process terminates.

To have shell metacharacters interpreted on the remote host, place the metacharacters inside " " (double quotation marks). Otherwise, the metacharacters are interpreted by the local shell.

When using the **rsh** command, you can create a link to a path (to which you have permission to write), by using a host name that is specified by the *HostName* parameter as the link name. For example:

```
ln -s /usr/bin/rsh HostName
```

After the link is established, you can specify the *HostName* parameter and a command that is specified by the *Command* parameter from the command line. The **rsh** command remotely runs the command on the remote host. The syntax is:

```
HostName Command
```

For example, if you are linked to remote host `opus` and want to run the **date** command, enter:

```
opus date
```

Because you can not specify the **-l** *User* flag, the remote command is successful only if the local user has a user account on the remote host. Otherwise, the **rsh** command returns a `Login incorrect` error message. When you specify the *HostName* parameter without a command, the **rsh** command calls the **rlogin** command, which logs you into the remote host. Again, for successful login, the local user must have a user account on the remote host.

Flags

-a
Indicates that the standard error of the remote command is the same as standard output. No provision is made for sending arbitrary signals to the remote process.

-f
Causes the credentials to be forwarded. This flag is ignored if Kerberos 5 is not the current authentication method. Authentication fails if the current DCE credentials are not marked forwardable.

-F
Causes the credentials to be forwarded. In addition the credentials on the remote system is marked forwardable (allowing them to be passed to another remote system). This flag is ignored if Kerberos 5 is not the current authentication method. Authentication fails if the current DCE credentials are not marked forwardable.

-k *realm*
Allows the user to specify the realm of the remote station if it is different from the local systems realm. For these purposes, a realm is synonymous with a DCE cell. This flag is ignored if Kerberos 5 is not the current authentication method.

-l *User*
Specifies that the **rsh** command must log in to the remote host as the user specified by the *User* variable instead of the local user name. If this flag is not specified, the local and remote user names are the same.

-n
Specifies that the **rsh** command must not read from standard input.

-S
Secure option, force remote IP address of the standard error connection to be the same as the standard output connection.

-u
Use standard AIX authentication only.

Exit Status

This command returns the following exit values:

0
Successful completion.

>0
An error occurred.

Security

The remote host allows access only if at least one of the following conditions is satisfied:

- The local user ID is listed as a principal in the authentication database and had performed a **kinit** to obtain an authentication ticket.
- If a **$HOME/.klogin** file exists, it must be in the local user's **$HOME** directory on the target system. The local user and any user must be listed or the services that are allowed to the **rsh** command is considered. This file performs a similar function to a local **.rhosts** file. Each line in this file must contain a principal in the form of *principal.instance@realm*. If the originating user is authenticated as one of the principals that are named in the **.klogin** file, access is granted to the account. The owner of the account is granted access if the **.klogin** file is not present.

For security reasons, any **$HOME/.klogin** file must be owned by the remote user and only the AIX owner ID has read and write access (permissions = 600) to the **.klogin** file.

Attention RBAC users and Trusted AIX users: This command can run privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations that are associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

In the following examples, the local host, `host1`, is listed in the **/etc/hosts.equiv** file at the remote host, `host2`.

1. To check the amount of free disk space on a remote host, enter:

   ```
   rsh host2 df
   ```

   The amount of free disk space on `host2` is displayed on the local system.
2. To append a remote file to another file on the remote host, place the >> metacharacters in quotation marks, and enter:

   ```
   rsh host2 cat test1 ">>" test2
   ```

   The file `test1` is appended to `test2` on remote host `host2`.
3. To append a remote file at the remote host to a local file, omit the quotation marks, and enter:

   ```
   rsh host2 cat test2 >> test3
   ```

   The remote file `test2` on `host2` is appended to the local file `test3`.
4. To append a remote file to a local file and use a remote user's permissions at the remote host, enter:

   ```
   rsh host2  -l jane cat test4 >> test5
   ```

   The remote file `test4` is appended to the local file `test5` at the remote host, with user `jane`'s permissions.
5. This example shows how the root user can issue an **rcp** on a remote host when the authentication is Kerberos 4 on both the target and server. The root user must be in the authentication database and

must have already issued **kinit** on the local host. The command is issued at the local host to copy the file, stuff, from node r05n07 to node r05n05 on an SP.

```
/usr/lpp/ssp/rcmd/bin/rsh r05n07 'export KRBTKTFILE=/tmp/rcmdtkt$$; \
/usr/lpp/ssp/rcmd/bin/rcmdtgt; \
/usr/lpp/ssp/rcmd/bin/rcp /tmp/stuff r05n05:/tmp/stuff;'
```

The root user sets the KRBTKTFILE environment variable to the name of a temporary ticket-cache file and then obtains a service ticket by issuing the **rcmdtgt** command. The **rcp** uses the service ticket to authenticate from host r05n07 to host r05n05.

Files

Item	Description
$HOME/.klogin	Specifies remote users that can use a local user account.
/usr/lpp/ssp/rcmd/bin/rsh	Link to AIX Secure **/usr/bin/rsh** that calls the SP Kerberos 4 **rsh** routine if applicable.
/usr/lpp/ssp/rcmd/bin/remsh	Link to AIX Secure **/usr/bin/rsh** that calls the SP Kerberos 4 **rsh** routine if applicable.

Prerequisite Information

Refer to the chapter on security in IBM Parallel System Support Programs for *AIX: Administration Guide* for an overview. You can access this publication at the following Web site: *http://www.rs6000.ibm.com/resource/aix_resource*

Refer to the "RS/6000 SP Files and Other Technical Information" section of IBM Parallel System Support Programs for AIX: Command and Technical Reference for additional Kerberos information. You can access this publication at the following Web site: *http://www.rs6000.ibm.com/resource/aix_resource*

rshd Daemon

Purpose

Provides the server function for remote command execution.

Syntax

Note: The **rshd** daemon is usually started by the **inetd** daemon. It can also be controlled from the command line, using SRC commands.

/usr/sbin/rshd [**-c**] [**-s**] [**p**]

Description

The **/usr/sbin/rshd** daemon is the server for the **rcp** and **rsh** commands. The **rshd** daemon provides remote execution of shell commands. These commands are based on requests from privileged sockets on trusted hosts. The shell commands must have user authentication. The **rshd** daemon listens at the socket defined in the **/etc/services** file.

Changes to the **rshd** daemon can be made using the System Management Interface Tool (SMIT) or System Resource Controller (SRC), by editing the **/etc/inetd.conf** or **/etc/services** file. Entering **rshd** at the command line is not recommended. The **rshd** daemon is started by default when it is uncommented in the **/etc/inetd.conf** file.

The **inetd** daemon get its information from the **/etc/inetd.conf** file and the **/etc/services** file.

After changing the **/etc/inetd.conf** or **/etc/services** file, run the `refresh -s inetd` or `kill -1 InetdPID` command to inform the **inetd** daemon of the changes to its configuration file.

Service Request Protocol

When the **rshd** daemon receives a service request, it initiates the following protocol:

1. The **rshd** daemon checks the source port number for the request. If the port number is not in the range 512 through 1023, the **rshd** daemon terminates the connection.
2. The **rshd** daemon reads characters from the socket up to a null byte. The string read is interpreted as an ASCII number (base 10). If this number is nonzero, the **rshd** daemon interprets it as the port number of a secondary stream to be used as standard error. A second connection is created to the specified port on the client host. The source port on the local host is also in the range 512 through 1023.
3. The **rshd** daemon uses the source address of the initial connection request to determine the name of the client host. If the name cannot be determined, the **rshd** daemon uses the dotted decimal representation of the client host's address.
4. The **rshd** daemon retrieves the following information from the initial socket:
   - A null-terminated string of at most 16 bytes interpreted as the user name of the user on the client host.
   - A null-terminated string of at most 16 bytes interpreted as the user name to be used on the local server host.
   - Another null-terminated string interpreted as a command line to be passed to a shell on the local server host.
5. The **rshd** daemon attempts to validate the user using the following steps:
   a. The **rshd** daemon looks up the local user name in the **chdir** subroutine). If either the lookup or the directory change fails, the **rshd** daemon terminates the connection.
   b. If the local user ID is a nonzero value, the **rshd** daemon searches the **/etc/hosts.equiv** file to see if the name of the client workstation is listed. If the client workstation is listed as an equivalent host, the **rshd** daemon validates the user.
   c. If the **$HOME/.rhosts** file exists, the **rshd** daemon tries to authenticate the user by checking the **.rhosts** file.
   d. If either the **$HOME/.rhosts** authentication fails or the client host is not an equivalent host, the **rshd** daemon terminates the connection.
6. After the **rshd** daemon validates the user, the **rshd** daemon returns a null byte on the initial connection and passes the command line to the user's local login shell. The shell then inherits the network connections established by the **rshd** daemon.

The **rshd** daemon should be controlled using the System Management Interface Tool (SMIT) or by changing the **/etc/inetd.conf** file. Typing `rshd` at the command line is not recommended.

Manipulating the rshd Daemon with the System Resource Controller

The rshd daemon is a subserver of the inetd daemon, which is a subsystem of the System Resource Controller (SRC). The rshd daemon is a member of the tcpip SRC subsystem group. This daemon is enabled by default in the /etc/inetd.conf file and can be manipulated by the following SRC commands:

Item	Description
startsrc	Starts a subsystem, group of subsystems, or a subserver.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
lssrc	Gets the status or a subsystem, group or subsystems, or a subserver.

Flags

Item	Description
c	Suppresses the sanity check of a host name lookup.

Item	Description
p	Runs your *.profile* file whenever you issues the **rsh** command in the non-interactive mode. Without this flag, your *.profile* file is not run in case of the rsh command in the non-interactive mode.
s	Turns on socket-level debugging.

Security

The **rshd** daemon is a PAM-enabled application with a service name of *rsh*. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the **usw** stanza of the **/etc/security/login.cfg** file, to the **PAM_AUTH** attribute as the root user.

The authentication mechanisms used when PAM is enabled depend on the configuration for the rsh service in the /**etc/pam.conf** file. The **rshd** daemon requires the **/etc/pam.conf** entries for the **auth**, **account**, and **session** module types. Listed below is a recommended configuration in the **/etc/pam.conf** file for the*rsh* service:

```
#
# AIX rsh configuration
#
rsh auth      sufficient   /usr/lib/security/pam_rhosts_auth

rsh account   required     /usr/lib/security/pam_aix

rsh session   required     /usr/lib/security/pam_aix
```

Examples

Note: The arguments for the **rshd** daemon can be specified by using SMIT or by editing the **/etc/inetd.conf** file.

1. To start the **rshd**daemon, type the following:

   ```
   startsrc -t shell
   ```

   This command starts the **rshd** subserver.
2. To stop the **rshd**daemon, type the following:

   ```
   stopsrc -t shell
   ```

   This command allows all pending connections to start and existing connections to complete but prevents new connections from starting.
3. To force stop the **rshd** daemon and all **rshd** connections, type the following: :

   ```
   stopsrc -t -f shell
   ```

   This command terminates all pending connections and existing connections immediately.
4. To display a short status report about the **rshd** daemon, type the following: :

   ```
   lssrc -t shell
   ```

   This command returns the daemon's name, process ID, and state (active or inactive).

rstatd Daemon

Purpose

Returns performance statistics obtained from the kernel.

Syntax

/usr/sbin/rpc.rstatd

Description

The **rstatd** daemon is a server that returns performance statistics obtained from the kernel. The **rstatd** daemon is normally started by the **inetd** daemon.

Files

Item	Description
/etc/inetd.conf	TCP/IP configuration file that starts RPC daemons and other TCP/IP daemons.
/etc/services	Contains an entry for each server available through Internet.

rsyslogd Daemon

Purpose

Logs system messages.

Description

The **rsyslogd** daemon reads a socket and sends the message line to a destination that is specified by the `/etc/rsyslog.conf` configuration file. The **rsyslogd** daemon reads the configuration file when it is activated. You can start the **rsyslogd** daemon from the source master by using the following commands:

```
startsrc -s syslogd
stopsrc -s syslogd
```

The `startsrc` option starts the **rsyslogd** daemon. To start multiple **rsyslogd** daemons, run the `startsrc` option repeatedly with a new `pid` file by using the `-i` command-line option. The **startsrc** command specifies the arguments for the **rsyslogd** daemon by using the `startsrc -a` flag. The arguments must be protected from interpretation by the shell with double quotation marks.

The `stopsrc` option stops all instances of the **rsyslogd** daemon. To stop a specific instance, you must specify the `-p <pid>` option.

```
stopsrc -p <pid of syslogd daemon>
```

Default logging application:

After the **rsyslogd** daemon is installed, it cannot be started immediately and **syslogd** daemon continues to be used to log system messages. To configure the **rsyslogd** daemon to log messages by default, run the `syslog_ssw` script by using the `-r` option.

After the **rsyslogd** daemon is configured to log system messages, the **rsyslogd** daemon starts with a default command-line argument of `-c5`. This option ensures that the **rsyslogd** daemon starts in a normal mode and is not compatible with an earlier version.

Default `rsyslog.conf` file:

To configure and use the **rsyslogd** daemon, see the reference section of the documentation.

After installation, the default `/etc/rsyslog.conf` configuration file has the following information:

```
#############################################################################
# Rsyslog is free software: it is distributed under the          #
# terms of the GNU General Public License as published by        #
# the Free Software Foundation, under version 3 of the License.  #
```

```
#                                                                    #
# if you experience problems, check                                  #
# http://www.rsyslog.com/doc/troubleshoot.html for assistance        #
#                                                                    #
# Load the UNIX socket for local communication                       #
  $ModLoad imuxsock                                                  #
#                                                                    #
# Load the UDP module for remote communication                       #
  $ModLoad imudp                                                     #
#                                                                    #
# Run the UDP server on the default port 514                         #
  $UDPServerRun 514                                                  #
#                                                                    #
######################################################################
```

Almost all parameters in the `syslog.conf` file functions with the **rsyslogd** daemon except for the AIX specific parameters such as pureScale API support. To convert a `syslog.conf` file into a supported `rsyslog.conf` file, the switching script must be used with the `-c` option.

Switching script usage

```
syslog_ssw [ -r | -s | -c SourceSyslogConffile DestRsyslogConffile ]
```

Item	Descriptor
`-r`	Switch to `rsyslog` daemon as the default logging application.
`-s`	Switch to `syslog` daemon as the default logging application.
`-c`	Convert configuration rules in the `syslog.conf` file to the rules in the `rsyslog.conf` file. However, the AIX specific parameters that are not understood by the `rsyslogd` daemon are removed during conversion.

When you switch the default logging application by using the `-r` or the `-s` option, this choice remains persistent across restart.

The `startsrc -s syslogd` command starts the **rsyslogd** or the **syslogd** daemon that is based on the default logging application that is set.

The `syslog_ssw` script is not present by default, and is available after the **rsyslogd** daemon is installed.

Examples

1. To stop the existing **syslogd** daemon and to start the **rsyslogd** daemon, run the following command:

   ```
   syslog_ssw -r
   ```

2. To stop the existing **rsyslogd** daemon and to start the **syslogd** daemon, run the following command:

   ```
   syslog_ssw -s
   ```

3. To convert the `syslog.conf` file to `rsyslog.conf` file, and to create an `rsyslog.conf` file if the file does not exist, run the following command:

   ```
   syslog_ssw -c syslog.conf rsyslog.conf
   ```

   This conversion removes the AIX specific parameters and allows the newly created file to be used with the **rsyslogd** daemon.

4. To start the default logging application, run the following command:

```
startsrc -s syslogd
```

The default logging application can be the **syslogd** daemon or the **rsyslogd** daemon.

Files

Item	Descriptor
`/etc/rsyslog.conf`	Controls the output of the **rsyslogd** daemon.
`/etc/rsyslogd.pid`	Contains the process ID.

rtcd Daemon

Purpose

Monitors the file modification events, checks for the resulting compliance violations, and alerts the administrators.

Description

The **rtcd** daemon reads the configuration information that is defined in the `/etc/security/rtc/rtcd.conf` file. The **rtcd** daemon runs the **aixpert** command to check for compliance violation during startup. It alerts the recipients who are specified in the `/etc/security/rtc/rtcd.conf` file by email if any violation is determined.

The **rtcd** daemon continuously monitors the files that are specified in the`/etc/security/rtc/rtcd_policy.conf` file for file changes. If any files change, the **rtcd** runs the **aixpert** command to check for the compliance violations and sends an alert email for any violations.

The **rtcd** daemon is placed under the SRC control after successful configuration of Real-Time Compliance. You must manage the **rtcd** daemon by using the System Resource Controller (SRC) commands.

Security

The **rtcd** daemon is owned by the root user and the system group. Only the root user and users with `aix.system.config.src` authorization are authorized to manage the command.

Examples

1. To start the **rtcd** daemon, enter the following command:

```
# startsrc -s rtcd
```

2. To check the **rtcd** daemon, enter the following command:

```
# lssrc -s rtcd
```

3. To stop the **rtcd** daemon, enter the following command:

```
# stopsrc -s rtcd
```

Files

Item	Description
/etc/security/rtc/rtcd_policy.conf	Contains the configuration information for the **rtcd** daemon.
/etc/security/rtc/rtcd.conf	Grants read (r) and write (w) access to the root user.

rtl_enable Command

Purpose

Relinks shared objects to enable the runtime linker to use them.

Syntax

rtl_enable [**-R** | **-o** *Name*] [**-l**] [**-s**] *File* [*ldFlag* ...] [**-F** *ObjsLibs* ...]

Description

The **rtl_enable** command relinks a module, or an archive containing modules, with the **-G** flag, to enable runtime linking. A module is an XCOFF file containing a loader section. A shared object is a module with the F_SHROBJ flag set in the XCOFF header.

In its simplest form, the **rtl_enable** command creates a new file with the name *File***.new**. If *File* is a module, *File***.new** will be the same kind of module. If *File* is an archive, *File***.new** will be an archive whose members have the same names as the members of *File*. The **rtl_enable** command relinks the modules in the new archive to enable run-time linking. The **rtl_enable** command archives other members unchanged into the output file.

The **rtl_enable** command uses the loader section in *File* (or its members) to create import and export files, to determine the **libpath** information, and to determine the entry point.

Flags

Item	Description
-F *ObjsLibs* ...	Adds *ObjsLibs* to the beginning of the generated **ld** command. The *ObjsLibs* parameter is either an object file or a library (specified with the **ld** command's **-l** (lowercase L) flag). If you are enabling an archive, adds the *ObjsLibs* to the **ld** command for all shared objects in the archive.
-l	(Lowercase L) Leaves the import and export files in the current directory instead of deleting them. Import files have the suffix **.imp** and export files, the suffix **.exp**. The **rtl_enable** command adds the suffixes to the input file name if *File* is a module. It adds the suffixes to the names of members that are modules if *File* is an archive.
-o *Name*	Specifies an alternate output file name instead of *File***.new**. Do not use this flag with the **-R** flag.
-R	Replaces the input file instead of creating a new file. It will not overwrite the input file if any errors occur. Do not use this flag with the **-o** flag.

Item	Description
-s	Generates a script of commands in the current directory that you can use to create a new output file or archive, but does not relink anything. It names the script *Base***.sh**, where *Base* is the basename of the input file with any suffix stripped off. It writes generated import and export files in the current directory as well. You can modify the script and the import and export files to customize the output objects.

Parameters

Item	Description
File	Specifies the input file.
ldFlag ...	Copies the specified **ld** command flags to the end of the generated **ld** command, overriding default options. **Note:** Do not use the **-o** flag in the *ldFlag* parameter to name the output file. To specify an alternate output file name, use the **rtl_enable** command's **-o** *Name* flag.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Note: Depending on the error, some output files may have been created.

Security

Access Control: Any User

Auditing Events: N/A

Examples

To create a new version of **libc.a** with runtime linking enabled, enter:

1. Create a directory for runtime version by entering:

```
mkdir /tmp/rtllibs
```

2. Make `/tmp/rtllibs` your current directory by entering:

```
cd /tmp/rtllibs
```

3. To create the runtime version of libc.a with the same name, enter:

```
rtl_enable -o libc.a /lib/libc.a
```

To use this version of libc.a when linking programs, use **-L /tmp/rtllibs** with the **ld** command.

Files

Item	Description
/usr/bin/rtl_enable	Contains the **rtl_enable** command. This is a symbolic link to **/usr/ccs/bin/rtl_enable**.

runacct Command

Purpose

Runs daily accounting.

Syntax

/usr/sbin/acct/runacct [*mmdd* [*State*]]

Description

The **runacct** command is the main daily accounting shell procedure. Normally initiated by the **cron** daemon, the **runacct** command processes connect, fee, disk, queuing system (printer), and process accounting data files for the current day to produce the binary daily report, **/var/adm/acct/nite(x)/dayacct**. The **runacct** command also prepares summary files for the **prdaily** procedure to prepare the ASCII daily report, **/var/adm/acct/sum(x)/rprt***mmdd*, or for billing purposes.

The **acctmerg** command adds the **dayacct** report to the cumulative summary report for the accounting period, **/var/adm/acct/sum(x)/tacct**. The **tacct** report is used by the **monacct** command to produce the monthly report, **/var/adm/acct/fiscal(x)**.

This command has two parameters that must be entered from the keyboard should you need to restart the **runacct** procedure. The date parameter, *mmdd*, enables you to specify the day and month for which you want to rerun the accounting. The *State* parameter enables a user with administrative authority to restart the **runacct** procedure at any of its states. For more information on restarting **runacct** procedures and on recovering from failures.

The **runacct** command protects active accounting files and summary files in the event of run-time errors, and records its progress by writing descriptive messages into the **/var/adm/acct/nite(x)/active** file. When the **runacct** procedure encounters an error, it sends mail to users root and adm, and exits.

The **runacct** procedure also creates two temporary files, **lock** and **lock1**, in the directory **/var/adm/acct/nite(x)**, which it uses to prevent two simultaneous calls to the **runacct** procedure. It uses the **lastdate** file (in the same directory) to prevent more than one invocation per day.

The **runacct** command breaks its processing into separate, restartable states. As it completes each state, it writes the name of the next state in the **/var/adm/acct/nite(x)/state** file. The **runacct** procedure processes the various states in the following order:

State	Actions
SETUP	Moves the active accounting files to working files and restarts the active files.
WTMPFIX	Verifies the integrity of the **wtmp** file, correcting date changes if necessary.
CONNECT1	Calls the **acctcon1** command to produce connect session records.
CONNECT2	Converts connect session records into total accounting records (**tacct.h** format).
PROCESS	Converts process accounting records into total accounting records (**tacct.h** format).
MERGE	Merges the connect and process total accounting records.
FEES	Converts the output of the **chargefee** command into total accounting records (**tacct.h** format) and merges them with the connect and process total accounting records.

State	Actions
DISK	Merges disk accounting records with connect, process, and fee total accounting records.
QUEUEACCT	Sorts the queue (printer) accounting records, converts them into total accounting records (**tacct.h** format), and merges them with other total accounting records.
MERGETACCT	Merges the daily total accounting records in the **daytacct** report file with the summary total accounting records in the **/var/adm/acct/sum(x)/tacct** report file.
CMS	Produces command summaries in the file **/var/adm/acct/sum(x)/cms**.
USEREXIT	If the **/var/adm/siteacct** shell file exists, calls it at this point to perform site-dependent processing.
CLEANUP	Deletes temporary files and exits.

Restarting runacct Procedures

To restart the **runacct** command after a failure, first check the **/var/adm/acct/nite(x)/active** file for diagnostic messages, then fix any damaged data files, such as **pacct** or **wtmp**. Remove the **lock** files and **lastdate** file (all in the **/var/adm/acct/nite(x)** directory), before restarting the **runacct** command. You must specify the *mmdd* parameter if you are restarting the **runacct** command. It specifies the month and day for which the **runacct** command is to rerun the accounting. The **runacct** procedure determines the entry point for processing by reading the **/var/adm/acct/nite(x)/statefile** file. To override this default action, specify the desired *state* on the **runacct** command line.

It is not usually a good idea to restart the **runacct** command in the SETUP *state*. Instead, perform the setup actions manually and restart accounting with the WTMPFIX state, as follows:

```
/usr/lib/acct/runacct mmdd WTMPFIX
```

If the **runacct** command fails in the PROCESS state, remove the last **ptacct** file, because it will be incomplete.

Flags

Item	Description
-X	Processes all available characters for each user name instead of truncating to the first 8 characters. The **-X** flag will also cause the **runacct** command and all commands it calls to use the **/var/adm/acct/sumx** and **/var/adm/acct/nitex** directories instead of the **/var/adm/acct/sum** and **/var/adm/acct/nite** directories.

Security

Access Control: This command should grant execute (x) access only to members of the **adm** group.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To start daily accounting procedures for system resources, add the following command line to a **crontab** file so the **runacct** command will be run automatically by the **cron** daemon:

```
0 4 * * 1-6 /usr/sbin/acct/runacct 2> \
/var/adm/acct/nite/accterr
```

To start daily accounting procedures with long user name support add the following line to the crontab file:

```
0 4 * * 1-6 /usr/sbin/acct/runacct -X 2> \
/var/adm/acct/nitex/accterr
```

This example shows the instructions that the **cron** daemon will read and act upon. The **runacct** command will run at 4 a.m. (04) every Monday through Saturday (1-6) and write all standard error output (2>) to the **/var/adm/acct/nite(x)/accterr** file. This command is only one of the accounting instructions normally given to the **cron** daemon.

2. To start daily accounting procedures for system resources from the command line (start the **runacct** command), enter the following:

```
nohup /usr/sbin/acct/runacct 2> \
/var/adm/acct/nite/accterr &
```

Although it is preferable to have the **cron** daemon start the **runacct** procedure automatically (see example 1), you can give the command from the keyboard. The **runacct** command will run in the background (&), ignoring all INTERRUPT and QUIT signals (the **nohup** command), and write all standard error output (2>) to the **/var/adm/acct/nite/accterr** file.

3. To restart the system accounting procedures for a specific date, enter a command similar to the following:

```
nohup /usr/sbin/acct/runacct 0601 2>> \
/var/adm/acct/nite/accterr &
```

This example restarts **runacct** for the day of June 1 (0601). The **runacct** command reads the file **/var/adm/acct/nite(x)/statefile** to find out the state with which to begin. The **runacct** command will run in the background (&), ignoring all INTERRUPT and QUIT signals (**nohup**). Standard error output (2) is added to the end (>>) of the **/var/adm/acct/nite(x)/accterr** file.

4. To restart the system accounting procedures for a particular date at a specific state, enter a command similar to the following:

```
nohup /usr/sbin/acct/runacct 0601 MERGE 2>> \
 /var/adm/acct/nite(x)/accterr &
```

This example restarts the **runacct** command for the day of June 1 (0601), starting with the MERGE state. The **runacct** command will run in the background (&), ignoring all INTERRUPT and QUIT signals (the **nohup** command). Standard error output (2) is added to the end (>>) of the **/var/adm/acct/nite(x)/accterr** file.

Files

Item	Description
/var/adm/wtmp	Log in/log off history file.
/var/adm/pacct*	Process accounting file.
/var/adm/acct/nite(x)/daytacct	Disk usage accounting file.
/var/adm/qacct	Active queue accounting file.
/var/adm/fee	Record of fees charged to users.
/var/adm/acct/sum(x)/*	Command and total accounting summary files.
/var/adm/acct/nite(x)/ptacct*.*mmdd*	Concatenated version of **pacct** files.
/var/adm/acct/nite(x)/active	The **runacct** message file.
/var/adm/acct/nite(x)/lock*	Prevents simultaneous invocation of **runacct**.
/var/adm/acct/nite(x)/lastdate	Contains last date **runacct** was run.

Item	Description
/var/adm/acct/nite(x)/statefile	Contains current state to process.

runact Command

Purpose

Runs an action on a resource class.

Syntax

`runact -s` "*selection_string*" [`-N` { *node_file* | `"-"` }] [`-f` *resource_data_input_file*] [`-l` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] *resource_class action* [*in_element*=*value*...] [*rsp_element*...]

`runact -r` [`-f` *resource_data_input_file*] [`-l` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] *resource_handle action* [*in_element*=*value*...] [*rsp_element*...]

`runact -c` [`-f` *resource_data_input_file*] [`-n` *node_name*] [`-l` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] *resource_class action* [*in_element*=*value*...] [*rsp_element*...]

`runact -C` *domain_name*... [`-f` *resource_data_input_file*] [`-l` | `-t` | `-d` | `-D` *delimiter*] [`-x`] [`-h`] [`-TV`] *resource_class action* [*in_element*=*value*...] [*rsp_element*...]

Description

The `runact` command requests that the RMC subsystem run the specified action on the specified resource class.

Instead of specifying multiple node names in *selection_string*, you can use the `-N` *node_file* flag to indicate that the node names are in a file. Use `-N "-"` to read the node names from standard input.

Before you run this command, use the `lsactdef` command to list the resource class actions that are supported by this resource class. Also, use the `lsactdef` command to list the required input action elements that must be specified when invoking an action. The `lsactdef` command also identifies the data type for each input element. The value specified for each input element must match this data type.

Flags

-c
Invokes the action on the resource class.

To invoke the class action on a globalized resource class on all peer domains defined on the management server, set **CT_MANAGEMENT_SCOPE=3** and use the **-c** flag.

-C *domain_name...*
Invokes a class action on a globalized resource class on one or more RSCT peer domains that are defined on the management server. Globalized classes are used in peer domains and management domains for resource classes that contain information about the domain.

-f *resource_data_input_file*
Specifies the name of the file that contains resource action input elements and values. Use the `lsactdef` command with the `-i` flag to generate a template for this input file.

-d
Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the `-D` flag if you want to change the default delimiter.

-D *delimiter*
Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify a delimiter other than the default colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

-l
Specifies "long" format — one entry per line. This is the default display format.

-n *node_name*
Specifies the name of the node on which to run the class action. You can only use this flag in conjunction with the **-c** flag.

-N { *node_file* | "-" }
Specifies that node names are read from a file or from standard input. Use `-N` *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (*#*) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use `-N "-"` to read the node names from standard input.

The CT_MANAGEMENT_SCOPE environment variable determines the scope of the cluster. If CT_MANAGEMENT_SCOPE is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and CT_MANAGEMENT_SCOPE is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-r "*resource_handle*"
Specifies a resource handle. The resource handle must be specified in this format:

```
"0xnnnn 0xnnnn 0xnnnnnnnn 0xnnnnnnnn 0xnnnnnnnn 0xnnnnnnnn"
```

where n is a hexadecimal character. Use this flag to invoke the action on the resource that matches *resource_handle*.

-s "*selection_string*"
Specifies a selection string. All selection strings must be enclosed within either double or single quotation marks. If the selection string contains double quotation marks, enclose the entire selection string in single quotation marks. For example:

```
    -s 'Name == "testing"'
    -s 'Name ?= "test"'
```

Only persistent attributes can be listed in a selection string. For information on how to specify selection strings, see the *Administering RSCT*.

-t
Specifies table format. Each attribute is displayed in a separate column, with one resource per line.

-x
Suppresses header printing.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software-service organization's use only.

-V
Writes the command's verbose messages to standard output.

Parameters

action
Specifies the name of the action to be invoked.

in_element=value...
Specifies the action input element names and values. If you use the `-f` flag, don't enter any *in_element=value* pairs on the command line.

in_element is any of the input structured data element names. There should be one *in_element_n=value* pair for each of the defined structured data (SD) input elements for the specified action. Use `lsactdef` with the `-s i` flag to list the input elements for a particular resource class and action. Use `lsactdef -i` to generate an input file template, which, after appropriate editing, can be used as the input file.

value must be the appropriate datatype for the specified element. For example, if `NodeNumber` is defined as a `uint32` datatype, enter a positive numeric value.

resource_class
Specifies the name of the resource class with the actions that you want to invoke.

resource_handle
Specifies the resource handle for the resource and class with the actions that you want to invoke.

rsp_element
Specifies one or more of action response structured data element names. If you specify one or more element names, only those elements are displayed in the order specified. If you do not specify any element names, all elements of the response are displayed.

Security

This command requires `root` authority.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output.

The command output and all verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To invoke the **TestClassAction** resource class action on the resource class `IBM.Example`, enter:

```
runact -c IBM.Example TestClassAction Int32=99
```

The output will look like this:

```
Resource Class Action Response for: TestClassAction
sd_element 1:
   Int32 = 99
```

Location

/opt/rsct/bin/runact
Contains the `runact` command

runcat Command

Purpose

Pipes output data from the **mkcatdefs** command to the **gencat** command.

Syntax

runcat *CatalogName SourceFile* [*CatalogFile*]

Description

The **runcat** command invokes the **mkcatdefs** command and pipes the message catalog source data (the output from **mkcatdefs**) to the **gencat** program.

The file specified by the *SourceFile* parameter contains the message text with your symbolic identifiers. The **mkcatdefs** program uses the *CatalogName* parameter to generate the name of the symbolic definition file by adding **_msg.h** to the end of the *CatalogName* value, and to generate the symbolic name for the catalog file by adding **MF_** to the beginning of the *CatalogName* value. The definition file must be included in your application program. The symbolic name for the catalog file can be used in the library functions (such as the **catopen** subroutine).

The *CatalogFile* parameter is the name of the catalog file created by the **gencat** command. If you do not specify this parameter, the **gencat** command names the catalog file by adding **.cat** to the end of the *CatalogName* value. This file name can also be used in the **catopen** library function.

Example

To generate a catalog named `test.cat` from the message source file `test.msg`, enter:

```
runcat test test.msg
```

File

Item	Description
/usr/bin/runcat	Contains the **runcat** command.

runlpcmd Command

Purpose

Runs a least-privilege (LP) resource.

Syntax

To run an LP resource:

- On the local node:

  `runlpcmd -N` *resource_name* | *RunCmdName* [`-h`] [`-TV`] [`"`*flags_and_parms*`"`]

- On all nodes in a domain:

  `runlpcmd -a -N` *resource_name* | *RunCmdName* [`-h`] [`-TV`] [`"`*flags_and_parms*`"`]

- On a subset of nodes in a domain:

  `runlpcmd -n` *host1* [`,`*host2*`,`...] `-N` *resource_name* | *RunCmdName* [`-h`] [`-TV`] [`"`*flags_and_parms*`"`]

Description

The `runlpcmd` command runs an LP resource, which is a `root` command or script to which users are granted access based on permissions in the LP access control lists (ACLs). You can use the `runlpcmd` command to call the LP command corresponding to a particular *RunCmdName* value with access permissions that match the permissions of the calling user. When `runlpcmd` is called with the `-N` flag, the LP command that is specified by the *resource_name* parameter is run. Specify all parameters and flag needed for command invocation using the *flags_and_parms* parameter. If this parameter is not specified, an empty string is passed to the LP command. This is the default.

If the `CheckSum` attribute value is `0`, `runlpcmd` returns an error if the `ControlFlags` value is set to check for `CheckSum`; otherwise, no errors are returned. If the `ControlFlag` attribute of the LP command

was set to validate the CheckSum before the LP command was run, runlpcmd performs such a check. The command is run only if the calculated CheckSum matches the value of the corresponding CheckSum attribute. If the two do not match, the command is rejected. If, however, the ControlFlags attribute is set to the default value, CheckSum validation is not performed.

You can specify the *RunCmdName* parameter along with with the -N *resource_name* flag and parameter combination. However, one restriction applies when you use the *RunCmdName* parameter. If more than one resource matches the *RunCmdName* value and the permissions of the calling user, runlpcmd returns an error. If one match exists for the *RunCmdName* value and the the permissions of the calling user, runlpcmd *RunCmdName* returns successfully. In order to circumvent this restriction, runlpcmd also lets users run LP commands by specifying their unique names, using the -N *resource_name* flag and parameter combination.

Before calling the LP command, runlpcmd checks to see if a FilterScript value exists. If so, it passes the FilterArg value and the *flags_and_parms* parameter string specified on the command line to FilterScript. If FilterScript returns a 0, runlpcmd calls the LP command. If FilterScript execution resulted in a non-zero value, runlpcmd returns an error. If FilterScript was empty, runlpcmd performs some checks, as specified in ControlFlags, and then calls the LP command directly.

The output of this command may include "RC=*return_code*" as the last line.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the -a flag. If you want this command to run on a subset of nodes in a domain, use the -n flag. Otherwise, this command runs on the local node.

Flags

-a
Changes one or more resources on all nodes in the domain. The CT_MANAGEMENT_SCOPE environment variable's setting determines the cluster scope. If CT_MANAGEMENT_SCOPE is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The runlpcmd command runs once for the first valid scope that the LP resource manager finds. For example, suppose a management domain and a peer domain exist and the CT_MANAGEMENT_SCOPE environment variable is not set. In this case, runlpcmd –a runs in the management domain. To run runlpcmd –a in the peer domain, you must set CT_MANAGEMENT_SCOPE to 2.

-n *host1*[,*host2*,...]
Specifies the node or nodes in the domain on which the LP resource is to be changed. By default, the LP resource is changed on the local node. The –n flag is valid only in a management or peer domain. If the CT_MANAGEMENT_SCOPE variable is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The runlpcmd command runs once for the first valid scope that the LP resource manager finds.

-N *resource_name*
Specifies the name of the LP resource that you want to run on one or more nodes in the domain.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error.

-V
Writes the command's verbose messages to standard output.

Parameters

RunCmdName
Specifies the name of the LP resource that you want to run on one or more nodes in the domain.

"*flags_and_parms*"
Specifies the flags and parameters that are required input for the LP command or script. If this parameter is not specified, an empty string is passed to the LP command. This is the default.

Security

To run the `runlpcmd` command, you need:

- read permission in the Class ACL of the `IBM.LPCommands` resource class.
- execute permission in the Resource ACL.

  As an alternative, the Resource ACL can direct the use of the Resource Shared ACL if this permission exists in the Resource Shared ACL.

Permissions are specified in the LP ACLs on the contacted system. See the `lpacl` file for general information about LP ACLs and the *RSCT Administration Guide* for information about modifying them.

Exit Status

0
The command has run successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
The resource was not found.

Environment Variables

CT_CONTACT
Determines the system that is used for the session with the RMC daemon. When `CT_CONTACT` is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the LP resources that are processed.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to process the LP resources. The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. When the `-V` flag is specified, this command's verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

To run the LP resource called LP1, which has required input flags and parameters `-a -p User Group`, enter:

```
runlpcmd LP1 "-a -p User Group"
```

Location

/opt/rsct/bin/runlpcmd
Contains the `runlpcmd` command

rup Command

Purpose

Shows the status of a remote host on the local network.

Syntax

/usr/bin/rup [**-h** | **-l** | **-t**] [*Host ...*]

Description

The **rup** command displays the status of a remote host by broadcasting on the local network and then displaying the responses it receives. Specify a flag if you want to sort the output. If you do not specify a flag, the **rup** command displays responses in the order they are received. If you specify multiple hosts on the command line, the **rup** command ignores any flags and displays output in the order you specified the hosts. You must use the **inetd** daemon.

Notes:

1. Broadcasting does not work through gateways. Therefore, if you do not specify a host, only hosts on your network can respond to the **rup** command.
2. Load-average statistics are not kept by the kernel. The load averages are always reported as 0 (zero) by this command.

Flags

Item	Description
-h	Sorts the display alphabetically by host name.
-l	Sorts the display by load average.
-t	Sorts the display by length of runtime on the network.

Examples

1. To find out the status of all hosts on the network and to sort the list alphabetically by host name, enter:

   ```
   /usr/bin/rup  -h
   ```
2. To display a list of all hosts on the network according to each machine's load average, enter:

   ```
   /usr/bin/rup  -l
   ```
3. To display the status of a host, enter:

   ```
   /usr/bin/rup brutus
   ```

   In this example, the **rup** command displays the status of the host named `brutus`.
4. To display the status of all hosts on the network sorted by each machine's length of runtime, enter:

   ```
   /usr/bin/rup  -t
   ```

Files

Item	Description
html	

ruptime Command

Purpose

Shows the status of each host on a network.

Syntax

ruptime [**-a**] [**-r**] [**-l** | **-t** | **-u**]

Description

The **/usr/bin/ruptime** command displays the status of each host that is on a local network and is running the **rwhod** daemon. The status lines are sorted by host name unless the **-l**, **-t**, or **-u** flag is indicated. The status information is provided in packets broadcast once every 3 minutes by each network host running the **rwhod** daemon. Any activity (such as power to a host being turned on or off) that takes place between broadcasts is not reflected until the next broadcast. Hosts for which no status information is received for 11 minutes are reported as down.

Output is in the following format: hostname, status, time, number of users, and load average. Load average represents the load averages over 1-, 5-, and 15-minute intervals prior to a server's transmission. The load averages are multiplied by 10 to represent the value in decimal format.

Flags

Item	Description
-a	Includes all users. Without this flag, users whose sessions are idle an hour or more are not included.
-l	Sorts the list by the load average.
-r	Reverses the sort order. The **-r** flag should be used with the **-l**, **-t** or **-u** flag.
-t	Sorts the list by the uptime.
-u	Sorts the list by the number of users.

Examples

1. To get a status report on the hosts on the local network, enter:

   ```
   ruptime
   ```

   Information similar to the following is displayed:

   ```
   host1      up      5:15,   4 users,   load 0.09, 0.04, 0.04
   host2      up      7:45,   3 users,   load 0.08, 0.07, 0.04
   host7      up      7:43,   1 user,    load 0.06, 0.12, 0.11
   ```

2. To get a status report sorted by load average, enter:

   ```
   ruptime  -l
   ```

   Information similar to the following is displayed:

   ```
   host2      up      7:45,   3 users,   load 0.08, 0.07, 0.04
   host1      up      5:18,   4 users,   load 0.07, 0.07, 0.04
   host7      up      7:43,   1 user,    load 0.06, 0.12, 0.11
   ```

Files

Item	Description
/var/spool/rwho/whod.*	Indicates data files received from remote **rwhod** daemons.

ruser Command

Purpose

Directly manipulates entries in three separate system databases that control foreign host access to programs.

Syntax

To Add or Delete a Database File Name Entry

ruser { **-a** | **-d** } { **-f** "*UserName ...*" | **-p** "*HostName ...*" | **-r** "*HostName ...*" }

To Delete or Display All Name Entries in a Database File

ruser { **-X** | **-s** } { **-F** | **-P** | **-R** } [**-Z**]

Description

The **ruser** low-level command adds or deletes entries in three separate system databases. Which database you are manipulating is determined by using the **-p**, **-r**, or **-f** flags. In addition, the **ruser** command can show one or all entries in one of the databases. Each database is a list of names. The three databases are as follows:

- **/etc/ftpusers** file
- **/etc/hosts.equiv** file
- **/etc/hosts.lpd** file

Note: The **-p** and **-r** options can be used together to add a name to databases at the same time, but the **-f** option cannot be used with either.

You could also use the System Management Interface Tool (SMIT) **smit users** fast path to run this command or type:

```
smit rprint
```

Flags

Item	Description
-a	Adds a name to the database. The **-a** flag must be used with either the **-p**, **-r**, or **-f** flag.
-d	Deletes a name from the database. Must be used with either the **-p**, **-r**, or **-f** flag.
-F	Deletes or shows all entries in the **/etc/ftpusers** file. Use this flag with the **-X** flag to delete all entries. Use this flag with the **-s** flag to show all entries.
-f "*UserName ...***"**	Adds or deletes the user name specified by the *UserName* variable to the **/etc/ftpusers** database that contains a list of local user names that cannot be used by remote FTP clients. The **-f** flag must be used with either the **-a** or **-d** flag.
-P	Deletes or shows all entries in the **/etc/hosts.lpd** file. Use this flag with the **-X** flag to delete all entries. Use this flag with the **-s** flag to show all entries.
-p "*HostName ...***"**	Adds or deletes the host name, specified by the *HostName* variable, in the database that specifies which foreign host may print on your machine. The **-p** flag must be used with either the **-a** or **-d** flag.
-R	Deletes or shows all entries in the **/etc/hosts.equiv** file. Use this flag with the **-X** flag to delete all entries. Use this flag with the **-s** flag to show all entries.
-r "*HostName ...***"**	Adds or deletes the host name, specified by the *HostName* variable, in the **/etc/hosts.equiv** database that specifies which foreign host may perform the remote commands (**rlogin**, **rcp**, **rsh**, or **print**) on your machine. The **-r** flag must be used with either the **-a** or **-d** flag.
-s	Shows all entries in the database. Use this flag with either the **-P**, **-R**, or **-F** flag.
-X	Deletes all names from the database. Use this flag with either the **-P**, **-R**, or **-F** flag.
-Z	The **-s** flag is required when the **-Z** flag is specified. If the **-Z** flag is specified, a brief title is displayed before the database display.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To add an entry in the **/etc/hosts.lpd** database, which specifies which foreign host may print on the local machine, type the command in the following format:

   ```
   ruser  -a  -p "host1"
   ```

   In this example, the foreign host is `host1`.
2. To delete an entry in the database that controls printing only (**/etc/hosts.lpd**), and also delete the same name from the database that controls remote access for the **rlogin**, **rcp**, and **rsh** commands (**/etc/hosts.equiv**), type:

   ```
   ruser  -d  -r "host2"  -p "host1"
   ```

   In this example, the host from which the database entry is deleted is `host1`.

rusers Command

Purpose

Reports a list of users logged on to remote machines.

Syntax

/usr/bin/rusers [**-a**] [**-l**] [**-u** | **-h** | **-i**] [*Host ...*]

Description

The **rusers** command produces a list of users who are logged on to remote machines. The **rusers** command does this by broadcasting to each machine on the local network and printing the responses it receives. Normally, the system prints the responses in the order they are received. To change this order, specify one of the flags. In addition, when you provide a *Host* parameter, the **rusers** command queries the host or hosts you specify, rather than broadcasting to all hosts.

By default, each entry contains a list of users for each machine. Each of these entries includes the names of all users logged in that machine. In addition, when the user does not type into the system for a minute or more, the **rusers** command reports the user's idle time.

A remote host responds only if it is running the **rusersd** daemon, which is normally started from the **inetd** daemon.

Note: Broadcasting does not work through gateways. Therefore, if you do not specify a host, only hosts on your network can respond to the **rusers** command.

Flags

Item	Description
-a	Gives a report for a machine even if no users are logged in.
-h	Sorts alphabetically by host name.
-i	Sorts by idle time.
-l	Gives a longer listing similar to the **who** command.
-u	Sorts by number of users.

Examples

1. To produce a list of the users on your network that are logged in remote machines, enter:

```
rusers
```

2. To produce a list of users sorted alphabetically by host name, enter:

   ```
   rusers -h
   ```

3. To produce a list of users on a host, enter:

   ```
   rusers -h pluto
   ```

   In this example, the **rusers** command produces a list of users on the host named `pluto`.

4. To produce a list of users logged in remote machines and sorted according to each machine's length of idle time, enter:

   ```
   rusers -i
   ```

5. To produce a list of users logged in remote machines and sorted by the number of users logged in, enter:

   ```
   rusers -u
   ```

Files

Item	Description
/etc/inetd.conf	TCP/IP configuration file that starts RPC daemons and other TCP/IP daemons.

rusersd Daemon

Purpose

Responds to queries from the **rusers** command.

Syntax

/usr/lib/netsvc/rusers/rpc.rusersd

Description

The **rusersd** daemon is a server that responds to queries from the **rusers** command by returning a list of users currently on the network. This daemon is normally started by the **inetd** daemon.

Files

Item	Description
/etc/inetd.conf	TCP/IP configuration file that starts RPC daemons and other TCP/IP daemons.
/etc/inetd.conf	Contains information on users logged in to the system.

rvsdrestrict Command

Purpose

rvsdrestrict – Displays and sets the run level of the Recoverable virtual shared disk subsystem. This command must be issued before the RVSD subsystem will start.

Syntax

rvsdrestrict
{`-l` | `-s` {`RVSD4.1` | RESET}}

Description

The `rvsdrestrict` command is used to restrict the level at which the Recoverable virtual shared disk subsystem will run. If a node has a lower level of the RVSD software installed than what is set with this command, then the RVSD subsystem will not start on that node.

This command does not dynamically change RVSD subsystem run levels across the peer domain. An RVSD subsystem instance will only react to this information after being restarted. If your peer domain runs at a given level, and you want to override this level, you must:

1. Stop the RVSD subsystem on all nodes.
2. Override the level.
3. Restart the RVSD subsystem.

Flags

-l
Lists the current RVSD subsystem run level.

-s
Sets the RVSD subsystem run level.

Parameters

None.

Security

You must have root authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

1. To set the RVSD subsystem run level to RVSD4.1, you would issue the command:

```
rvsdrestrict -s RVSD4.1
```

Location

`/opt/rsct/vsd/bin/rvsdrestrict`

rwall Command

Purpose

Sends messages to all users on the network.

Syntax

To Send a Message to Specified Hosts

/usr/sbin/rwall *HostName ...*

To Send a Message to Specified Networks

/usr/sbin/rwall -n *NetworkGroup ...*

To Send a Message to Specified Hosts on a Network

/usr/sbin/rwall -h *HostName ...* **-n** *NetworkGroup*

Description

The **rwall** command sends messages to all users on the network. To do this, the **rwall** command reads a message from standard input until it reaches an end-of-file character. The **rwall** command takes this message, which begins with the line `Broadcast Message...`, and broadcasts it to all users logged in to the specified host machines. Users receive messages only if they are running the **rwalld** daemon, which is started by the **inetd** daemon.

> **Note:** The time out is fairly short. This enables the **rwall** command to send messages to a large group of machines (some of which may be down) in a reasonable amount of time. Thus the message may not get through to a heavily loaded machine.

Flags

Item	Description
-h	Sends the message to machines specified by the *HostName* parameter.
-n	Sends the message to specific network groups only. Network groups are defined in the **netgroup** file.

Examples

1. To send a message to a host named `neptune`, enter:

   ```
   /usr/sbin/rwall neptune
   ```

   Type in your message. When you are done, enter:

   ```
   Ctrl D
   ```

2. To send a message to a host named `neptune` and every host in the `cosmos` netgroup, enter:

   ```
   rwall -n cosmos -h neptune
   ```

   Type in your message. When you are done, enter:

```
Ctrl D
```

Files

Item	Description
/etc/inetd.conf	TCP/IP configuration file that starts RPC daemons and other TCP/IP daemons.
/etc/netgroup	Contains information about each user group on the network.

rwalld Daemon

Purpose

Handles requests from the **rwall** command.

Syntax

/usr/lib/netsvc/rwall/rpc.rwalld

Description

The **rwalld** daemon handles requests from the **rwall** command. The **inetd** daemon invokes the **rwalld** daemon.

Files

Item	Description
/etc/inetd.conf	Specifies the TCP/IP configuration.

rwho Command

Purpose

Shows which users are logged in to hosts on the local network.

Syntax

rwho [**-a**]

Description

The **/usr/bin/rwho** command displays the user name, host name, and start date and time of each session for everyone on the local network who is currently logged in to a host running the **rwhod** daemon. If a workstation is inactive for at least 3 minutes, the **rwho** command reports the idle time as a number of minutes in the last column. After an hour of inactivity, a user is not included unless the **-a** flag is specified.

> **Note:** Since this command displays a lot of output, use this command with caution if the local network has a large number of users.

Status information is broadcast once every 3 minutes by each network host running the **rwhod** daemon. Any activity (such as a user logging on or off) that takes place between broadcasts is not reflected until the next broadcast.

Flags

Item	Description
-a	Includes all users. Without this flag, users whose sessions are idle an hour or more are not included in the report.

Example

To get a report of all users currently logged in to hosts on the local network, enter:

```
rwho
```

Information similar to the following is displayed:

```
bob     host2:pts5       Nov 17 06:30 :20
bob     host7:console    Nov 17 06:25 :25
fran    host1:pts0       Nov 17 11:20 :51
fran    host1:pts8       Nov 16 15:33 :42
fran    host4:console    Nov 17 16:32
server  host2:console    Nov 17 06:58 :20
alice   host2:pts6       Nov 17 09:22
```

Files

Item	Description
/var/spool/rwho/whod.*	Indicates data files received from remote **rwhod** daemons.

rwhod Daemon

Purpose

Provides the server function for the **rwho** and **ruptime** commands.

Syntax

> **Note:** Use SRC commands to control the **rwhod** daemon from the command line. Use the **rc.tcpip** file to start the daemon with each system startup.

/usr/sbin/rwhod

Description

The **/usr/sbin/rwhod** daemon maintains the database used by the **rwho** and **ruptime** commands. Once started, the **rwhod** daemon operates as both producer and consumer of status information.

As a producer of status information, the **rwhod** daemon queries the state of the local host approximately every 3 minutes. It then constructs status messages and broadcasts them to the local network.

As a consumer of status information, the **rwhod** daemon listens for status messages from **rwhod** servers on remote hosts. When the **rwhod** daemon receives a status message, it validates the received status message. It then records the message in the **/var/spool/rwho** directory. (The **rwho** and **ruptime** commands use the files in the **/var/spool/rwho** directory to generate their status listings.)

The **rwhod** daemon broadcasts and receives status messages using the **rwho** socket as specified in the **/etc/services** file.

When creating these messages, the **rwhod** daemon calculates the entries for the average CPU load for the previous 1-, 5-, and 15-minute intervals. Before broadcasting these messages, the **rwhod** daemon converts them to the byte order that the network can use.

When the **rwhod** daemon receives messages on the **rwho** socket, it discards any that do not originate from an **rwho** socket. Additionally, it discards any messages that contain unprintable ASCII characters. When the **rwhod** daemon receives a valid message, it places the message in a **whod.***HostName* file in the **/var/spool/rwho** directory, overwriting any file with the same name.

The **rwhod** daemon should be controlled using the System Resource Controller (SRC). Entering `rwhod` at the command line is not recommended.

Manipulating the rwhod Daemon with the System Resource Controller

The **rwhod** daemon is a subsystem controlled by the System Resource Controller (SRC). The **rwhod** daemon is a member of the **tcpip** system group. This daemon is disabled by default and can be manipulated by the following SRC commands:

Item	Description
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
traceson	Enables tracing of a subsystem, group of subsystems, or a subserver.
tracesoff	Disables tracing of a subsystem, group of subsystems, or a subserver.
tracesoff	Gets the status of a subsystem, group of subsystems, or a subserver.

Examples

1. To start the **rwhod** daemon, enter the following:

   ```
   startsrc -s rwhod
   ```

   This command starts the daemon. You can use this command in the **rc.tcpip** file or on the command line. The **-s** flag specifies that the subsystem that follows is to be started.
2. To stop the **rwhod** daemon normally, enter the following:

   ```
   stopsrc -s rwhod
   ```

   This command stops the daemon. The **-s** flag specifies that the subsystem that follows is to be stopped.
3. To get a short status report from the **rwhod** daemon, enter the following:

   ```
   lssrc -s rwhod
   ```

   This command returns the name of the daemon, the process ID of the daemon, and the state of the daemon (active or inactive).
4. To enable tracing for **rwhod** daemon, enter the following:

   ```
   traceson -s rwhod
   ```

   This command enables socket level debugging. Use the **trpt** command to look at the output of this example command.

Files

Item	Description
/etc/utmp	Contains status information on users that are logged in to the local host.
/var/spool/rwho/*	Contains files used by the **rwho** and **ruptime** commands to generate their status list.
/var/spool/rwho/ whod.*HostName*	Contains the latest status information for the host specified by the *HostName* parameter.

s

The following AIX commands begin with the letter *s*.

sa Command

Purpose

Summarizes accounting records.

Syntax

/usr/sbin/sa [**-a**] [**-b**] [**-c**] [**-C**] [**-d**] [**-D**] [**-i**] [**-j**] [**-k**] [**-K**] [**-l**] [**-m**] [**-n**] [**-r**] [**-s**] [**-t**] [**-u**] [**-v***Number* [**-f**]] [**-S***SaveFile*] [**-U***UserFile*] [*File* ...]

Description

The **sa** command summarizes the information in the file that collects the raw accounting data, either the **/var/adm/pacct** file or the file specified by the *File* parameter, and writes a usage summary report to the **/var/adm/savacct** file. Then the **sa** command deletes the data in the **/var/adm/pacct** file so it can collect new accounting information. The next time the **sa** command executes, it reads the usage summary and the new data and incorporates all the information in its report.

The flags used with the **sa** command vary the type of information that is reported. The reports can contain the following fields:

Item	Description
`avio`	Indicates the average number of I/O operations per execution.
`cpu`	Indicates the sum of user and system time (in minutes).
k	Indicates the average K-blocks of CPU-time per execution.
`k*sec`	Indicates the CPU storage integral in kilo-core seconds.
`re`	Indicates the minutes of real time.
`s`	Indicates the minutes of system CPU time.
`tio`	Indicates the total number of I/O operations.
`u`	Indicates the minutes of user CPU time.

If you run the **sa** command without specifying any flags, the summary report includes the number of times each command was called as well as the `re`, `cpu`, `avio`, and k fields.

> **Note:** The **-b**, **-d**, **-D**, **-k**, **-K**, and **-n** flags determine how output is sorted. If you specify more than one of these flags on the command line, only the last one specified will take effect.

Summary files created under this release of the base operating system are saved in a format that supports large user IDs (8 characters or longer). Summary files created under previous releases may be in the old format that supports only user IDs of up to 7 characters. The **sa** command recognizes and supports both formats of the summary file. If you need to convert old format summary files to the new format, use the **-C** flag instead of the **-s** flag. You need to do this conversion only once. After converting you can use either the **-s** or the **-C** flag.

Flags

Item	Description
-a	Prints all command names, including those with unprintable characters. Commands that were used once are placed under the `other` category.
-b	Sorts output by the sum of user and system time divided by the number of calls. Otherwise, output is the sum of user and system time.
-c	Prints the time used by each command as a percentage of the time used by all the commands. This is in addition to the user, system and real time.
-C	Merges the accounting file into the summary file. If the summary file is in the old format, it is converted into the new format.
-d	Sorts the output by the average number of disk I/O operations.
-D	Sorts and prints the output by the total number of disk I/O operations.
-f	Does not force interactive threshold compression. This flag must be used with the **-v** flag.
-i	Reads only the raw data, not the summary file.
-j	Prints the number of seconds per call instead of the total minutes per category.
-k	Sorts the output by the average CPU time.
-K	Sorts and prints the output by the CPU-storage integral.
-l	Separates system and user time, instead of combining them.
-m	Prints the number of processes and the number of CPU minutes for each user.
-n	Sorts output by the number of calls.
-r	Reverses the order of the sort.
-s	Merges the accounting file into the summary file.
-S *SaveFile*	Uses the specified saved file as the command summary file, instead of the **/var/adm/savacct** file.
-t	Prints the ratio of real time to the sum of user and system time for each command.
-u	Suspends all other flags and prints the user's numeric ID and the command name for each command.
-U *UserFile*	Uses the specified file instead of the **/var/adm/usracct** file to accumulate the per-user statistics printed by the **-m** flag.
-v *Number*	Types the name of each command used the specified number times or fewer. When queried, if you type `y` (yes), the command is added to the `junk` category and appears in future summaries as part of that category.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To summarize accounting records for all the commands in the **/var/adm/pacct** file, enter:

```
sa -a
```

Commands used only once are placed under the `other` field.

2. To summarize accounting records by average CPU time, enter:

   ```
   sa -k
   ```

Files

Item	Description
/usr/sbin/sa	Contains the **sa** command.
/etc/sa	Contains the symbolic link to the **sa** command.
/var/adm/pacct	Contains raw accounting records.
/var/adm/savacct	Contains summary accounting records.
/var/adm/usracct	Contains summary accounting records by user.

sa1 Command

Purpose

Collects and stores binary data in the **/var/adm/sa/sa***dd* file.

Syntax

/usr/lib/sa/sa1 [*Interval Number*]

Description

The **sa1** command is a shell procedure variant of the **sadc** command and handles all of the flags and parameters of that command. The **sa1** command collects and stores binary data in the **/var/adm/sa/sa***dd* file, where *dd* is the day of the month. The *Interval* and *Number* parameters specify that the record should be written *Number* times at *Interval* seconds. If you do not specify these parameters, a single record is written. You must have permission to write in the **/var/adm/sa** directory to use this command.

The **sa1** command is designed to be started automatically by the **cron** command. If the **sa1** command is not run daily from the **cron** command, the **sar** command displays a message about the nonexistence of the **/usr/lib/sa/sa1** data file.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To create a daily record of **sar** activities, place the following entry in your adm **crontab** file:

```
0 8-17 * * 1-5 /usr/lib/sa/sa1 1200 3 &
```

Files

Item	Description
/var/adm/sa	Specifies the directory containing the daily data files.

Item	Description
/var/adm/sa/sa*dd*	Contains the daily data file, where the *dd* parameter is a number representing the day of the month.
/usr/lib/sa/sa1	Contains the **sa1** command.

sa2 Command

Purpose

Writes a daily report in the **/var/adm/sa/sar***dd* file.

Syntax

/usr/lib/sa/sa2

Description

The **sa2** command is a variant shell procedure of the **sar** command, which writes a daily report in the **/var/adm/sa/sar***dd* file, where *dd* is the day of the month. The **sa2** command handles all of the flags and parameters of the **sar** command.

The **sa2** command is designed to be run automatically by the **cron** command and run concurrently with the **sa1** command.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To run the **sa2** command daily, place the following entry in the root **crontab** file:

```
5 18 * * 1-5 /usr/lib/sa/sa2 -s 8:00 -e 18:01 -i 3600 -ubcwyaqvm &
```

This will generate a daily report called **/var/adm/sa/sar***dd*. It will also remove a report more than one week old.

Files

Item	Description
/var/adm/sa	Specifies the directory containing the daily data files.
/var/adm/sa/sar*dd*	Contains daily data file, where the *dd* parameter is a number representing the day of the month.
/usr/lib/sa/sa2	The path to the shell script of the **sa2** command.

sact Command

Purpose

Displays current SCCS file-editing status.

Syntax

sact *File ...*

Description

The **sact** command reads Source Code Control System (SCCS) files and writes to standard output the contents, if any, of the p-file associated with the specified value of the *File* variable. The p-file is created by the **get -e** command. If a **-** (minus sign) is specified for the *File* value, the **sact** command reads standard input and interprets each line as the name of an SCCS file. If the *File* value is a directory, the **sact** command performs its actions on all SCCS files.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

To display the contents of a p-file, enter:

```
sact File
```

Files

Item	Description
/usr/bin/sact	Contains the path to the SCCS **sact** command.

sadc Command

Purpose

Provides a system data collector report.

Syntax

/usr/lib/sa/sadc [*Interval Number*] [*Outfile*]

/usr/lib/sa/sa1 [*Interval Number*]

/usr/lib/sa/sa2

Description

The **sadc** command, the data collector, samples system data a specified number of times (*Number*) at a specified interval measured in seconds (*Interval*). It writes in binary format to the specified outfile or to the standard output. When both *Interval* and *Number* are not specified, a dummy record, which is used at system startup to mark the time when the counter restarts from 0, will be written. The **sadc** command is intended to be used as a backend to the **sar** command.

The operating system contains a number of counters that are incremented as various system actions occur. The various system actions include:

- System Configuration Parameters
- System unit utilization counters
- Buffer usage counters
- Disk and tape I/O activity counters
- Tty device activity counters
- Switching and subroutine counters
- File access counters
- Queue activity counters
- Interprocess communication counters

Note: The **sadc** command reports only local activity.

Security

Access Control: These commands should grant execute (x) access only to members of the **adm** group.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To write 10 records of one second intervals to the **/tmp/rpt** binary file, enter:

```
sadc 1 10 /tmp/rpt
```

Files

Item	Description
/var/adm/sa/sa*dd*	Contains the daily data file, *dd* represents the day of the month.
/var/adm/sa/sar*dd*	Contains the daily report file, *dd* represents the day of the month.
/tmp/rpt	Contains the binary file used for input by the **sar** command.
/tmp/sa.adrf1	Contains the address file.

sar Command

Purpose

Collects, reports, or saves system activity information.

Syntax

/usr/sbin/sar [{ **-A** [-M] | [**-a**] [**-b**] [**-c**] [**-d**][**-k**] [**-m**] [**-q**] [**-r**] [**-u**] [**-v**] [**-w**] [**-y**] [-M] }] [**-P** *processoridentifier*, ... | **ALL** | **RST [-O** *{sortcolumn=col_name[,sortorder={asc|desc}][,topcount=n}}]]*] [[**-@** *wparname*] [**-e**[YYYYYMMDD]hh [**:***mm* [**:***ss*]]] [**-f***file*] [**-i***seconds*] [**-o***file*] [**-s**[YYYYYMMDD]hh [**:**mm [**:***ss*]]]**[-x]** [*Interval* [*Number*]]

sar [**-X** [-o *filename*]] [interval[count]]

Description

The **sar** command writes to standard output the contents of selected cumulative activity counters in the operating system. The accounting system, based on the values in the *number* and *interval* parameters, writes information the specified number of times spaced at the specified intervals in seconds. The default sampling interval for the *number* parameter is 1 second. The collected data can also be saved in the file specified by the **-o** *file* flag.

The **sar** command generates an XML file when the **–X** option is specified.

The **sar** command extracts and writes to standard output records previously saved in a file. This file can be either the one specified by the **-f** flag or, by default, the standard system activity daily data file, the **/var/adm/sa/sa***dd* file, where the *dd* parameter indicates the current day.

Without the **-P** flag, the **sar** command reports system-wide (global among all processors) statistics, which are calculated as averages for values expressed as percentages, and as sums otherwise. If the **-P** flag is given, the **sar** command reports activity which relates to the specified processor or processors. If **-P ALL** is given, the **sar** command reports statistics for each individual processor, followed by system-wide statistics. If **-P ALL** is used in a workload partition environment and the WPAR is associated with an **rset** registry, the resource set statistics and the system-wide statistics are displayed; the processors that belong to the resource set are prefixed with an asterisk symbol (*).

You can select information about specific system activities using flags. If you do not specify any flags, you select only system and WPAR unit activity. Specifying the **-A** flag selects all activities. The `sar` command prints the number of processors and the number of disks that are currently active before starting to print the statistics.

The default version of the **sar** command (processor utilization report) might be one of the first facilities the user runs to begin system activity investigation, because it monitors major system resources. If processor utilization is near 100 percent (user + system), the workload sampled is processor-bound. If a considerable percentage of time is spent in I/O wait, it implies that processor execution is blocked waiting for disk I/O. The I/O may be required file accesses or it may be I/O associated with paging due to a lack of sufficient memory.

Note: The time the system spends waiting for *remote* file access is *not* accumulated in the I/O wait time. If CPU utilization and I/O wait time for a task are relatively low, and the response time is not satisfactory, consider investigating how much time is being spent waiting for remote I/O. Since no high-level command provides statistics on remote I/O wait, trace data may be useful in observing this. If there is a change in system configuration that affects the output of the `sar` command, `sar` prints the average values up to the current iteration and then a warning message about the configuration change. It then continues the output, after printing the updated system configuration information.

Methods Used to Compute CPU Disk I/O Wait Time

The AIX operating system contains enhancements to the method used to compute the percentage of processor time spent waiting on disk I/O (*wio* time). The wio time is reported by the commands **sar** (*%wio*), **vmstat** (*wa*) and **iostat** (*% iowait*).

At each clock interrupt on each processor (100 times a second per processor), a determination is made as to which of the four categories (usr/sys/wio/idle) to place the last 10 ms of time. If the processor was busy in usr mode at the time of the clock interrupt, then usr gets the clock tick added into its category. If the processor was busy in kernel mode at the time of the clock interrupt, then the sys category gets the tick. If the processor was not busy, a check is made to see if any I/O to disk is in progress. If any disk I/O is in progress, the wio category is incremented. If no disk I/O is in progress and the processor is not busy, the idle category gets the tick. The inflated view of wio time results from all idle processors being categorized as wio regardless of the number of threads waiting on I/O. For example, systems with just one thread doing I/O could report over 90 percent wio time regardless of the number of processors it has.

The AIX operating system marks an idle processor as wio if an outstanding I/O was started on that processor. This method can report much lower wio times when just a few threads are doing I/O and the system is otherwise idle. For example, a system with four processors and one thread doing I/O will report a maximum of 25 percent wio time. A system with 12 processors and one thread doing I/O will report

a maximum of 8 percent wio time. NFS client reads/writes go through the VMM, and the time that biods spend in the VMM waiting for an I/O to complete is now reported as I/O wait time.

If multiple samples and multiple reports are desired, it is convenient to specify an output file for the **sar** command. Direct the standard output data from the **sar** command to `/dev/null` and run the **sar** command as a background process. The syntax for this is:

```
sar -A -o data.file interval count > /dev/null &
```

All data is captured in binary form and saved to a file (`data.file`). The data can then be selectively displayed with the **sar** command using the **-f** option.

The **sar** command calls a process named **sadc** to access system data. Two shell scripts (`/usr/lib/sa/sa1` and `/usr/lib/sa/sa2`) are structured to be run by the **cron** command and provide daily statistics and reports. Sample stanzas are included (but commented out) in the **/var/spool/cron/crontabs/adm crontab** file to specify when the **cron** daemon should run the shell scripts. Collection of data in this manner is useful to characterize system usage over a period of time and determine peak usage hours.

You can insert a dummy record into the standard system activity daily data file at the time of system start by un-commenting corresponding lines in the **/etc/rc** script. The `sar` command reports `time change not positive` for any record where processor times are less than the previous record. This occurs if you reboot the system with the dummy record insertion lines in **/etc/rc** commented out.

Beginning with AIX 5.3, the `sar` command reports utilization metrics `physc` and `%entc` which are related to Micro-Partitioning and simultaneous multithreading environments. These metrics will only be displayed on Micro-Partitioning and simultaneous multithreading environments. `physc` indicates the number of physical processors consumed by the partition (in case of system wide utilization) or logical processor (if the `-P` flag is specified) and `%entc` indicates the percentage of the allocated entitled capacity (in case of system wide utilization) or granted entitled capacity (if the `-P` flag is specified). When the partition runs in capped mode, the partition cannot get more capacity than it is allocated. In uncapped mode, the partition can get more capacity than it is actually allocated. This is called granted entitled capacity. If the `-P` flag is specified and there is unused capacity, `sar` prints the unused capacity as separate processor with `cpu id U`.

Beginning with AIX 6.1, the **sar** command reports the utilization metric `%resc`, which is related to the workload partition (WPAR) environment. The `%resc` metric indicates the percentage of processor resource that the WPAR consumes. This field is displayed only if the processor-resource limit is enforced in the WPAR. The **sar -P** command reports the resource set (RSET) utilization metrics R for the WPAR.

Restriction: The **sar** command only reports on local activities.

You could also use the System Management Interface Tool (SMIT) **smit sar** fast path to run this command.

Flags

Item	Description
-@ *wparname*	The **-@** flag specifies that the command reports the processor use in WPAR from the global environment. The *wparname* parameter specifies which WPAR processor statistics are to be reported. **Note:** The **-@** flag is not supported when executed within a workload partition. **Note:** Do not use the **-@** flag with the **-d**, **-r**, **-y**, **-f**, or **-X** flags.

Item	Description
-A	Without the **-P** flag, using the **-A** flag is equivalent to specifying **-abcdkmqruvwy**. When used with the **-P** flag, the **-A** is equivalent to specifying **-acmuw**. Without the `-M` flag, headers are only printed once in multiple lines grouped together before the data for the first interval. When this flag is used with the `-M` flag, each line of data at each iteration is preceded by the appropriate header.
-a	Reports use of file access system routines specifying how many times per second several of the system file access routines have been called. When used with the **-P** flag, the information is provided for each specified processor; otherwise, it is provided only system-wide. The following values are displayed: **dirblk/s** Number of 512-byte blocks read by the directory search routine to locate a directory entry for a specific file. **iget/s** Calls to any of several i-node lookup routines that support multiple file system types. The **iget** routines return a pointer to the i-node structure of a file or device. **lookuppn/s** Calls to the directory search routine that finds the address of a v-node given a path name.
-b	Reports buffer activity for transfers, accesses, and cache (kernel block buffer cache) hit ratios per second. Access to most files in Version 3 bypasses kernel block buffering and therefore does not generate these statistics. However, if a program opens a block device or a raw character device for I/O, traditional access mechanisms are used making the generated statistics meaningful. The following values are displayed: **bread/s, bwrit/s** Reports the number of block I/O operations. These I/Os are generally performed by the kernel to manage the block buffer cache area, as discussed in the description of the **lread/s** value. **lread/s, lwrit/s** Reports the number of logical I/O requests. When a logical read or write to a block device is performed, a logical transfer size of less than a full block size may be requested. The system accesses the physical device units of complete blocks and buffers these blocks in the kernel buffers that have been set aside for this purpose (the block I/O cache area). This cache area is managed by the kernel, so that multiple logical reads and writes to the block device can access previously buffered data from the cache and require no real I/O to the device. Application read and write requests to the block device are reported statistically as logical reads and writes. The block I/O performed by the kernel to the block device in management of the cache area is reported as block reads and block writes. **pread/s, pwrit/s** Reports the number of I/O operations on raw devices. Requested I/O to raw character devices is not buffered as it is for block devices. The I/O is performed to the device directly. **%rcache, %wcache** Reports caching effectiveness (cache hit percentage). This percentage is calculated as: [(100)x(lreads - breads)/ (lreads)].

Item	Description
-c	Reports system calls. When used with the **-P** flag, the information is provided for each specified processor; otherwise, it is provided only system-wide. The following values are displayed: **exec/s, fork/s** Reports the total number of **fork** and **exec** system calls. **sread/s, swrit/s** Reports the total number of read/write system calls. **rchar/s, wchar/s** Reports the total number of characters transferred by read/write system calls. **scall/s** Reports the total number of system calls. **Tip:** The **sar** command itself can generate a considerable number of reads and writes depending on the interval at which it is run. Run the **sar** statistics without the workload to understand the **sar** command's contribution to your total statistics.
-d	Reports activity for each block device with the exception of tape drives. The following data is reported: **%busy** Reports the portion of time the device was busy servicing a transfer request. **avque** Reports the average number of requests waiting to be sent to disk. **read/s, write/s, blk/s** Reports the read-write transfers from or to a device in kilobytes/second. **avwait, avserv** Average wait time and service time per request in milliseconds. **Restriction:** The **-d** flag is restricted in workload partitions.
-e[YYYYMMDD] hh[:mm[:ss]]	Sets the ending time of the report. The default ending time is 18:00. • If you specify the year, month, and date in the YYYYMMDD format, then the **-x** flag is turned on implicitly. • If you do not specify the year, month, and date in the YYYYMMDD format, then the year, month, and date are considered to be that of the first record in the activity data file that matches the specified time
-f *file*	Extracts records from the *file* (created by **-o** *file* flag). The default value of the *file* parameter is the current daily data file, the **/var/adm/sa/sa***dd* file. **Restriction:** If you specify the [*interval* [*number*]] parameter, the **-f** flag is ignored. The **-f** flag is restricted in workload partitions.
-i *seconds*	Selects data records at seconds as close as possible to the number specified by the *Seconds* parameter. Otherwise, the **sar** command reports all seconds found in the data file.

Item	Description
-k	Reports kernel process activity. The following values are displayed: **kexit/s** Reports the number of kernel processes terminating per second. **kproc-ov/s** Reports the number of times kernel processes could not be created because of enforcement of process threshold limit. **ksched/s** Reports the number of kernel processes assigned to tasks per second.
-M	Enables multiple headers in output when used with at least two combinations of [abckmqruvwy] or with the -A flag. In this mode, each line of data is preceded by the corresponding header at each iteration. **Restriction:** This flag is ignored when used without [*interval* [*number*]].
-m	Reports message (sending and receiving) and semaphore (creating, using, or destroying) activities per second. When used with the **-P** flag, the information is provided for each specified processor; otherwise, it is provided only system-wide. The following values are displayed: **msg/s** Reports the number of IPC message primitives. **sema/s** Reports the number of IPC semaphore primitives.
-o *file*	Saves the readings in the file in binary form. Each reading is in a separate record and each record contains a tag identifying the time of the reading.
-P *processoridentifier***, ...** | **ALL** |**RST**	Reports per-processor statistics for the specified processor or processors. Specifying the **ALL** keyword reports statistics for each individual processor, and globally for all processors. Specifying the **RST** option reports statistics for the processors present in the **rset** registry that is associated with the WPAR. Of the flags that specify the statistics to be reported, only the **-a**, **-c**, **-m**, **-u**, and **-w** flags are meaningful with the **-P** flag in the global environment. In the WPAR environment, do not use any flag with the **-P** flag. **Note:** The statistics for each processor that the **sar** command reports for WPAR are always system-wide.
-q	Reports queue statistics. The following values are displayed: **runq-sz** Reports the average number of kernel threads in the run queue. **%runocc** Reports the percentage of the time the run queue is occupied. **swpq-sz** Reports the average number of kernel threads that are waiting in the virtual memory manager queue for resource, input, or output. **%swpocc** Reports the percentage of the time the swap queue is occupied. **Tip:** A blank value in any column indicates that the associated queue is empty.

Item	Description
-r	Reports paging statistics. The following values are displayed: **cycle/s** Reports the number of page replacement cycles per second. **fault/s** Reports the number of page faults per second. This is not a count of page faults that generate I/O, because some page faults can be resolved without I/O. **slots** Reports the number of free pages on the paging spaces. **odio/s** Reports the number of non paging disk I/Os per second. **Restriction:** The **-r** flag is restricted in workload partitions.
-s[YYYYMMDD] hh[:mm[:ss]]	Sets the starting time of the data, causing the **sar** command to extract records time-tagged at, or following, the time specified. The default starting time is 08:00. • If you specify the year, month, and date in the YYYYMMDD format, then the **-x** flag is turned on implicitly. • If you did not specify the year, month, and date in the YYYYMMDD format, then the year, month, and date are considered to be that of the first record in the activity data file that matches the specified time.

Item	Description
-u	Reports per processor or system-wide statistics. When used with the **-P** flag, the information is provided for each specified processor; otherwise, it is provided only system-wide. Because the **-u**flag information is expressed as percentages, the system-wide information is simply the average of each individual processor's statistics. Also, the I/O wait state is defined system-wide and not per processor. The following values are displayed: **%idle** Reports the percentage of time the processor or processors were idle with no outstanding disk I/O requests. **%sys** Reports the percentage of time the processor or processors spent in execution at the system (or kernel) level. **%usr** Reports the percentage of time the processor or processors spent in execution at the user (or application) level. **%wio** Reports the percentage of time the processor(s) were idle during which the system had outstanding disk/NFS I/O request(s). See detailed description above. **physc** Reports the number of physical processors consumed. This data will be reported if the partition is dedicated and enabled for donation, or is running with shared processors or simultaneous multithreading enabled. **%entc** Reports the percentage of entitled capacity consumed. This will be reported only if the partition is running with shared processors. Because the time base over which this data is computed can vary, the entitled capacity percentage can sometimes exceed 100%. This excess is noticeable only with small sampling intervals. **%resc** Reports the percentage of processor resource consumed. This metric is applicable only for the WPAR environment. It is reported only if the WPAR enforces processor-resource limit. **Tips:** • The **sar** command reports system unit activity if no other specific content options are requested. If the `-P` flag is used and the partition is running with shared processors, and if the partition capacity usage is what is allocated, then a processor row with `cpuid U` will be reported to show the system-wide unused capacity. If the partition is running with shared processors in uncapped mode, then `%entc` will report the percentage of granted entitled capacity against each processor row and percentage of allocated entitled capacity in the system-wide processor row. The individual processor utilization statistics is calculated against the actual physical consumption (physc). The system wide statistics is computed against the entitlement and not physical consumption. However, in an uncapped partition, the system wide statistics is still calculated against the physical consumption. • Since the time base over which the data is computed varies, the sum of all of the **%utilization** fields (**%user**, **%sys**, **%idle**, and **%wait**) can exceed 100 percent.

Item	Description
-v	Reports status of the process, kernel-thread, i-node, and file tables. The following values are displayed: **file-sz, inod-sz, proc-sz , thrd-sz** Reports the number of entries in use for each table.
-w	Reports system switching activity. When used with the **-P** flag, the information is provided for each specified processor; otherwise, it is provided only system-wide. The following value is displayed: **pswch/s** Reports the number of context switches per second.
-y	Reports tty device activity per second. **canch/s** Reports tty canonical input queue characters. This field is always 0 (zero). **mdmin/s** Reports tty modem interrupts. **outch/s** Reports tty output queue characters. **rawch/s** Reports tty input queue characters. **revin/s** Reports tty receive interrupts. **xmtin/s** Reports tty transmit interrupts. **Restriction:** The **-y** flag is restricted in workload partitions.
-x	Displays the date and time for each entry. The **-x** flag is turned on implicitly whenever the user specifies the data in the YYYYMMDD format for the **-s** flag or the **-e** flag.
-O*Options*	Allows users to specify the command option. **-O options=value...** Following are the supported options: • **sortcolumn** = Name of the metrics in the **sar** command output • **sortorder** = [asc|desc] • **topcount** = Number of CPUs to be displayed in the **sar** command sorted output
-X	Generates the XML output. The default file name is **sar_DDMMYYHHMM.xml** unless the user specifies a different file name using with the **–o** option.
-o	Specifies the file name for the XML output.

Security

Access Control: These commands should grant execute (x) access only to members of the **adm** group.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To report system unit activity, enter the following command:

```
sar
```

2. To report current tty activity for each 2 seconds for the next 40 seconds, enter the following command:

```
sar -y -r 2 20
```

3. To watch system unit for 10 minutes and sort data, enter the following command:

```
sar -o temp 60 10
```

4. To report processor activity for the first two processors, enter the following command:

```
sar -u -P 0,1
```

This produces output similar to the following:

```
cpu  %usr  %sys  %wio  %idle
0      45    45     5      5
1      27    65     3      5
```

5. To report message, semaphore, and processor activity for all processors and system-wide, enter the following command:

```
sar -mu -P ALL
```

On a four-processor system, this produces output similar to the following (the last line indicates system-wide statistics for all processors) :

```
cpu  msgs/s  sema/s  %usr  %sys  %wio  %idle
0      7       2       45    45     5     5
1      5       0       27    65     3     5
2      3       0       55    40     1     4
3      4       1       48    41     4     7
-     19       3       44    48     3     5
```

6. To see physical processor consumed and entitlement consumed for all processors system-wide, run `sar` command in a shared processor logical partition machine, as follows:

```
sar –P ALL
```

On a two-logical processor system, this produces output similar to the following (the last line indicates system-wide statistics for all processors, and the line with `cpuid U` indicates the system-wide Unused capacity):

```
cpu    %usr    %sys    %wio   %idle   physc   %entc
 0        0       0       0     100    0.02     3.1
 1        0       0       0     100    0.00     1.0
 U        -       -       0      96    0.48    96.0
 -        0       0       0     100    0.02     4.0
```

7. To report system call, kernel process, and paging activities with separate headers for each of the three lines of data at each iteration for every 2 seconds for the next 40 seconds, enter the following command:

```
sar -Mckr 2 20
```

8. To report all activities with multiple sets of headers for every 2 seconds for the next 40 seconds, enter the following command:

```
sar -MA 2 20
```

9. To report the processor use statistics in a WPAR from the global environment, enter the following command:

```
sar -@ wparname
```

10. To report the processor activities for all of the processors present in the **rset** registry associated with the WPAR from inside a WPAR, enter the following command:

```
sar -P RST 1 1
```

In a WPAR that is associated with an RSET of two logical processors, the previous command generates a report similar to the following:

```
19:34:39 cpu    %usr    %sys    %wio   %idle   physc
19:34:40  0       0       2       0      98    0.54
          1       0       0       0     100    0.46
          R       0       1       0      99    1.00
```

11. To report all of the processor activities from inside a WPAR, enter the following command:

```
sar -P ALL 1 1
```

In a WPAR that is associated with an RSET of two logical processors, the previous command generates a report similar to the following:

```
19:34:39 cpu    %usr    %sys    %wio   %idle   physc
19:34:40 *0       0       2       0      98    0.54
         *1       0       0       0     100    0.46
          R       0       1       0      99    1.00
          -       0       1       0      99    1.00
```

12. To display the sorted output for the column **cswch/s** with the **–w** flag, enter the following command:

```
sar -w -P ALL  -O sortcolumn=cswch/s 1 1
```

13. To list the top ten CPUs, sorted on the **scall/s** column, enter the following command:

```
sar -c -O sortcolumn=scall/s,sortorder=desc,topcount=10 -P ALL 1
```

Files

Item	Description
/usr/sbin/sar	Contains the **sar** command.
/bin/sar	Indicates the symbolic link to the **sar** command.
/var/adm/sa/sa*dd*	Indicates the daily data file, where the *dd* parameter is a number representing the day of the month.

savebase Command

Purpose

Saves information about base-customized devices in the Device Configuration database onto the boot device.

Syntax

savebase [**-o** *Path*] [**-d** *File*] [**-v**]

Description

The **savebase** command stores customized information for base devices for use during phase 1 of system boot. By default, the **savebase** command retrieves this information from the **/etc/objrepos** directory.

However, you can override this action by using the **-o** flag to specify an ODM directory. The `savebase` command is typically run without any parameters. It uses the `/dev/ipl_blv` special file link to identify the output destination.

Alternatively, use the **-d** flag to specify a destination file or a device, such as the **/dev/hdisk0** device file. To identify a specific output destination, the `-d` flag identifies the file to which `savebase` writes the base customized device data. This file can be either a regular file or a device special file. The device special file identifies either a disk device special file or a boot logical volume device special file.

A disk device special file can be used where there is only one boot logical volume on the disk. The `savebase` command ensures that the given disk has only one boot logical volume present and is bootable. If neither of these conditions is true, `savebase` does not save the base customized device data to the disk and exits with an error.

When a second boot logical volume is on a disk, the boot logical volume device special file must be used as the destination device to identify which boot image the base customized device data will be stored in. A boot logical volume device special file can be used even if there is only one boot logical volume on the disk. The `savebase` command ensures that the given device special file is a boot logical volume and it is bootable before saving any data to it. If either of these checks fails, `savebase` exits with an error.

Note: The `-m` flag is no longer used by the `savebase` command. For compatibility reasons, the flag can be specified, but `savebase` effectively ignores it.

Flags

Item	Description
-d *File*	Specifies the destination file or device to which the base information will be written.
-o *Path*	Specifies a directory containing the Device Configuration database.
-v	Causes verbose output to be written to standard output.

Examples

1. To save the base customized information and see verbose output, enter:

   ```
   savebase -v
   ```

2. To specify an ODM directory other than the **/usr/lib/objrepos** directory, enter:

   ```
   savebase -o /tmp/objrepos
   ```

3. To save the base customized information to the **/dev/hdisk0** device file instead of to the boot disk, enter:

   ```
   savebase -d /dev/hdisk0
   ```

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Files

Item	Description
/usr/include/sys/cfgdb.h	Defines the type of boot mask for base devices.

Item	Description
/usr/lib/objrepos/PdDv	Contains entries for all known device types supported by the system.
/etc/objrepos/CuDv	Contains entries for all device instances defined in the system.
/etc/objrepos/CuAt	Contains customized device-specific attribute information.
/etc/objrepos/CuDep	Describes device instances that depend on other device instances.
/etc/objrepos/CuDvDr	Stores information about critical resources that need concurrency management through the use of the Device Configuration Library routines.

savecore Command

Purpose

Saves a system dump.

Syntax

savecore { [[**-c**] [**-d**] [**-f**]] | [**-F** [**-d**]] } *DirectoryName SystemName*

Description

The function of the **savecore** command is to save a system dump and is usually run at system startup.

The **savecore** command checks to see that you have a recent dump and that there is enough space to save it. The system dump is saved in the *DirectoryName*/**vmcore.***n* file, and the system is saved in the *DirectoryName*/**vmunix.***n* file. The *n* variable is specified in the *DirectoryName*/**bounds** file. If this file does not exist, it is created with a default of **0**, and the *n* variable uses this value. With each subsequent dump, the *n* variable is increased by 1.

The compressed dump is copied to a file named *DirectoryName*/**vmcore**. *n*.**Z**, where **.Z** is the standard indication that a file is compressed.

If the system dump was from a system other than */unix*, the name of the system must be supplied as *SystemName*.

Note: The **savecore** command saves only the current dump and the dump prior to the current one.

The directory may contain a file named **minfree**. This file contains the number of kbytes to leave free in the directory. The **minfree** file can be used to ensure a minimum amount of free space is left after the dump is copied.

Flags

Item	Description
-c	Marks the dump invalid (not recent), but does not copy it.
-d	Copies only the dump. It does not copy the system.
-f	Copies the dump even if it appears to be invalid.
-F	Reports the amount of space available for a dump in the copy directory. This may be more than the free space since the **savecore** command keeps the current dump and the previous dump, deleting others. No copying is done if the **-F** flag is specified. This flag is only valid with the **-d** flag.

Security

The Role Based Access Control (RBAC) Environment and Trusted AIX: This command implements and can perform privileged operations. Only privileged users can execute such privileged operations.

To review the list of privileges and the authorizations associated with this command, refer to the **/etc/security/privcmds** database.

Examples

1. To copy the dump (not the system) to *DirectoryName*, enter:

```
savecore -d DirectoryName
```

2. To copy the dump even if it is invalid, enter:

```
savecore -f -d DirectoryName
```

3. To mark the dump invalid, enter:

```
savecore -c
```

4. To copy the dump and the system, enter:

```
savecore -d DirectoryName SystemName
```

5. To see how much space is available for a dump, enter:

```
savecore -d -F DirectoryName
```

savevg Command

Purpose

Finds and backs up all files belonging to a specified volume group.

Syntax

savevg [**-a**] [**-A**] [**-b** *Blocks*] [**-e**] [**-f** *Device*] [**-i** | **-m**] [**-p**] [**-r**] [**-T**] [**-v**] [**-V**] [**-x** *file*] [**-X**] *VGName* [-Z]

Description

The **savevg** command finds and backs up all files belonging to a specified volume group. The volume group must be varied-on, and the file systems must be mounted. The **savevg** command uses the data file created by the **mkvgdata** command. This data file can be one of the following:

/image.data
Contains information about the root volume group (**rootvg**). The **savevg** command uses this file to create a backup image that can be used by Network Installation Management (NIM) to reinstall the volume group to the current system or to a new system.

/tmp/vgdata/*vgname*/*vgname*.data
Contains information about a user volume group. The *VGName* variable reflects the name of the volume group. The **savevg** command uses this file to create a backup image that can be used by the **restvg** command to remake the user volume group.

To create a backup of the operating system to CD, use the **mkcd** command.

Note: The **savevg** command will not generate a bootable tape if the volume group is the root volume group. Although the tape is not bootable, the first three images on the tape are dummy replacements for the images normally found on a bootable tape. The actual system backup is the fourth image.

Flags

Item	Description
-a	Does not back up extended attributes or NFS4 ACLs.
-A	Backs up DMAPI file system files.
-b *Blocks*	Specifies the number of 512-byte blocks to write in a single output operation. If this parameter is not specified, the **backup** command uses a default value appropriate for the physical device selected. Larger values result in larger physical transfers to tape devices. The value specified must be a multiple of the physical block size of the device being used.
-e	Excludes files specified in the **/etc/exclude.***vgname* file from being backed up by this command. **Note:** If you want to exclude certain files from the backup, create the **/etc/exclude.rootvg** file, with an ASCII editor, and enter the patterns of file names that you do not want included in your system backup image. The patterns in this file are input to the pattern matching conventions of the **grep** command to determine which files will be excluded from the backup. If you want to exclude files listed in the **/etc/exclude.rootvg** file, select the Exclude Files field and press the Tab key once to change the default value to yes. For example, to exclude all the contents of the directory called scratch, edit the exclude file to read as follows: `/scratch/` For example, to exclude the contents of the directory called **/tmp**, and avoid excluding any other directories that have **/tmp** in the pathname, edit the exclude file to read as follows: `^./tmp/` All files are backed up relative to . (current working directory). To exclude any file or directory for which it is important to have the search match the string at the beginning of the line, use ^ (caret character) as the first character in the search string, followed by **.** (dot character), followed by the filename or directory to be excluded. If the filename or directory being excluded is a substring of another filename or directory, use **^.** (caret character followed by dot character) to indicate that the search should begin at the beginning of the line and/or use **$** (dollar sign character) to indicate that the search should end at the end of the line.
-f *Device*	Specifies the device or file name on which the image is to be stored. The default is the **/dev/rmt0** device.
-i	Creates the data file by calling the **mkvgdata** command.
-m	Creates the data file with map files by calling the **mkvgdata** command with the **-m** flag.
-p	Disables software packing of the files as they are backed up. Some tape drives use their own packing or compression algorithms.
-r	Backs up user volume group information and administration data files. This backs up files such as **/tmp/vgdata/vgname/vgname.data** and map files if any exist. This does not back up user data files. This backup can be used to create a user volume group without restoring user data files. This cannot be done to rootvg.

Item	Description
-T	Create a backup using snapshots. This flag applies only for JFS2 file systems. When you specify the **-T** flag to use snapshots for creating a volume group backup, external JFS2 snapshots are created. Snapshots allow for a point-in-time image of a JFS2 file system and thus, do not require a system to be put into a temporarily inactive state. The size of the snapshot is 2 - 15% of the size of the file system. The snapshot logical volumes are removed when backup is finished. However, snapshots are not removed if a file system already has other snapshots. Additionally, if a file system has internal snapshots, then external snapshots cannot be created and snapshots are not used for creating the backup of the file system. The use of the **-T** flag does not affect any JFS file systems that are present in the volume group that is being backed up, These file systems are backed up in the same manner as done previously.
-v	Verbose mode. Lists files as they are backed up.
-V	Verifies a tape backup. This flag causes **savevg** to verify the file header of each file on the backup tape and report any read errors as they occur.
-x *file*	Exclude the file systems listed in the file from the volume group backup. One file system mount point is listed per line.
-X	Specifies to automatically expand the **/tmp** file system if necessary. The **/tmp** file system may need to be extended to make room for the boot image when creating a bootable backup to tape.
-Z	Specifies that the Encrypted File System (EFS) information for all the files, directories, and file systems is not backed up. The flag runs the **backup** command without the **-Z** flag.

Parameters

Item	Description
VGName	Specifies the name of the volume group to be backed up.

SMIT Fast Paths

1. To list the contents of a root volume group backup that is created with the **savevg** command, enter the following SMIT fast path:

   ```
   smit lsmksysb
   ```

2. To list the contents of a user volume group backup that is created with the **savevg** command, enter the following SMIT fast path:

   ```
   smit lsbackvg
   ```

3. To restore individual files from a root volume group backup, enter the following SMIT fast path:

   ```
   smit restmksysb
   ```

4. To restore individual files from a user volume group backup, enter the following SMIT fast path:

   ```
   smit restsavevg
   ```

Examples

1. To back up the root volume group (operating system image) to the **/mysys/myvg/myroot** backup file and create an **/image.data** file, enter:

```
savevg -i -f/mysys/myvg/myroot rootvg
```

2. To back up the **uservg** volume group to the default tape drive (**dev/rmt0**) and create a new **uservg.data** file, enter:

```
savevg -i uservg
```

3. To back up the **data2** volume group and create map files along with a new **data2.data** file on **rmt1** device, enter:

```
savevg -mf/dev/rmt1 data2
```

4. To back up the **data2** volume group, excluding the files listed in the **/etc/exclude.data2** file, enter:

```
savevg -ief/dev/rmt1 data2
```

5. To back up the volume group **my_vg** to the tape in **/dev/rmt0** and then verify the readability of file headers, enter:

```
savevg -f /dev/rmt0 -V my_vg
```

6. To back up the **uservg** volume group to the UDFS capable device**/dev/usbms0**, enter the following command:

```
savevg –i –f /dev/usbms0
```

Files

Item	Description
/image.data	Used when the volume group is **rootvg**.
/tmp/vgdata/*vgname* **/***vgname***.data**	Used when the volume group is not **rootvg** and where *vgname* is the name of the volume group.

savewpar Command

Purpose

Finds and backs up all files belonging to a specified workload partition.

Syntax

savewpar [**-a**] [**-A**] [**-B**] [**-b** *Blocks*] [**-e**] [**-f** *Device*] [**-i** | **-m**] [**-N**] [**-p**] [**-T**][**-v**] [**-V**] [**-X**] [**-Z**] [**-P**] *WparName*

Description

The **savewpar** command finds and backs up all files belonging to a specified workload partition (WPAR). The **savewpar** command uses the data file created by the **mkwpardata** command. This data file is located in the following directory, using the form:

```
/tmp/wpardata/WparName/image.data
```

The *WparName* variable reflects the name of the WPAR. The **savewpar** command uses this file to create a backup image that can be used by the **restwpar** command to re-create a workload partition. For more information, see the **restwpar** command.

To back up customized (not including *rootvg*) volume groups, see the **savevg** command.

Restriction:

- You cannot use the `savewpar` command to create a bootable tape. For best performance, properly end applications that open and close files frequently before you run the `savewpar` command.
- You must not run the `savewpar` command during an AIX live kernel update operation.

You cannot use the **savewpar** command to create a bootable tape. For best performance, properly end applications that open and close files frequently before you run the **savewpar** command.

Flags

Item	Description
-a	Does not backup extended attributes or NFS version 4 (NFS4) access control lists (ACLs).
-A	Backs up the data management application programming interface (DMAPI) file system files.
-B	Does not backup the files residing in the writable *namefs-mounted* file systems. The default is to include files from the writable *namefs-mounted* file systems in the backup.
-b *Blocks*	Specifies the number of 512-byte blocks to write in a single output operation. If you do not specify this parameter, the **backup** command uses a default value for the physical device that you selected. Larger values result in larger physical transfers to tape devices. The value that you specified must be a multiple of the physical block size of the device being used.

Item	Description
-e	Excludes files specified in the **/etc/exclude.***WparName* file from being backed up by this command. **Tip:** If you want to exclude certain files from the backup, create the **/etc/exclude.***WparName* file, with an ASCII editor, and enter the patterns of file names that you do not want to be included in the WPAR backup image. The patterns in this file are input to the pattern-matching conventions of the **grep** command to determine which files is to be excluded from the backup. All of the files are backed up relatively from the base directory (marked with the dot character ".") of the WPAR. To exclude any file or directory for which it is important to have the search match the string at the beginning of the line, use the caret character (^) as the first character in the search string, followed by the dot character (.), and the file name or directory to be excluded. For example, to exclude all of the contents of the `/tmp` directory, and avoid excluding any other directories that have the `/tmp` in the path name, edit the exclude file to read as follows: `^./tmp/` If the file name or the directory being excluded is a substring of another file name or directory, use the caret character (^) followed by the dot character (.) to indicate that the search begins at the beginning of the line, or use the dollar sign ($) to indicate that the search ends at the end of the line.
-f *Device*	Specifies the device or the file name that the image is to be stored on. The default value is the **/dev/rmt0** device.
-i	Creates the data file by calling the **mkwpardata** command.
-m	Creates the data file with map files by calling the **mkwpardata** command with the **-m** flag.
-N	Backs up files from writable NFS-mounted file systems in the mount group for the workload partition. By default, the command does not back up files from writable NFS-mounted file systems. **Requirement:** For NFS4-mounted file systems, the local and remote system must belong to the same security domain to properly establish ownership of the files on the remote server. If this is not the case, do not use the **-N** flag.
-p	Disables software packing of the files when they are backed up. Some tape drives use their own packing or compression algorithms.

Item	Description
-T	Create a backup by using snapshots. This flag applies only for JFS2 file systems. When you specify the **-T** flag to use snapshots for creating a backup for the workload partition, external JFS2 snapshots are created. Snapshots allow for a point-in-time image of a JFS2 file system and thus, do not require a system to be set in a temporarily inactive state. The size of the snapshot is 2% - 15% of the size of the file system. The snapshot logical volumes are removed when backup operation is complete. However, snapshots are not removed if a file system already has other snapshots. Additionally, if a file system has internal snapshots, external snapshots cannot be created and snapshots are not used for creating the backup of the file system. The use of the **-T** flag does not affect any JFS file systems that are present in the volume group that is being backed up.
-v	Specifies the verbose mode. Lists files when they are backed up.
-V	Verifies a tape backup. With the **-V** flag, the **savewpar** command verifies each file header on the backup tape and reports any reading errors when they occur.
-X	Specifies that the **/tmp** file system must be automatically expanded if necessary. **Requirement:** The **-X** flag is only applicable with the **-i** or **-m** flag, if necessary. **Note:** This file system expansion is not used to expand the device file system, where the backup image is saved even if device file system is the same **/tmp** file system.
-Z	Specifies that the Encrypted File System (EFS) information for all the files, directories, and file systems is not backed up. The flag runs the **backup** command with the **-Z** flag.
-P	Exclude files from the packing option listed in the `/etc/exclude_packing` directory.

Parameters

Item	Description
WparName	Specifies the name of the workload partition to be backed up.

Examples

1. To back up the `userwpar` workload partition to the default tape drive (**dev/rmt0**) and create a new `/tmp/wpardata/userwpar/image.data` file, enter the following command:

```
savewpar -i userwpar
```

2. To back up the `wpar2` workload partition and create map files along with a new `/tmp/wpardata/wpar2/image.data` file on the **rmt1** device, enter the following command:

```
savewpar -mf/dev/rmt1 wpar2
```

3. To back up the wpar2 workload partition, exclude the files listed in the /etc/exclude.wpar2 file, enter the following command:

```
savewpar -ief/dev/rmt1 wpar2
```

4. To back up the my_wpar workload partition to the tape in tape drive **/dev/rmt0** and then verify the readability of the file headers, enter the following command:

```
savewpar -f /dev/rmt0 -V my_wpar
```

5. To exclude all of the contents of the scratch directory, edit the exclude file to read as follows:

```
/scratch/
```

6. To exclude all of the contents of the **/tmp** directory, and avoid excluding any other directories that have the **/tmp** in the path name, edit the exclude file to read as follows:

```
^./tmp/
```

7. To back up the **wpar2** workload partition and create a new /tmp/wpardata/userwpar/image.data file to the UDFS capable device **/dev/usbms0**, enter the following command:

```
savewpar -f /dev/usbms0 wpar2
```

SMIT Fast Path

1. To create a workload partition backup, enter the following SMIT fast path:

```
smit savewpar
```

2. To list the contents of a workload partition backup that was created with the **savewpar** command, enter the following SMIT fast path:

```
smit lssavewpar
```

3. To restore individual files from a workload partition backup, enter the following SMIT fast path:

```
smit restwpar
```

Files

Item	Description
/tmp/wpardata/*WparName* **/***WparName***.data**	Used where the value for the *WparName* is the name of the tworkload partition.
/etc/exclude.*WparName*	Contains the files to be excluded from backup.

scan Command

Purpose

Produces a one line per message scan listing.

Syntax

scan [**+***Folder*] [*Messages*] [**-form** *FormFile* | **-format** *String*] [**-noheader** | **-header**] [**-clear** | **-noclear**] [**-help**]

Description

The **scan** command displays a line of information about the messages in a specified folder. Each line gives the message number, date, sender, subject, and as much of the message body as possible. By default, the **scan** command displays information about all of the messages in the current folder.

If a + (plus sign) is displayed after the message number, the message is the current message in the folder. If a - (minus sign) is displayed, you have replied to the message. If an * (asterisk) is displayed after the date, the `Date:` field was not present and the displayed date is the last date the message was changed.

Flags

Item	Description
-clear	Clears the display after sending output. The **scan** command uses the values of the **$TERM** environment variable to determine how to clear the display. If standard output is not a display, the **scan** command sends a form feed character after sending the output.
+*Folder*	Specifies which folder to scan. The default is the current folder.
-form *FormFile*	Displays the **scan** command output in the alternate format described by the *FormFile* variable.
-format *String*	Displays the **scan** command output in the alternate format described by the *String* variable.
-header	Displays a heading that lists the folder name and the current date and time.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.
Messages	Displays information about each specified message in the specified folder. You can use the following references when specifying messages: ***Number*** Specifies the number of the message. ***Sequence*** Specifies a group of messages specified by the user. Recognized values include: **all** All messages in a folder. This is the default. **cur or . (period)** Current message. **first** First message in a folder. **last** Last message in a folder. **next** Message following the current message. **prev** Message preceding the current message.
-noclear	Prevents clearing of the terminal after sending output. This is the default.
-noheader	Prevents display of a heading. This is the default.

Item	Description
-width *Number*	Sets the number of columns in the **scan** command output. The default is the width of the display.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Alternate-Mailboxes:`	Specifies the mailboxes.
`Current-Folder:`	Sets the default current folder.
`Path:`	Specifies the *UserMhDirectory*.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To get a one-line list of all the messages in the current folder, enter:

   ```
   scan
   ```

   The system responds with a message similar to the following:

   ```
   3   04/17 dale@athena Status meeting <<The weekly status meeting
   5   04/20 tom@venus   Due Dates      <<Your project is due to
   6   04/21 dawn@tech   Writing Clas   <<There will be a writing
   ```

2. To get a one-line list of messages 11 through 15 in the `test` folder, enter:

   ```
   scan  +test 11-15
   ```

   The system responds with a message similar to the following:

   ```
   11  04/16 karen@anchor Meeting    <<Today's meeting is at 2 p.m.
   12  04/18 tom@venus    Luncheon   <<There will be a luncheon to
   14  04/20 dale@athena  First Draft  <<First drafts are due
   15  04/21 geo@gtwn     Examples   <<The examples will be written
   ```

Files

Item	Description
$HOME/.mh_profile	Contains the MH user profile.
/etc/mh/scan.size	Contains a sample scan format string.
/etc/mh/scan.time	Contains a sample scan format string.
/etc/mh/scan.timely	Contains a sample scan format string.
/usr/bin/scan	Contains the executable form of the **scan** command.

sccs Command

Purpose

Administration program for SCCS commands.

Syntax

sccs [**-r**] [**-d***Path*] [**-p***Path*] *Command* [*CommandFlags*] *File ...*

Description

The **sccs** command is an administration program that incorporates the set of Source Code Control System (SCCS) commands into the operating system. Additionally, the **sccs** command can be used to assign or reassign file ownership (see the **-r** flag).

The **sccs** command activates a specified *Command* having the specified flags and arguments. Each file is normally placed in a directory named SCCS and named **s.filename**. The directory SCCS is assumed to exist relative to the working directory (unless the **-p** flag is used).

Two types of commands can be used in the **sccs** command syntax sentence. The first type consists of 14 **sccs** commands that can be entered at the prompt. The second type, pseudo-commands, can be used only as part of the **sccs** command syntax. There are 12 pseudo-commands, which perform the following actions:

Item	Description
edit	Equivalent to the **get -e** command.
delget	Performs a **delta** command on the named files and then gets new versions. The new versions of the files have expanded identification keywords and are not editable. Flags: **-m, -p, -r, -s, -y** Can be passed to the **delta** command. **-b, -c, -i, -l, -s, -x** Can be passed to the **get** command.
deledit	Equivalent to the **delget** pseudo-command, except that the **get** portion of the sentence includes the **-e** flag. The **deledit** pseudo-command is useful for creating a checkpoint in your current editing session. Flags: **-m, -p, -r, -s, -y** Can be passed to the **delta** command. **-b, -c, -i, -l, -s, -x** Can be passed to the **get** command.
create	Creates an SCCS file, copying the initial contents from a file of the same name. If the file is successfully created, the original file is renamed with a comma on the front. You do not have to move or remove the original file as with the **admin** command. Flags: Accepts the same flags as the **admin** command. The **-i** flag is implied.

Item	Description
fix	Removes a named delta, but leaves a copy of the delta with changes intact. This pseudo-command is useful for fixing small compiler errors. This pseudo-command does not keep a record of changes made to the file. Flags: **-r*SID*** Indicates a required flag.
clean	Removes all files from the current directory or from the designated directory that can be recreated from SCCS files. Does not remove files that are in the process of being edited. Flags: **-b** Ignores branches when determining which files are being edited. Branches being edited in the same directory can be lost.
unedit	Equivalent to the **unget** command. Any changes made since the **get** command was used are lost.
info	Lists all files being edited. Flags: **-b** Ignores branches when determining which files are being edited. **-u [*Argument*]** Lists only the files being edited by you or the user named by the *Argument* parameter.
check	Prints all files being edited. Returns a nonzero exit status if a file is being edited. The check program can be used in a makefile to ensure that files are complete before a version is installed. Check the return code before performing the install. Flags: **-b** Ignores branches when determining which files are being edited. **-u [*Argument*]** Lists only the files being edited by you or the user named by the *Argument* parameter.
tell	Lists all files being edited, with a new line after each entry, on standard output. Flags: **-b** Ignores branches when determining which files are being edited. **-u [*Argument*]** Lists only the files being edited by you or the user named by the *Argument* parameter.

Item	Description
diffs	Shows the difference between the current version of the program you are editing and the previous deltas. Flags: **-r, -c, -i, -x, -t** Can be passed to the **get** command. **-l, -s, -e, -f, -h, -b** Can be passed to the **diff** (not **sccsdiff**) command. **-C** Can be passed to the **diff** (not **sccsdiff**) command as a **-c** flag.
print (*filename*(s))	Prints **verbose** information about the named files. If the **PROJECTDIR** environment variable is set, its value determines the working directory. If this value begins with a / (slash), it is used directly. Otherwise, the value is interpreted as a user name whose home directory is examined for a subdirectory named **src** or **source**. If found, that subdirectory is used as the working directory.

Flags

Item	Description
-d*Path*	Specifies a working directory for the SCCS files. The default is the current directory. The **-d** flag is prefixed to the entire path name of a file. When the **PROJECTDIR** environment variable is set and the **-d** flag is used, the command line overrides the environment value in determining the working directory.
-p	Specifies a path name for the SCCS files. The default is the SCCS directory. The **-p** flag is inserted before the final component of the path name. All flags specified after the command are passed to that command during execution. For a description of command flags, see the appropriate command description. Example: `sccs -d/x -py get a/b` converts to: `get /x/a/y/s.b` This option is used to create aliases. For example: `alias syssccs sccs -d/usr/src` causes the **syssccs** command to become an alias command that can be used as follows: `syssccs get cmd/who.c` When used in this context, the above command will check the **/usr/src/cmd/SCCS** directory for the **s.who.c** file.
-r	Runs the **sccs** command as the real user instead of as the effective user to which the **sccs** command is set (using the **set user id** command). Certain commands, such as the **admin** command, cannot be run as **set user id**, which would allow anyone to change the authorizations. Such commands are always run as the real user.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To get a file for editing, edit it, and then produce a new delta, enter:

```
sccs get -e file.c
ex file.c
sccs delta file.c
```

2. To get a file from another directory, enter:

```
sccs -p/usr/src/sccs/ get cc.c
```

   OR

```
sccs get /usr/src/sccs/s.cc.c
```

3. To get a list of files being edited that are not on branches, enter:

```
sccs info -b
```

Files

Item	Description
/usr/bin/sccs	Contains the **sccs** command, which is the administration program for the SCCS commands.

sccsdiff Command

Purpose

Compares two versions of a SCCS file.

Syntax

sccsdiff **-r***SID1* **-r***SID2* [**-p**] [**-s***Number*] *File* ...

Description

The **sccsdiff** command reads two versions of an Source Code Control System (SCCS) file, compares them, and then writes to standard output the differences between the two versions. Any number of SCCS files can be specified, but the same arguments apply to all files.

Flags

Item	Description
-p	Pipes the output through the **pr** command.
-r*SID1*	Specifies *SID1* as one delta of the SCCS file for the **sccsdiff** command to compare.

Item	Description
-r*SID2*	Specifies *SID2* as the other delta of the SCCS file for the **sccsdiff** command to compare.
-s*Number*	Specifies the file-segment size for the **bdiff** command to pass to the **diff** command. This is useful when the **diff** command fails due to a high system load.

Examples

To display the difference between versions 1.1 and 1.2 of SCCS file `s.test.c`, enter:

```
sccsdiff -r1.1 -r1.2 s.test.c
```

Files

Item	Description
/usr/bin/sccsdiff	Contains the SCCS **sccsdiff** command. The **sccsdiff** command supports multibyte character set (MBCS) data for the file names.

sccshelp Command

Purpose

Provides information about a SCCS message or command.

Syntax

sccshelp [*ErrorCode*] [*Command*]

Description

The **sccshelp** command displays information about the use of a specified Source Code Control System (SCCS) command or about messages generated while using the commands. Each message has an associated code, which can be supplied as part of the argument to the **sccshelp** command. Zero or more arguments may be supplied. If you do not supply an argument, the **sccshelp** command prompts for one. You may include any of the SCCS commands as arguments to the **sccshelp** command.

The *ErrorCode* parameter specifies the code, consisting of numbers and letters, that appears at the end of a message. For example, in the following message, `(cm7)` is the code:

```
There are no SCCS identification keywords in the file. (cm7)
```

Examples

To get **sccshelp** on the **rmdel** command and two error codes, enter:

```
$ sccshelp rmdel gee ad3
```

The **sccshelp** command replies:

```
rmdel:
rmdel -r<SID> <file> ...
ERROR:
1255-141 gee is not a valid parameter. Specify a valid command or error code.
ad3:
The header flag you specified is not recognized.
The header flag you supplied with the -d or the -f flag is not correct.
Choose a valid header flag.
```

File

Item	Description
/usr/bin/sccshelp	Contains the SCCS **sccshelp** command.

schedo Command

Purpose

Manages processor scheduler tunable parameters.

Syntax

schedo [**-p** | **-r**] [**-y**] { **-o** *Tunable*[**=** *Newvalue*]}

schedo [**-p** | **-r**] [**-y**] { **-d** *Tunable* }

schedo [**-p** | **-r**] [**-y**] **-D**

schedo [**-p** | **-r**] [**-F**] **-a**

schedo -h [*Tunable*]

schedo [**-F**] **-L** [*Tunable*]

schedo [**-F**] **-x** [*Tunable*]

Note: Multiple flags **-o**, **-d**, **-x**, and **-L** flags are allowed

Description

Note: The **schedo** command can only be executed by root.

Use the **schedo** command to configure scheduler tuning parameters. This command sets or displays current or next boot values for all scheduler tuning parameters. This command can also make permanent changes or defer changes until the next reboot. Whether the command sets or displays a parameter is determined by the accompanying flag. The **-o** flag performs both actions. It can either display the value of a parameter or set a new value for a parameter.

Understanding the Effect of Changing Tunable Parameters

Misuse of this command can cause performance degradation or operating-system failure. Be sure that you have studied the appropriate tuning sections in the *Performance management* before using **schedo** to change system parameters.

Before modifying any tunable parameter, you must first carefully read about all its characteristics in the Tunable Parameters section below, and follow any Refer To pointer, in order to fully understand its purpose.

You must then make sure that the Diagnosis and Tuning sections for this parameter truly apply to your situation and that changing the value of this parameter could help improve the performance of your system.

If the Diagnosis and Tuning sections both contain only "N/A", you must never change this parameter unless specifically directed by AIX development.

Priority-Calculation Parameters

The priority of most user processes varies with the amount of processor time the process has used recently. The processor scheduler's priority calculations are based on two parameters that are set with **schedo**, *sched_R* and *sched_D*. The *sched_R* and *sched_D* values are in thirty-seconds (1/32); that is,

the formula used by the scheduler to calculate the amount to be added to a process's priority value as a penalty for recent processor use is:

```
CPU penalty = (recently used CPU value of the process) * (r/32)
```

and the once-per-second recalculation of the recently used processor value of each process is:

```
new recently used CPU value = (old recently used CPU value of the process) * (d/32)
```

Both r (*sched_R* parameter) and d (*sched_D* parameter) have default values of 16. This maintains the processor scheduling behavior of previous versions of the operating system. Before experimenting with these values, you must be familiar with "Tuning the processor scheduler" in the Performance Management Guide.

Memory-Load-Control Parameters

The operating system scheduler performs memory load control by suspending processes when memory is over committed. The system does not swap out processes; instead pages are *stolen* as they are needed to fulfill the current memory requirements. Typically, pages are stolen from suspended processes. Memory is considered over committed when the following condition is met:

Item	Description
p * *h s*	where: *p* is the number of pages written to paging space in the last second *h* is an integer specified by the v_repage_hi parameter *s* is the number of page steals that have occurred in the last second

A process is suspended when memory is over committed and the following condition is met:

Item	Description
r * *p f*	where: *r* r is the number of repages that the process has accumulated in the last second *p* is an integer specified by the v_repage_proc parameter *f* is the number of page faults that the process has experienced in the last second

In addition, fixed-priority processes and kernel processes are exempt from being suspended.

The term repages refers to the number of pages belonging to the process, which were reclaimed and are soon after referenced again by the process.

The user also can specify a minimum multiprogramming level with the v_min_process parameter. Doing so ensures that a minimum number of processes remain active throughout the process-suspension period. Active processes are those that are runnable and waiting for page I/O. Processes that are waiting for events and processes that are suspended are not considered active, nor is the wait process considered active.

Suspended processes can be added back into the mix when the system has stayed below the over committed threshold for n seconds, where n is specified by the v_sec_wait parameter. Processes are added back into the system based, first, on their priority and, second, on the length of their suspension period.

Before experimenting with these values, you must be thoroughly familiar with "VMM memory load control tuning with the schedo command" in the Performance Management Guide.

Time-Slice-Increment Parameter

The **schedo** command can also be used to change the amount of time the operating system allows a given process to run before the dispatcher is called to choose another process to run (the time slice). The default value for this interval is a single clock tick (10 milliseconds). The timeslice tuning parameter allows the user to specify the number of clock ticks by which the time slice length is to be increased.

In AIX Version 4, this parameter only applies to threads with the SCHED_RR scheduling policy. See Scheduling Policy for Threads.

fork() Retry Interval Parameter

If a **fork**() subroutine call fails because there is not enough paging space available to create a new process, the system retries the call after waiting for a specified period of time. That interval is set with the pacefork tuning parameter.

Special Terminology for Symmetric Multithreading

Multiple run queues are supported. Under this scheme each processor has it's own run queue. POWER5 processors support symmetric multithreading, where each physical processor has two execution engines, called *hardware threads*. Each hardware thread is essentially equivalent to a single processor. Symmetric multithreading is enabled by default, but it can be disabled (or re-enabled) dynamically. When symmetric multithreading is enabled, each hardware thread services a separate run queue. For example, on a 4-way system when symmetric multithreading is disabled or not present, there are 4 run queues in addition to the global run queue. When symmetric multithreading is enabled, there are 8 run queues in addition to the global run queue.

The hardware threads belonging to the same physical processor are referred to as *sibling threads*. A *primary sibling thread* is the first hardware thread of the physical processor. A *secondary sibling thread* is the second hardware thread of the physical processor.

Virtual Processor Management

More virtual processors can be defined than are needed to handle the work in a partition. The overhead of dispatching virtual processors can be reduced by using fewer virtual processors without a decrease in overall processor usage or a lack of virtual processors. Virtual processors are not dynamically removed from the partition, but instead are not used and are used again only when additional work is available. Each virtual processor uses a maximum of one physical processor. The number of virtual processes needed is determined by rounding up the sum of the physical processor utilization and the **vpm_xvcpus** tunable:

```
number = ceiling( p_util + vpm_xvcpus)
```

Where *number* is the number of virtual processors that are needed, *p_util* is the physical processor utilization, and **vpm_xvcpus** is a tunable that specifies the number of additional virtual processors to enable. If *number* is less than the number of currently enabled virtual processors, a virtual processor will be disabled. If *number* is greater than the number of currently enabled virtual processors, a disabled virtual processor will be enabled. Threads that are attached to a disabled virtual processor are still allowed to run on the disabled virtual processor.

Node Load

The *node load*, or simply *load*, is the average run queue depth across all run queues, including the global run queue multiplied by 256, and is strongly smoothed over time. For example, a load of 256 means that if we have 16 processors (including symmetric multithreading processors), then we have had approximately 16 runnable jobs in the system for the last few milliseconds.

Flags

Item	Description
-a	Displays the current, reboot (when used in conjunction with **-r**) or permanent (when used in conjunction with **-p**) value for all tunable parameters, one per line in pairs *Tunable* = *Value*. For the permanent option, a value is only displayed for a parameter if its reboot and current values are equal. Otherwise NONE displays as the value.
-d *Tunable*	Resets*Tunable* to its default value. If a tunable needs to be changed (that is, it is currently not set to its default value, and **-r** is not used in combination, it won't be changed but a warning is displayed.
-D	Resets all tunables to their default value. If tunables needing to be changed are of type Bosboot or Reboot, or are of type Incremental and have been changed from their default value, and **-r** is not used in combination, they will not be changed but a warning displays.
-F	Forces the display of restricted tunable parameters when you specify the **-a**, **-L** or **-x** options on the command line, to list all of the tunables. If you do not specify the **-F** flag, restricted tunables are not included, unless they are specifically named in association with a display option.

Item	Description
-h [*Tunable*]	Displays help about the *Tunable* parameter if one is specified. Otherwise, displays the **schedo** command usage statement.
-L [*Tunable*]	Lists the characteristics of one or all tunables, one per line, using the following format: `NAME CUR DEF BOOT MIN MAX UNIT TYPE DEPENDENCIES` `v_repage_hi 0 0 0 0 2047M D` `v_repage_proc 4 4 4 0 2047M D` `v_sec_wait 1 1 1 0 2047M seconds D` `...` `where:` `CUR = current value` `DEF = default value` `BOOT = reboot value` `MIN = minimal value` `MAX = maximum value` `UNIT = tunable unit of measure` `TYPE = parameter type: D (for Dynamic), S (for Static), R (for Reboot), B (for Bosboot), M (for Mount), I (for Incremental), C (for Connect), and d (for Deprecated)` `DEPENDENCIES = list of dependent tunable parameters, one per line`
-o *Tunable* [**=***Newvalue*]	Displays the value or sets *Tunable* to *Newvalue*. If a tunable needs to be changed (the specified value is different than current value), and is of type Bosboot or Reboot, or if it is of type Incremental and its current value is bigger than the specified value, and **-r** is not used in combination, it will not be changed but a warning displays. When **-r** is used in combination without a new value, the nextboot value for tunable is displayed. When **-p** is used in combination without a new value, a value displays only if the current and next boot values for tunable are the same. Otherwise NONE displays as the value.
-p	Makes changes apply to both current and reboot values, when used in combination with **-o**, **-d** or **-D**, that is, turns on the updating of the **/etc/tunables/nextboot** file in addition to the updating of the current value. These combinations cannot be used on Reboot and Bosboot type parameters because their current value can't be changed. When used with **-a** or **-o** without specifying a new value, values are displayed only if the current and next boot values for a parameter are the same. Otherwise NONE displays as the value.
-r	Makes changes apply to reboot values when used in combination with **-o**, **-d** or **-D**, that is, turns on the updating of the **/etc/tunables/nextboot** file. If any parameter of type Bosboot is changed, the user will be prompted to run bosboot. When used with **-a** or **-o** without specifying a new value, next boot values for tunables display instead of current values.
-x [*Tunable*]	Lists characteristics of one or all tunables, one per line, using the following (spreadsheet) format: `tunable,current,default,reboot,min,max,unit,type,{dtunable }` `where:` `current = current value` `default = default value` `reboot = reboot value` `min = minimal value` `max = maximum value` `unit = tunable unit of measure` `type = parameter type: D (for Dynamic), S (for Static), R (for Reboot), B (for Bosboot),M (for Mount), I (for Incremental), C (for Connect), and d (for Deprecated)` `dtunable = space separated list of dependent tunable parameters`
-y	Suppresses the confirmation prompt before executing the **bosboot** command.

Note: Options **-o**, **-d**, and **-D** are not supported within a workload partition because they attempt to change the value of a scheduler tunable parameter.

If you make any change (with the **-o**, **-d**, or **-D** options) to a restricted tunable parameter, it results in a warning message that a tunable parameter of the restricted-use type, has been modified. If you also specified the **-r** or **-p** options on the command line, you will be prompted to confirm the change. In addition, at system reboot, restricted tunables that are displayed in the **/etc/tunables/nextboot** file, which were modified to values that are different from their default values (using a command line specifying the **-r** or **-p** options), causes an error log entry that identifies the list of these modified tunables.

When modifying a tunable, you can specify the tunable value using the abbreviations such as K, M, G, T, P and E to indicate units. See units. The following table shows the prefixes and values that are associated with the number abbreviations:

Abbreviation	Power of 2
K	1024
M	1 048 576
G	1 073 741 824
T	1 099 511 627 776
P	1 125 899 906 842 624
E	1 152 921 504 606 846 976

Thus, a tunable value of 1024 might be specified as 1K.

Any change (with **-o**, **-d** or **-D**) to a parameter of type Mount results in a message displaying to warn you that the change is only effective for future mountings.

Any change (with **-o**, **-d** or **-D** flags) to a parameter of type Connect will result in **inetd** being restarted, and in a message being displayed to warn the user that the change is only effective for future socket connections.

Any attempt to change (with **-o**, **-d** or **-D**) a parameter of type Bosboot or Reboot without **-r**, results in an error message.

Any attempt to change (with**-o**, **-d** or **-D** but without **-r**) the current value of a parameter of type Incremental with a new value smaller than the current value, results in an error message.

Tunable Parameters Type

All the tunable parameters manipulated by the tuning commands (**no**, **nfso**, **vmo**, **ioo**, `raso`, and **schedo**) have been classified into these categories:

Item	Description
Dynamic	If the parameter can be changed at any time
Static	If the parameter can never be changed
Reboot	If the parameter can only be changed during reboot
Bosboot	If the parameter can only be changed by running bosboot and rebooting the machine
Mount	If changes to the parameter are only effective for future file systems or directory mounts
Incremental	If the parameter can only be incremented, except at boot time
Connect	If changes to the parameter are only effective for future socket connections
Deprecated	If changing this parameter is no longer supported by the current release of AIX.

For parameters of type Bosboot, whenever a change is performed, the tuning commands automatically prompt the user to ask if they want to execute the **bosboot** command. For parameters of type Connect, the tuning commands automatically restart the **inetd** daemon.

Note that the current set of parameters managed by the **schedo** command only includes Dynamic, and Reboot types.

Compatibility Mode

When running in pre 5.2 compatibility mode (controlled by the **pre520tune** attribute of **sys**0, see **AIX 5.2 compatibility mode** in the *Performance management*), reboot values for parameters, except those of type Bosboot, are not really meaningful because in this mode they are not applied at boot time.

In pre 5.2 compatibility mode, setting reboot values to tuning parameters continues to be achieved by imbedding calls to tuning commands in scripts called during the boot sequence. Parameters of type **Reboot** can therefore be set without the **-r** flag, so that existing scripts continue to work.

This mode is automatically turned ON when a machine is MIGRATED to AIX 5.2. For complete installations, it is turned OFF and the reboot values for parameters are set by applying the content of the **/etc/tunables/nextboot** file during the reboot sequence. Only in that mode are the **-r** and **-p** flags fully functional. See **Kernel Tuning** in the *Performance Tools Guide and Reference* for more information.

Tunable Parameters

For default values and range of values for tunables, refer **schedo** command help (**-h** <*tunable_parameter_name*>).

Item	Description
affinity_lim	**Purpose:** Sets the number of intervening dispatches after which the SCHED_FIFO2 policy no longer favors a thread. **Tuning:** Once a thread is running with SCHED_FIFO2 policy, tuning of this variable may or may not have an effect on the performance of the thread and workload. Ideal values must be determined by trial and error.
big_tick_size	**Purpose:** Sets physical tick interval and synchronizes ticks across cpus. **Tuning:** The **big_tick_size** value times 10 ms as a tick interval, and must evenly divide into 100. Use of this parameter will make system statistics less accurate.
ded_cpu_donate_thresh	**Purpose:** Specifies the utilization threshold for donation of a dedicated processor. **Tuning:** In a dedicated processor partition that is enabled for donation, idle processor capacity can be donated to the shared processor pool for use by shared processor partitions. If a dedicated processor's utilization is less than this threshold, the dedicated processor will be donated for use by other partitions when the processor is idle. If a dedicated processor's utilization is equal to or greater than this threshold, the dedicated processor will not be donated for use by other partitions when the dedicated processor is idle.
fixed_pri_global	**Purpose:** Keep fixed priority threads on global run queue. **Tuning:** If 1, then fixed priority threads are placed on the global run queue.
force_grq	**Purpose:** Keep non-MPI threads on the global run queue. **Tuning:** If 1, only MPI and bound threads will use local run queues.
maxspin	**Purpose:** Sets the number of times to spin on a kernel lock before going to sleep. **Tuning:** Increasing the value on MP systems may reduce idle time; however, it might also waste CPU time in some situations. Increasing it on uniprocessor systems is not recommended.
pacefork	**Purpose:** The number of clock ticks to wait before retrying a failed fork call that has failed for lack of paging space. **Tuning:** Use when the system is running out of paging space and a process cannot be forked. The system will retry a failed fork five times. For example, if a **fork()** subroutine call fails because there is not enough paging space available to create a new process, the system retries the call after waiting the specified number of clock ticks.
proc_disk_stats	**Purpose:** A value of 1 enables and a value of 0 disables the process scope disk statistics. The default value is 1 and ranges from 0 to 1. **Tuning:** Disabling process scope disk statistics improves performance when the statistics are not wanted.

Item	Description
sched_D	**Purpose:** Sets the short term CPU usage decay rate. **Tuning:** The default is to decay short-term CPU usage by 1/2 (16/32) every second. Decreasing this value enables foreground processes to avoid competition with background processes for a longer time.
sched_R	**Purpose:** Sets the weighting factor for short-term CPU usage in priority calculations. **Tuning:** Run the command **ps al**. If you find that the PRI column has priority values for the foreground processes (those with NI values of 20) that are higher than the PRI values of some background processes (NI values > 20), you can reduce the r value. The default is to include 1/2 (16/32) of the short term CPU usage in the priority calculation. Decreasing this value makes it easier for foreground processes to compete.
tb_balance_S0	**Purpose:** Controls SMT-cores busy balancing. **Tuning:** A value of 0 indicates that the balancing is disabled. A value of 1 indicates that the balancing is enabled only within MCMs (S2 groups). A value of 2 indicates fully enabled.
tb_balance_S1	**Purpose:** Controls processor busy balancing. **Tuning:** A value of 0 indicates that the balancing is disabled. A value of 1 indicates that the balancing is enabled only within MCMs (S2 groups). A value of 2 indicates fully enabled.
tb_threshold	**Purpose:** Number of ticks to consider a thread busy for the purposes of optimization for **thread_busy** load balancing. **Tuning:** A value of 100 corresponds to 1 second. The values 10 and 1000 correspond to 0.1 and 10 seconds, respectively.
timeslice	**Purpose:** The number of clock ticks a thread can run before it is put back on the run queue. **Tuning:** Increasing the timeslice value can reduce overhead of dispatching threads. The value refers to the total number of clock ticks in a timeslice and only affects fixed-priority processes.
vpm_fold_policy	**Purpose:** Controls the application of the virtual processor management feature of processor folding in a partition. **Tuning:** The virtual processor management feature of processor folding can be enabled or disabled based on whether a partition has shared or dedicated processors. In addition, when the partition is in static power saving mode, processor folding is automatically enabled for both shared or dedicated processor partitions. When processor folding is enabled, the **vpm_vxcpus** tunable can be used to control processor folding. There are 3 bits in **vpm_fold_policy** to control processor folding: • Bit 0 (0x1): When set to 1, this bit indicates processor folding is enabled if the partition is using shared processors. • Bit 1 (0x2): When set to 1, this bit indicates processor folding is enabled if the partition is using dedicated processors. • Bit 2 (0x4): When set to 1, this bit disables the automatic setting of processor folding when the partition is in static power saving mode. You can perform an OR operation on the Bit 0, Bit 1, and Bit 2 values to form the desired value. **Note:** If the **Idle power saver** option is set to **Enabled** in the Advanced System Management Interface (ASMI), and if processor utilization falls below the specified threshold values, then you can set the processors to a low frequency or low voltage state. You can also set the **vpm_fold_policy** tunable parameter at runtime to 0x3.
vpm_throughput_core_threshold	Specifies the number of cores that must be unfolded before **vpm_throughput_mode** parameter is honored. Till that, the system behaves with the value of **vpm_throughput_mode** parameter set as **1**.
vpm_throughput_mode	Specifies the desired level of SMT exploitation for scaled throughput mode. A value of **0** gives default behavior (raw throughput mode). A value of **1**, **2**, or **4** selects the scaled throughput mode and the desired level of SMT exploitation.
vpm_xvcpus	**Purpose:** Setting this tunable to a value greater than -1 will enable the scheduler to enable and disable virtual processors based on the partition's CPU utilization. **Tuning:** The value specified signifies the number of virtual processors to enable in addition to the virtual processors required to satisfy the workload.

Examples

1. To list the current and reboot value, range, unit, type and dependencies of all tunables parameters managed by the schedo command, enter:

```
schedo -L
```

2. To list (spreadsheet format) the current and reboot value, range, unit, type, and dependencies of all tunables parameters managed by the **schedo** command, enter:

```
schedo -x
```

3. To reset v_sec_wait to default, enter:

```
schedo -d v_sec_wait
```

4. To display help on sched_R, enter:

```
schedo -h sched_R
```

5. To set v_min_process to 4 after the next reboot, enter:

```
schedo -r -o v_min_process=4
```

6. To permanently reset all schedo tunable parameters to default, enter:

```
schedo -p -D
```

7. To list the reboot value for all schedo parameters, enter:

```
schedo -r -a
```

scls Command

Purpose

Produces a list of module and driver names.

Syntax

scls [**-c** | **-l**] [**-m** *sc_module_name*] [*Module* ...]

Description

The **scls** command provides a method for the user to query the current **Portable Streams Environment** (PSE) configuration. The **scls** command produces a list of module and driver names. Flags can be used to produce enhanced lists. Any further parameters on the command line are module or driver names, and the output produced is for only those names.

> **Note:** The **scls** command requires the **sc** STREAMS module and the **nuls** driver. If either one is not available, the **scls** command will not be successful.

Flags

Item	Description
-c	Produces a listing showing the number of times an interface routine was called.
-l	Produces a long listing that shows the extension type, major number, and information pertaining to the **module_info** structure.
-m	Pushes the module pointed to by the *sc_module_name* to the top of the current stream, just below the stream head.

The **-c** and **-l** flags are mutually exclusive.

Parameters

Item	Description
module	Specifies the name of the modules or drivers for which to output information.
sc_module_name	Specifies a module name that needs to be pushed to the current stream, just below the stream head.

Files

Item	Description
sc	Dynamically loadable STREAMS configuration module
nuls	Dynamically loadable STREAMS null device.

script Command

Purpose

Makes a typescript of a terminal session.

Syntax

script [**-a**] [**-q**] [*File*]

Description

The **script** command makes a typescript of everything displayed on your terminal. The typescript is written to the file specified by the *File* parameter. The typescript can later be sent to the line printer. If no file name is given, the typescript is saved in the current directory with the file name **typescript**.

The script ends when the forked shell exits.

This command is useful for producing hardcopy records when hardcopy terminals are in short supply. For example, use the **script** command when you are working on a CRT display and need a hardcopy record of the dialog.

Because the **script** command sets the **SetUserID** mode bit, due to security reasons the value of LIBPATH variable is unset when the command is invoked. However, LIBPATH is automatically reset in the forked shell if it is defined in the environment file. This behavior is also true for the NLSPATH environment variable. For related information, see the **exec** subroutine.

Flags

Item	Description
-a	Appends the typescript to the specified file or to the **typescript** file.
-q	Suppresses diagnostic messages.

Files

Item	Description
/usr/bin/script	Contains the **script** command.

sctpctrl Command

Purpose

Controls and configures SCTP.

Syntax

sctpctrl {*load*|*dump*|*set*}

sctpctrl *stats* [*reset*] [*interval*]

sctpctrl *set* {*name=value*|*default* [*name*]}

sctpctrl *get* [*name*]

Description

The **sctpctrl** command is used to control and configure the SCTP kernel extension. This command can be used to load and unload the SCTP kernel extension. This can also be used to dump SCTP data and set or retrieve various SCTP tunable. Further, **sctpctrl** command can be used to read and reset the SCTP specific network statistics.

Parameters

Item	Description
load	Loads the SCTP kernel extension if not loaded.
dump	Dump information about internal SCTP structures.
stats [*reset*] [*interval*]	Displays SCTP statistics. The optional **reset** command will clear (zero) the statistics. If the *interval* parameter (in seconds) is added, the program does not exit, but outputs the statistics every [*interval*] seconds.
set {*name=value*|*default* [*name*]}	Sets the SCTP tunable to a value. If *default* is specified then it will set all the tunable to their default values. If optional [*name*] is specified followed by *default* then it will set tunable described by *name* to its default value.
get [*name*]	Gets the value of the tunable described by their optional *name* parameter. If *name* parameter is not specified then it gets the values of all the tunable.

Tunable Parameters

The **sctpctrl** command is also used to configure the SCTP tuning parameters. The changes made are not permanent and they have to be set every time a system gets rebooted. The tunables parameters are explained in the following table.

Item	Description		
Parameter	**Purpose**	**Scope**	**Default**

Item	Description		
sctp_low_rto	When nonzero, this value is used in place of *RTO.min* (retransmission time-out). It is specified in terms of milliseconds. Values less than 200 are not allowed. The available time-out values are 200, 250, 300, 350, and so on.	This value is examined each time a new RTT (round trip time) measurement is made and also when RTO is adjusted due to packet loss.	As specified in the RFC 4960 (Request for Comment) document, the default value for this tunable is zero, which means the minimum value of *RTO.Min* is used, which is 1 second.
sctp_enable_shutdown_guard	When nonzero, this tunable enables a T5-shutdown guard-timer. It is not RFC compliant because it begins timing when association enters shutdown-pending state.	This value is only examined at an association shutdown.	The default value for this tunable is zero, which means that the T5-shutdown guard-timer is not used.
sctp_max_init_attempts	This tunable parameter specifies the maximum number of retransmission attempts that are allowed for the retransmission of the INIT packets.		The default value for this tunable parameter is 8.
sctp_max_init_timeo	When the value of this tunable parameter is nonzero, the tunable value is used instead of the *RTO.max* parameter (retransmission timeout) for INIT packets. It is specified in milliseconds. Values less than 200 milliseconds are not allowed. The available timeout values are 200, 250, 300, 350, and so on.	This tunable parameter specifies the maximum value to be used when the INIT packets are retransmitted.	The default value for this tunable parameter is zero seconds. In this case, the `sctp_rttmax` tunable parameter is used.
sctp_shutdown_guard_timer	When the *sctp_enable_shutdown_guard* parameter is a nonzero value, this tunable defines the shutdown time-out value in seconds.	This value is only examined at an association shutdown.	The default value is 300 seconds, which is the RFC-specified value for the T5-shutdown guard-timer.
sctp_peerchangespath	When nonzero, this tunable causes a primary path change based on an incoming data chunk from a different path than the current primary path.	This value is examined on every inbound data chunk.	The default value for this tunable is 1, which retains the existing behavior.

Item	Description		
sctp_delack_timer	This tunable specifies the timer value in ticks (1 tick = 50 ms (milliseconds)) for the delayed-ack timer.	For an *ACCEPTCONN* socket, this value is established during setup and is used for all associations that share that socket. For a socket other than an *ACCEPTCONN* socket, it is set at association creation. So changes to this tunable do not affect associations already in existence.	The default value is 4 ticks (200 ms).
sctp_drop_gapacks	If set to 1, it causes the sender side to drop all *GAPACKED* packets from the socket send buffer, thus making some space free for new packets. **Note:** This is an RFC noncompatible tunable and could impact interoperability with other implementations, potentially resulting in a message loss.	This tunable is checked each time *GAPACKED* packets are processed.	The default value is 0, which means disabled.
sctp_dontdelayack	If set to 1, a *SACK* packet is sent for every other *DATA* packet. Otherwise, a delayed-ack timer is started.	Any updates to this tunable have an immediate impact.	The default value is 1.
sctp_nagle	If set to 1, it ensures that at least 1 MTU (maximum transmission unit) of data is sent.	Any updates to this tunable have an immediate impact.	The default value is 1 (a *nagle* is enabled).
sctp_maxburst	If nonzero, it limits the maximum number of packets sent out to this value.	Any updates to this tunable have an immediate impact.	The default value is 8 packets.

Item	Description		
sctp_rttmax	This tunable specifies the maximum value to be used when RTO computations are made.	Similar to the *sctp_low_rto* parameter, this value is examined each time a new RTT measurement is made (and RTO calculated with that) and also when RTO is adjusted due to packet loss.	The default value is 60 seconds.
sctp_rttmin	This tunable specifies the minimum value to be used when RTO computations are made.	If the *sctp_low_rto* parameter is nonzero, this value is ignored. Otherwise, it is examined each time a new RTT measurement is made and when RTO is stopped due to packet loss.	The default value is 1 second, which ensures that the minimum RTO cannot go below that.
sctp_assoc_maxerr	This tunable sets the overall association error count. If an error count exceeds this value, the association is ended. Currently, this value is ignored. The *assoc_maxerr* parameter is calculated based on the path error count and number of *faddrs*.	For an *ACCEPTCONN* socket, this value is established during setup and is used for all associations that share that socket. For a socket that is not an *ACCEPTCONN* socket, it is set at association creation. So changes to this tunable do not affect associations already in existence.	The default value is 10.

Item	Description		
sctp_path_maxerr	This tunable sets the maximum error count for each destination. If the error count exceeds this value, the path is marked down and an alternative path is chosen.	For an *ACCEPTCONN* socket, this value is established during setup and is used for all associations that share that socket. For a socket that is not an *ACCEPTCONN* socket, it is set at association creation. So changes to this tunable do not affect associations already in existence.	The default value is 5.
sctp_use_checksum	This tunable allows an administrator to use different checksum computation methods. Possible values follows: • 0: CRC32 checksum • 1: No checksum computation is made • 2: Internet checksum. The packets are dropped if different values are used by two peers.	This parameter is examined for each outgoing and incoming packet.	The default value is zero, which is the RFC-specified CRC32 checksum.
sctp_sendspace	This tunable specifies the socket buffer size for sending data. The optimum buffer size is the product of the media bandwidth and the average round-trip time of a packet: *optimum_window = bandwidth * average_round_trip_time*	This parameter is accessed when a new association is created. Use the *setsockopt* function to override this parameter.	The default value is 65536.
sctp_recvspace	This tunable specifies the socket buffer size for receiving data.	This parameter is accessed when a new association is created. Use the *setsockopt* function to override this parameter.	The default value is 65536.

Item	Description		
sctp_send_fewsacks	When enabled, this tunable parameter implements *recv side silly window avoidance*. It prevents sending a window update until a receiver can fit in 1 MTU of data.	This parameter is accessed each time data is read by an application and a window update is being sent.	The default value is 0.
sctp_cookie_life	This tunable specifies the time duration in seconds for which a cookie is considered to be valid.	This parameter is used to determine a stale cookie during connection establishment.	The default value is 60 seconds.
sctp_ecn	This tunable enables or disables the explicit congestion notification (RFC 3168).	It is accessed during connection establishment.	The default value is 1.
sctp_ephemeral_high	This tunable specifies the largest port number to allocate for the SCTP (Stream Control Transmission Protocol) ephemeral ports.	It is used when an application is trying to bind to a port.	The default value is 65535.
sctp_ephemeral_low	This tunable specifies the lowest port number to allocate for the SCTP ephemeral ports.	It is used when an application is trying to bind to a port.	The default value is 32768.
sctp_instreams	This tunable specifies the default number of inbound streams that an association uses.	It is used during connection establishment.	The default is 2048.
sctp_outstreams	This tunable specifies the default number of outbound streams that an association uses.	It is used during connection establishment.	The default value is 10.
sctp_pmtu_discover	If enabled, sets the *Don't Fragment* bit in an IP header of an outgoing packet.	It is accessed when the sending packets are sent out.	The default value is 1.
sctp_recv_multibuf	This tunable controls the socket receive buffer accounting. The default value is 0 and it indicates that all the associations belonging to the socket share the same receive buffer space. When set to nonzero, each association has its own receive buffer space of this value. The *setsockopt* function overrides this value.	It is accessed when an association is being created.	The default value is 0 (*multibuf* is not used).

Item	Description		
sctp_send_multibuf	This tunable controls the socket send buffer accounting. The default value is 0 and indicates that all the associations belonging to a socket share the same send buffer space. When set to nonzero, each association has its own send buffer space of this value. The *setsockopt* function overrides this value.	It is accessed when an association is being created.	The default is 0 (*multibuf* is not used).
sctp_failover_type	When enabled, it causes a new path to be chosen after every retransmit timeout. Otherwise, failover happens only after the *path error count* value exceeds *max path error count* value.	It is accessed whenever RTO starts (when there is a packet drop).	The default value is 1.
sctp_check_associd	Governs the pattern related to checking the association ID passed by an application when sending an *ABORT* packet. If set to 0, it ignores the association ID. The association is found by using the foreign address. If set to 1, it performs strict association ID matching. If an association is not found with the passed *assoc_id* value, an *EINVAL* error is returned. If set to 2, it performs association ID matching, but uses the foreign address when a reserved *assoc_id* value is used.	It is accessed whenever a user application issues an ABORT packet.	The default value is 0.

Examples

1. To load the SCTP kernel extension, type the following:

```
sctpctrl load
```

2. To reset the SCTP statistics, type the following:

```
sctpctrl stats reset
```

   This command will zero-out all the SCTP statistics.

3. To get the values of the SCTP tunable, type the following:

```
sctpctrl get
```

   This will list all the SCTP tunable and their values. Here is a sample output.

```
sctp_assoc_maxerr = 10
sctp_cookie_life = 60
sctp_delack_timer = 4
sctp_dontdelayack = 1
sctp_ecn = 1
sctp_ephemeral_high = 65535
sctp_ephemeral_low = 32768
sctp_instreams = 2048
sctp_maxburst = 8
sctp_outstreams = 10
```

```
sctp_path_maxerr = 5
sctp_pmtu_discover = 1
sctp_rttmax = 60
sctp_rttmin = 1
sctp_recvspace = 65536
sctp_sendspace = 65536
sctp_send_fewsacks = 0
```

4. To set **sctp_path_maxerr** to a value of 6, type the following:

```
sctpctrl set sctp_path_maxerr=6
```

Location

/usr/sbin/sctpctrl

Files

Item	Description
/usr/sbin/sctpctrl	Contains the **sctpctrl** command.
/usr/lib/drivers/sctp	Contains the SCTP kernel extension.

sdiff Command

Purpose

Compares two files and displays the differences in a side-by-side format.

Syntax

sdiff [**-l** | **-s**] [**-o** *OutFile*] [**-w** *Number*] *File1 File2*

Description

The **sdiff** command reads the files specified by the *File1* and *File2* parameters, uses the **diff** command to compare them, and writes the results to standard output in a side-by-side format. The **sdiff** command displays each line of the two files with a series of spaces between them if the lines are identical. It displays a < (less than sign) in the field of spaces if the line only exists in the file specified by the *File1* parameter, a > (greater than sign) if the line only exists in the file specified by the *File2* parameter, and a | (vertical bar) for lines that are different.

When you specify the **-o** flag, the **sdiff** command merges the files specified by the *File1* and *File2* parameters and produces a third file.

Note: The **sdiff** command invokes the **diff -b** command to compare two input files. The **-b** flag causes the **diff** command to ignore trailing spaces and tab characters and to consider other strings of spaces as equal.

Flags

Item	Description
-l	Displays only the left side when lines are identical.

Item	Description
-o *OutFile*	Creates a third file, specified by the *OutFile* variable, by a controlled line-by-line merging of the two files specified by the *File1* and the *File2* parameters. The following subcommands govern the creation of this file: **e** Starts the **ed** command with an empty file. **e b or e |** Starts the **ed** command with both sides. **e l or e <** Starts the **ed** command with the left side. **e r or e >** Starts the **ed** command with the right side. **l** Adds the left side to the output file. **r** Adds the right side to the output file. **s** Stops displaying identical lines. **v** Begins displaying identical lines. **q** Performs one of the following functions: • Exits the **ed** command. • Exits the **sdiff** command if no **ed** command is running. • Exits both commands. This action occurs when there are no more lines to be merged into the output file. Each time you exit from the **ed** command, the **sdiff** command writes the resulting edited file to the end of the file specified by the *OutFile* variable. If you do not save the changes before exiting (for example, you press the Ctrl-C key sequence), the **sdiff** command writes the initial input to the output file.
-s	Does not display identical lines.
-w *Number*	Sets the width of the output line. The default value of the *Number* variable is 130 characters. The maximum width of the *Number* variable is 2048. The minimum width of the *Number* variable is 20. The **sdiff** command uses 2048 if a value greater than 2048 is specified.

Exit Status

The **sdiff** command returns the following exit values:

Table 20. Exit status

Item	Description
1	Successful completion.
2	An error occurred.

Examples

1. To print a comparison of two files, enter:

```
sdiff chap1.bak chap1
```

The **sdiff** command displays a side-by-side listing that compares each line of the `chap1.bak` and `chap1` files.

2. To display only the lines that differ, enter:

   ```
   sdiff -s -w 80 chap1.bak chap1
   ```

   The **sdiff** command displays the differences at the workstation. The `-w 80` flag and variable sets the page width to 80 columns. The `-s` flag indicates lines that are identical in both files will not be displayed.

3. To selectively combine parts of two files, enter:

   ```
   sdiff -s -w 80 -o chap1.combo chap1.bak chap1
   ```

   The **sdiff** command combines the `chap1.bak` and `chap1` files into a new file called `chap1.combo`. For each group of differing lines, the **sdiff** command prompts you which group to keep or whether you want to edit them using the **ed** command.

4. To combine and edit two files, `staff.jan` and `staff.apr`, and write the results to the `staff.year` file, perform the steps indicated.

   The `staff.jan` file contains the following lines:

   ```
   Members of the Accounting Department
   Andrea
   George
   Karen
   Sam
   Thomas
   ```

   The `staff.apr` file contains the following lines:

   ```
   Members of the Accounting Department
   Andrea
   Fred
   Mark
   Sam
   Wendy
   ```

   a. Enter the following command:

      ```
      sdiff -o staff.year staff.jan staff.apr
      ```

      The **sdiff** command will begin to compare the contents of the `staff.jan` and `staff.apr` files and write the results to the `staff.year` file. The **sdiff** command displays the following:

      ```
      Members of the Accounting Dept   Members of the Accounting Dept
      Andrea                            Andrea
      George                          | Fred
      %
      ```

      The % (percent sign) is the command prompt.

   b. Enter the **e b** subcommand to start editing the output file with the **ed** command.

      The **sdiff** command displays a sequence of digits, indicating the byte count of lines being merged. In this case, the byte count is 23.

   c. Enter the **q** subcommand to exit the **ed** command and continue combining and editing the two files. The **sdiff** command displays the following:

      ```
      Sam                               Sam
      Thomas                          | Wendy
      ```

   d. Enter the **e b** subcommand again. The **ed** command must be run each time a set of lines from the original two files are to be merged into the output file. The byte count in this instance is 13.

e. Enter the **q** subcommand to save the changes. When all the lines of the two files have been merged into the output file, the **q** subcommand exits the **ed** and **sdiff** commands.

The `staff.year` file now contains the following:

```
Members of the Accounting Department
Andrea
George
Karen
Fred
Mark
Sam
Thomas
Wendy
```

Files

Item	Description
/usr/bin/sdiff	Contains the **sdiff** command.

secldapclntd Daemon

Purpose

Provides and manages connection and handles transactions between the LDAP load module and the LDAP Security Information Server.

Syntax

/usr/sbin/secldapclntd [**-C** *CacheSize*] [**-p** *NumOfThread*] [**-t** *CacheTimeOut*] [**-T** *HeartBeatIntv*] [**-o** *ldapTimeOut*]

Description

The **secldapclntd** daemon accepts requests from the LDAP load module, forwards the request to the LDAP Security Information Server, and passes the result from the server back to the LDAP load module. This daemon reads the configuration information defined in the **/etc/security/ldap/ldap.cfg** file during its startup, authenticates to the LDAP Security Information Server using the specified server distinguished name and password, and establishes a connection between the local host and the server.

If multiple servers are specified in the **/etc/security/ldap/ldap.cfg** file, the **secldapclntd** daemon connects to all of the servers. At a specific time, however, it talks to only one of them. The priority of the server connection is determined by its location in the server list with the highest priority server listed first. The **secldapclntd** daemon can detect when the server it is currently communicating with is down, and automatically switches to another available server. It can also detect when a server becomes available again and re-establish connection to that server. If the reconnected server is of higher priority than the current server then communication is switched to it. This auto-detect feature is done by the **secldapclntd** daemon checking on each of the servers periodically. The time interval between subsequent checking is defaulted to 300 seconds, and can be changed at the daemon startup time from the command line with the **-T** option or by modifying the **heartbeatinterval** value in the **/etc/ security/ldap/ldap.cfg** file.

At startup, the **secldapclntd** daemon tries to establish a connection to the LDAP servers. If it cannot connect to any of the servers, it goes to sleep, and tries again in 30 seconds. It repeats this process twice, and if it still cannot establish any connection, the **secldapclntd** daemon process exits.

The **secldapclntd** daemon is a multi-threaded program. The default number of threads used by this daemon is 10. An administrator can fine-tune the system performance by adjusting the number of threads used by this daemon.

The **secldapclntd** daemon caches information retrieved from the LDAP Security Information Server for performance purpose. If the requested data can be found in the cache and the cache entry is not expired,

the data in the cache is handed back to the requester. Otherwise, the **secldapclntd** daemon makes a request to the LDAP Security Information Server for the information.

The valid number of cache entries for users is in the range of 100-10,000, and that for groups is in the range of 10-1,000. The default is 1000 entries for users, and 100 entries for groups.

The cache timeout or TTL (time to live) can be from 60 seconds to 1 hour (60*60=3600 seconds). By default, a cache entry expires in 300 seconds. If the cache timeout is set to 0, the caching feature is disabled.

Communication between the **secldapclntd** daemon and the LDAP server is performed using asynchronous methods. This allows the daemon to request information from the server and then perform other steps while waiting for the request to return. The length of time that the client will wait for a response from a server is configurable by the administrator and defaults to 60 seconds.

When connecting to LDAP servers, the **secldapclntd** daemon needs to do host lookups. The **nis_ldap** resolver may cause the lookup to be routed back to the daemon itself, resulting in a hang situation. To avoid this problem, the **secldapclntd** daemon ignores the system order of name resolution. Instead, it uses the order defined by the **nsorder** attribute in the **/etc/security/ldap/ldap.cfg** file.

Flags

Note: By default, the **secldapclntd** daemon reads the configuration information specified in the **/etc/security/ldap/ldap.cfg** file at startup. If the following options are given on the command line when starting the **secldapclntd** process, the options from the command line will override the values in the **/etc/security/ldap/ldap.cfg** file.

Flag	Description
-C *CacheSize*	Sets the maximum cache entries used by the **secldapclntd** daemon to *CacheSize* number of entries. The valid range is 100-65536 entries for user cache entry. The default value is 1000. The valid range is 10-65536 for group cache entry. The default is value 100. If you set the user cache entry in the **start-secldapclntd** command, by using the **-C** option, the group cache entry is set to 10% of the user cache entry.
-o *ldapTimeOut*	Timeout period in seconds for LDAP client requests to the server. This value determines how long the client will wait for a response from the LDAP server. Valid range is 0 - 3600 (1 hour). Default is 60 seconds. Set this value to 0 to disable the timeout and force the client to wait indefinitely.
-p *NumOfThread*	Sets the number of threads used by the **secldapclntd** daemon to *NumOfThread* threads. Valid range is 1-256. The default is 10.
-t *CacheTimeout*	Sets the cache to expire in *CacheTimeout* seconds. Valid range is 60- 3600 seconds. The default is 300 seconds.
-T *HeartBeatIntv*	Sets the time interval of heartbeat between this client and the LDAP server. Valid values are 60-3,600 seconds. Default is 300.

Examples

1. To start the **secldapclntd** daemon, type:

```
/usr/sbin/secldapclntd
```

2. To start the **secldapclntd** using 20 threads and cache timeout value of 600 seconds, type:

```
/usr/sbin/secldapclntd -p 20 -t 600
```

Use of the **start-secldapclntd** command is recommended for starting the **secldapclntd** daemon. It is also recommended configuration values are specified in the **/etc/security/ldap/ldap.cfg** file instead of using command line flags, so that these values will be used each time you start the **secldapclntd** process.

secldifconv Command

Purpose

Converts user and group entries of an LDIF from one schema type to another.

Syntax

secldifconv [**-R** *load_module*] **-S** *schematype* **-i** *inputFile* [**-r**]

Description

The **secldifconv** command reads the ldif formatted input file specified by the **-i** option, converts the user and group data using the schema type specified by the **-S** option, and prints the result to stdout. If redirected to a file, the result can be added to an LDAP server with the **ldapadd** command or the **ldif2db** command.

The **-S** option specifies the conversion schema type used for the ldif output. The **secldifconv** command accepts the following schema types:

- **AIX** - AIX schema (aixaccount and aixaccessgroup objectclasses)
- **RFC2307** - RFC 2307 schema (posixaccount, shadowaccount, and posixgroup objectclasses)
- **RFC2307AIX** - RFC 2307 schema with full AIX support (posixaccount, shadowaccount , and posixgroup objectclasses, plus the aixauxaccount and aixauxgroup objectclasses).

The input file specified with the **-i** option can include entries in any of the above supported schemas. The **secldifconv** command will convert user and group entries according to the attribute mapping defined in the **/etc/security/ldap/*.map** files for the corresponding schema type. Only user and group entries will be converted, other entries are output unaltered.

Use of the **-r** option allows the removal of attributes in user and group entries that are not included in the specified output schema. If the option is not specified then unrecognized attributes are assumed to be valid and are output unaltered. Note that if the user or group attribute is defined in the schema **secldifconv** is converting from but not in the schema requested to convert into, then the attribute will not be output. This behavior allows for conversion between the **AIX** and **RFC2307AIX** schemas to the **RFC2307** schema which contains a subset of attributes.

If the **db2ldif** command is used to generate the input file for **secldifconv**, passwords without an encryption prefix are output in {IMASK} format. In order to convert the {imask} format into the proper {crypt} format, the **-R** option should be used to specify the Loadable I&A module to read the password from for conversions from **AIX** schema type, assuming the system has been previously configured to be an LDAP client.

Care should be taken when adding users and groups from other systems to the LDAP server using the **secldifconv** command output. The **ldapadd** and **ldif2db** commands check only for entry name (user name or group name) but not for the numeric ID when adding entries. Merging users and groups from multiple servers using **secldifconv** output can result in sharing of a numeric ID by multiple accounts, which is a security violation. Note that IBM Directory Server 5.2 and later supports a unique attribute feature that can be used to avoid this issue.

Flags

Item	Description
-R *load_module*	Specifies the loadable I&A module used to retrieve the user's password if necessary.
-S *schematype*	Specifies the output LDAP schema type. Valid values are **AIX**, **RFC2307**, and **RFC2307AIX**.

Item	Description
-i *inputFile*	Specifies the input file in ldif format that contains user and group data to convert.
-r	Specifies to remove any attributes that are not defined in the specified schema type.

Exit Status

This command returns the following exit values:

Item	Description
0	The command completed successfully.
>0	An error occurred.
-1	Memory failure (that is, Memory allocation failure).

Examples

1. To convert entries in a ldif formatted file to the `rfc2307` schema, type the following:

   ```
   secldifconv -S rfc2307 -i input.ldif
   ```

   This displays the converted file to stdout in ldif format. User entries and group entries are converted into the **rfc2307** schema type.
2. To convert entries in a ldif formatted file to the `rfc2307aix` schema and remove unrecognized attributes, type the following:

   ```
   secldifconv -R LDAP -S rfc2307aix -i input.ldif -r > convert.ldif
   ```

   This sends the output of the command to the `convert.ldif` file in ldif format. Unrecognized attributes are removed during conversion and user passwords will be requested from the LDAP module if necessary.

Location

/usr/sbin/secldifconv

Files

Mode	File
r	**/etc/security/ldap/2307aixgroup.map**
r	**/etc/security/ldap/2307aixuser.map**
r	**/etc/security/ldap/2307group.map**
r	**/etc/security/ldap/2307user.map**
r	**/etc/security/ldap/aixgroup.map**
r	**/etc/security/ldap/aixuser.map**

sectoldif Command

Purpose

Prints users and groups defined locally to **stdout** in ldif format.

Syntax

sectoldif -d *baseDN* [**-S** *schematype*] [**-u** *username*]

Description

The **sectoldif** command reads users and groups defined locally, and prints the result to **stdout** in ldif format. If redirected to a file, the result can be added to a LDAP server with the **ldapadd** command or the **ldif2db** command.

The **-S** option specifies the schema type used for the ldif output. The **sectoldif** command accepts three schema types:

- **AIX** - AIX schema (**aixaccount** and **aixaccessgroup** objectclasses)
- **RFC2307** - RFC 2307 schema (**posixaccount**, **shadowaccount**, and **posixgroup** objectclasses)
- **RFC2307AIX** - RFC 2307 schema with full AIX support (**posixaccount**, **shadowaccount** , and **posixgroup** objectclasses, plus the **aixauxaccount** and **aixauxgroup** objectclasses).

The **sectoldif** command is called by the **mksecldap** command to export users and groups during LDAP server setup. One needs to be extra cautious when exporting additional users and groups from other systems to the LDAP server using the **sectoldif** output. The **ldapadd** and **ldif2db** commands check only for entry name (user name or group name) but not for the numeric id when adding entries. Exporting users and groups from multiple systems using **sectoldif** output can result in sharing of a numeric id by multiple accounts, which is a security violation.

The **sectoldif** command reads the **/etc/security/ldap/sectoldif.cfg** file to determine what to name the user, group and system sub-trees that the data will be exported to. The **sectoldif** command only exports data to the USER, GROUP and SYSTEM types. The names specified in the file will be used to create sub-trees under the base DN specified with the **-d** flag. Refer to the **/etc/security/ldap/sectoldif.cfg** file documentation for more information.

Flags

Item	Description
-d *baseDN*	Specifies the base DN under which to place the user and group data.
-S *schematype*	Specifies the LDAP schema used to represent user/group entries in the LDAP server. Valid values are AIX, RFC2307, and RFC2307AIX. Default is AIX.
-u *username*	Specifies to print a specific user.

Examples

1. To print all users and groups defined locally, enter the following:

   ```
   sectoldif -d cn=aixsecdb,cn=aixdata -S rfc2307aix
   ```

   This prints all users and groups defined locally to **stdout** in ldif format. User entries and group entries are represented using the rfc2307aix schema type. The base DN is set to `cn=aixsecdb, cn=aixdata`.

2. To print only locally defined user foo, enter the following:

   ```
   sectoldif -d cn=aixsecdb,cn=aixdata -u foo
   ```

   This prints locally defined user foo to **stdout** in ldif format. Without the **-S** option, the default AIX schema type is used to represent foo's ldif output.

Files

Mode	File
r	/etc/passwd
r	/etc/group
r	/etc/security/passwd
r	/etc/security/limits
r	/etc/security/user
r	/etc/security/environ
r	/etc/security/user.roles
r	/etc/security/lastlog
r	/etc/security/smitacl.user
r	/etc/security/mac_user
r	/etc/security/group
r	/etc/security/smitacl.group
r	/etc/security/login.cfg

securetcpip Command

Purpose

Enables the operating system network security feature.

Syntax

securetcpip

Description

The **securetcpip** command provides enhanced security for the network. This command performs the following:

1. Runs the **tcbck -a** command, which disables the nontrusted commands and daemons: **rcp**, **rlogin**, **rlogind**, **rsh** , **rshd**, **tftp**, and **tftpd**. The disabled commands and daemons are not deleted; instead, they are changed to mode 0000. You can enable a particular command or daemon by re-establishing a valid mode.
2. Adds a TCP/IP security stanza to the **/etc/security/config** file. The stanza is in the following format:

```
tcpip:
   netrc = ftp,rexec     /* functions disabling netrc */
```

Before running the **securetcpip** command, acquiesce the system by logging in as root user and executing the **killall** command to stop all network daemons.

> **Attention:** The **killall** command kills all processes except the calling process. If logged in or applications are running, exit or finish before executing the **killall** command.

After issuing the **securetcpip** command, shut down and restart your system. All of your TCP/IP commands and network interfaces should be properly configured after the system restarts.

Files

Item	Description
/etc/security/config	Contains information for the security system.
/etc/security/sysck.cfg	Contains file definitions for the trusted computing base.

sed Command

Purpose

A stream editor.

Syntax

sed [**-n**] [**-u**] *Script* [*File* ...]

sed [**-n**] [**-u**] [**-e** *Script*] ... [**-f** *ScriptFile*] ... [*File* ...]

Description

The **sed** command modifies lines from the specified *File* parameter according to an edit script and writes them to standard output. The **sed** command includes many features for selecting lines to be modified and making changes only to the selected lines.

The **sed** command uses two workspaces for holding the line being modified: the pattern space, where the selected line is held; and the hold space, where a line can be stored temporarily.

An edit script consists of individual subcommands, each one on a separate line. The general form of **sed** subcommands is the following:

[*address-range*] *function*[*modifiers*]

The **sed** command processes each input *File* parameter by reading an input line into a pattern space, applying all **sed** subcommands in sequence whose addresses select that line, and writing the pattern space to standard output. It then clears the pattern space and repeats this process for each line specified in the input *File* parameter. Some of the **sed** subcommands use a hold space to save all or part of the pattern space for subsequent retrieval.

When a command includes an address (either a line number or a search pattern), only the addressed line or lines are affected by the command. Otherwise, the command is applied to all lines.

An address is either a decimal line number, a $ (dollar sign), which addresses the last line of input, or a context address. A context address is a regular expression similar to those used in the **ed** command except for the following differences:

- You can select the character delimiter for patterns. The general form of the expression is:

  ```
  \?pattern?
  ```

  where ? (question mark) is a selectable character delimiter. You can select any character from the current locale except for the space or new-line character. The \ (backslash) character is required only for the first occurrence of the ? (question mark).

  The default form for the pattern is the following:

  ```
  /pattern/
  ```

  A \ (backslash) character is not necessary.
- The **\n** sequence matches a new-line character in the pattern space, except the terminating new-line character.

- A **.** (period) matches any character except a terminating new-line character. That is, unlike the **ed** command, which cannot match a new-line character in the middle of a line, the **sed** command can match a new-line character in the pattern space.

Certain commands called *addressed* commands allow you to specify one line or a range of lines to which the command should be applied. The following rules apply to addressed commands:

- A command line without an address selects every line.
- A command line with one address, expressed in context form, selects each line that matches the address.
- A command line with two addresses separated by commas selects the entire range from the first line that matches the first address through the next line that matches the second. (If the second address is a number less than or equal to the line number first selected, only one line is selected.) Thereafter, the process is repeated, looking again for the first address.

Flags

Item	Description
-e *Script*	Uses the *Script* variable as the editing script. If you are using just one **-e** flag and no **-f** flag, the **-e** flag can be omitted.
-f *ScriptFile*	Uses the *ScriptFile* variable as the source of the edit script. The *ScriptFile* variable is a prepared set of editing commands applied to the *File* parameter.
-n	Suppresses all information normally written to standard output.
-u	Displays the output in an unbuffered mode. When this flag is set, the **sed** command displays the output instantaneously instead of buffering the output. The default is buffered mode.

Note: You can specify multiple **-e** and **-f** flags. All subcommands are added to the script in the order specified, regardless of their origin.

sed Subcommands

The **sed** command contains the following **sed** script subcommands. The number in parentheses preceding a subcommand indicates the maximum number of permissible addresses for the subcommand.

Note:

1. The *Text* variable accompanying the **a**, **c**, and **i** subcommands can continue onto more than one line, provided all lines but the last end with a \ (backslash) to quote the new-line character. Backslashes in text are treated like backslashes in the replacement string of an **s** command and can be used to protect initial blanks and tabs against the stripping that is done on every script line. The *RFile* and *WFile* variables must end the command line and must be preceded by exactly one blank. Each *WFile* variable is created before processing begins.
2. The **sed** command can process up to 999 subcommands in a pattern file.

Item	Description
(1) **a***Text*	Places the *Text* variable in output before reading the next input line.
(2)**b[***label***]**	Branches to the **:** command bearing the *label* variable. If the *label* variable is empty, it branches to the end of the script.
(2)**c***Text*	Deletes the pattern space. With 0 or 1 address or at the end of a 2-address range, places the *Text* variable in output and then starts the next cycle.
(2)**d**	Deletes the pattern space and then starts the next cycle.
(2)**D**	Deletes the initial segment of the pattern space through the first new-line character and then starts the next cycle.

Item	Description
(2)**g**	Replaces the contents of the pattern space with the contents of the hold space.
(2)**G**	Appends the contents of the hold space to the pattern space.
(2)**h**	Replaces the contents of the hold space with the contents of the pattern space.
(2)**H**	Appends the contents of the pattern space to the hold space.
(1)**i***Text*	Writes the *Text* variable to standard output before reading the next line into the pattern space.
(2)**l**	Writes the pattern space to standard output showing nondisplayable characters as 4-digit hexadecimal values. Long lines are folded.
(2)**l**	Writes the pattern space to standard output in a visually unambiguous form. The characters \\\, \\a, \\b, \\f, \\r, \\t, and \\v are written as the corresponding escape sequence. Non-printable characters are written as 1 three-digit octal number (with a preceding backslash character) for each byte in the character (most significant byte first). This format is also used for multibyte characters. This subcommand folds long lines. A backslash followed by a new-line character indicates the point of folding. Folding occurs at the 72nd column position. A $ (dollar sign) marks the end of each line.
(2)**n**	Writes the pattern space to standard output if the default output is not suppressed. It replaces the pattern space with the next line of input.
(2)**N**	Appends the next line of input to the pattern space with an embedded new-line character (the current line number changes). You can use this to search for patterns that are split onto two lines.
(2)**p**	Writes the pattern space to standard output.
(2)**P**	Writes the initial segment of the pattern space through the first new-line character to standard output.
(1)**q**	Branches to the end of the script. It does not start a new cycle.
(2)**r** *RFile*	Reads the contents of the *RFile* variable. It places contents in output before reading the next input line.

Item	Description
(2)**s/***pattern***/***replacement***/***flags*	Substitutes the *replacement* string for the first occurrence of the *pattern* parameter in the pattern space. Any character that is displayed after the **s** subcommand can substitute for the **/** (slash) separator except for the space or new-line character. See the **Pattern Matching** section of the **ed** command. The value of the *flag*s variable must be zero or more of: **g** Substitutes all non-overlapping instances of the *pattern* parameter rather than just the first one. **n** Substitutes for the *n-th* occurrence only of the *pattern* parameter. **p** Writes the pattern space to standard output if a replacement was made. **w *WFile*** Writes the pattern space to the *WFile* variable if a replacement was made. Appends the pattern space to the *WFile* variable. If the *WFile* variable was not already created by a previous write by this **sed** script, the **sed** command creates it.
(2)**t***label*	Branches to the **:***label* variable in the script file if any substitutions were made since the most recent reading of an input line execution of a **t** subcommand. If you do not specify the *label* variable, control transfers to the end of the script.
(2)**w***WFile*	Appends the pattern space to the *WFile* variable.
(2)**x**	Exchanges the contents of the pattern space and the hold space.
(2)**y/***pattern1***/***pattern2***/**	Replaces all occurrences of characters in the *pattern1* variable with the corresponding *pattern2* characters. The number of characters in the *pattern1* and *pattern2* variables must be equal. The new-line character is represented by **\n**.
(2)**!***sed-cmd*	Applies the specified **sed** subcommand only to lines not selected by the address or addresses.
(0)**:***label*	Marks a branch point to be referenced by the **b** and **t** subcommands. This label can be any sequence of eight or fewer bytes.
(1)**=**	Writes the current line number to standard output as a line.
(2)**{***subcmd* **}**	Groups subcommands enclosed in {} (braces).
(0)	Ignores an empty command.
(0)**#**	The "#" and the remainder of the line are ignored (treated as a comment), with one exception. For the first line of a script file, if the character after the # is an n, the default output is suppressed. The rest of the line after the #n is ignored.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.

Item	Description
>0	An error occurred.

Examples

1. To perform a global change, enter:

```
sed  "s/happy/enchanted/g" chap1  >chap1.new
```

   This command sequence replaces each occurrence of the word happy found in the file chap1 with the word enchanted. It puts the edited version in a separate file named chap1.new. The **g** character at the end of the **s** subcommand tells the **sed** command to make as many substitutions as possible on each line. Without the **g** character, the **sed** command replaces only the first occurrence of the word happy on a line.

   The **sed** command operates as a filter. It reads text from standard input or from the files named on the command line (chap1 in this example), modifies this text, and writes it to standard output. Unlike most editors, it does not replace the original file. This makes the **sed** command a powerful command when used in pipelines.

2. To use the **sed** command as a filter in a pipeline, enter:

```
pr  chap2 | sed "s/Page *[0-9]*$/(&)/" | enq
```

   This command sequence encloses the page numbers in parentheses before printing the file chap2. The **pr** command puts a heading and page number at the top of each page, then the **sed** command puts the page numbers in parentheses, and the **enq** command prints the edited listing.

   The **sed** command pattern /Page *[0-9]*$/ matches page numbers that appear at the end of a line. The **s** subcommand changes this to (&), where the & stands for the page number that was matched.

3. To display selected lines of a file, enter:

```
sed  -n  "/food/p" chap3
```

   The sed -n displays each line in the file chap3 that contains the word food. Normally, the **sed** command copies every line to standard output after it is edited. The **-n** flag stops the **sed** command from doing this. You then use subcommands like **p** to write specific parts of the text. Without the **-n** flag, this example displays all the lines in the file chap3, and it shows each line containing food twice.

4. To perform complex editing, enter:

```
sed  -f  script.sed  chap4  >chap4.new
```

   This command sequence creates a **sed** script file when you want to do anything complex. You can then test and modify your script before using it. You can also reuse your script to edit other files. Create the script file with an interactive text editor.

5. A sample **sed** script file:

```
:join
/\\$/{N
s/\\\n//
b join
}
```

   This **sed** script joins each line that ends with a \ (backslash) to the line that follows it. First, the pattern /\\$/ selects a line that ends with a \ for the group of commands enclosed in {} (braces). The **N** subcommand then appends the next line, embedding a new-line character. The s/\\\n// deletes the \ and embedded new-line character. Finally, b join branches back to the label :join to

check for a \ at the end of the newly joined line. Without the branch, the **sed** command writes the joined line and reads the next one before checking for a second \.

Note: The **N** subcommand causes the **sed** command to stop immediately if there are no more lines of input (that is, if the **N** subcommand reads an end-of-file character). It does not copy the pattern space to standard output before stopping. This means that if the last line of the input ends with a \, it is not copied to the output.

6. To copy an existing file (`oldfile`) to a new file (`newfile`) and replace all occurrences of the `testpattern` text string with the contents of the $REPL shell variable, enter:

```
cat oldfile | sed -e "s/testpattern/$REPL/g" > newfile
```

7. To replace all occurrences of A with a, B with b, C with c, and all occurrences of newlines with character Z in the input file, enter:

```
$ sed -f command.file input.file
```

where *command.file* is the script file and *input.file* is the input file.

```
   $cat command.file
   y/ABC\n/abcZ/
```

Alternatively, the following command can also be executed for the same function:

```
sed "y/ABC\n/abcZ/" input.file
```

sedmgr Command

Purpose

Displays and sets Stack Execution Disable flag of the system or executable files.

Syntax

`sedmgr` [`-m` {`off` | `all` | `select` | `setidfiles`}] [`-o` {on | `off`}] [`-c` {`system` | `request` | `exempt`} {*file_name* | *file_group*}] [`-d` {*file_name* | *directory_name*}] [`-h`]

Description

The `sedmgr` command is the manager of the Stack Execution Disable (SED) facility. You can use the command to enable and control the level of stack execution done in the system. This command can also be used to set the various flags in an executable file, controlling the stack execution disable. Any changes to the system wide mode setting will take effect only after a system reboot.

The system wide setting can only be modified by the root user. Other set and reset options on individual executable files will be successful only if the user has write permissions to the file. The SED facility is available only in the AIX 64 bit kernel operating systems.

If invoked without any parameter, the `sedmgr` command will display the current setting in regards to the stack execution disable environment.

For more information, refer to the *Stack Execution Disable Protection* section in **Login control** in the *Security*.

Flags

Item	Description
`-c`	Sets or resets the "request" and "exempt" SED flags in the header of an executable file. Also, sets or resets the SED request and exempt checking flag in the headers of all the executable files in a *file_group*. This option requires write privilege to the file, or root privilege if *file_group* is specified. The possible values are as follows: **system** If the file has the system flag in the executable's header, the operating system decides the operation for the process based on the system-wide SED flags. When the file does not specify any flags, the operating system also decides the operation for the process based on the system wide SED flags. **exempt** Sets a flag in the executable's header that indicates that this file does stack/head based execution and as a result needs exemption from the SED mechanism. The SED request checking bit is turned off. **request** Sets a flag in the executable's header that indicates that this file does not do any stack/data area based execution and as a result is SED capable. The SED exempt checking bit is turned off. You can specify a file group that represents a group of files, such as TCB files. If the specified file name string does not identify a file, then the string is assumed to identify a *file_group*. Currently only the `TCB_files` file group is defined. You can set or reset the SED request and exempt flags for both 32-bit and 64-bit executable. The `-c` flag cannot be used with the `-m`, `-o`, and `-d` flags.
`-d`	Displays the SED request and exempt checking flag for executable files. The SED request and exempt flags are in the file header of an executable. If a directory is specified, then all executable under that directory and its subdirectories are displayed with their SED related flags. This flag requires read privilege to the *file_name* or *directory_name*. The `-d` flag cannot be used with the `-m`, `-o` and `-c` flags.
`-h`	Displays the syntax of the `sedmgr` command.

Item	Description
`-m`	Sets the system-wide stack execution disable mode if the processor supports SED. Any changes to the system-wide setting require a system reboot to take effect. This option will accept one of the following values: **all** Enforces stack execution disable for all files except the ones requesting (marked for) exemption. **off** Turns off the stack execution disable functionality on the system. **select** Sets the mode of operation to select the set of processes that will be enabled and monitored for stack execution disable. Only processes from files with the "request" SED flag set in their headers will be selected. **setidfiles** Sets the mode of operation so that the operating system performs SED for the files with the "request" SED flag set and enables SED for the executable files with the following characteristics: • **setuid** files owned by root. • `setid` files with primary group as "system" or "security". The configured SED attribute is effective at the next 64-bit kernel boot time. Because the SED attribute in ODM does not affect 32-bit kernels, the SED monitoring flag is turned off in that case. If a processor does not support SED, the `sedmgr` command returns an error with the `-m` flag. The `-m` flag cannot be used with the `-c` and `-d` flags.

Item	Description
`-o`	This option enables SED to monitor instead of terminating the processes when exceptions occur. This option allows you to evaluate if an executable is doing any legitimate stack execution. This setting works with the system-wide mode set using the `-c` option. The SED Monitoring Control flag is part of the system-wide SED settings stored in ODM. Changing this setting requires root privilege. The possible values for this flag are as follows: **on** Turns on the monitoring for SED facility. When operating in this mode, the system will allow the process to continue operating even if an SED related exception occurs. Instead of terminating the process, the operating system logs the exception in the AIX error log subsystem. **off** Turns off the monitoring mode for SED facility. In this mode, the operating system terminates any process that violates and raises an exception per SED facility. The configured SED attribute is effective at the next 64-bit kernel boot time. Because the SED attribute in ODM does not affect 32-bit kernels, the SED monitoring flag is turned off in that case. If a processor does not support SED, the `sedmgr` command returns an error with the `-m` flag. The `-o` flag cannot be used with the `-c` and `-d` flags.
None	If no flag is specified, the `sedmgr` command displays the current setting in regards to the stack execution disable environment. It displays the current SED setting in the kernel `var` structure and the system-wide SED settings in ODM.

Parameters

Item	Description
file_name	Name of the executable file whose SED settings are changed. Requires write privilege.
file_group	Group of executable files whose SED settings are changed when a file name is not specified. Requires root privilege.
directory_name	Directory of executable files and any subdirectories of executable files whose SED checking flags are displayed with the `-d` flag.

Exit Status

Item	Description
`0`	The command completed successfully.

Item	Description
255	An error occurred.

Security

Access Control: This command should be a standard user command and have the trusted computing base attribute.

Examples

1. To change the system-wide SED Mode flag to `setidfiles` and the SED Control flag to on, type:

```
sedmgr -m setidfiles -o on
```

2. To change the SED checking flag to `exempt` for the `plans` file, type:

```
sedmgr -c exempt plans
```

3. To change the SED checking flag to `select` for all the executable files marked as a TCB file, type:

```
sedmgr -c request TCB_files
```

4. To display the SED checking flag of the `plans` file, type:

```
sedmgr -d plans
```

Restrictions

Auditing Events: If the auditing subsystem has been properly configured and is enabled, the `sedmgr` command generates the following audit record (event):

Event	Information
SEDMGR_Odm	System wide SED setting.
SEDMGR_File	SED setting in an executable file header.

See Setting up auditing in the **Auditing overview** section of *Security* for more details about how to properly select and group audit events, and how to configure audit event data collection.

Location

`/usr/sbin/sedmgr`

Files

Item	Description
`/usr/bin/tcbck`	Accessed in executable mode.
`/usr/bin/ldedit`	Accessed in executable mode.

send Command

Purpose

Sends a message.

Syntax

send [*File* ... | { **-draft** | **-nodraftfolder** | **-draftfolder +***Folder* | **-draftmessage** *Message* }] [**-alias** *File*] [**-format** | **-noformat**] [**-nomsgid** | **-msgid**] [**-nofilter** | **-filter** *File*] [**-nopush** | **-push**] [**-forward** | **-noforward**] [**-noverbose** | **-verbose**] [**-nowatch** | **-watch**]

Description

The **send** command routes messages through the mail delivery system. If the delivery fails, the **send** command displays an error message. By default, `From:` and `Date:` fields are added to each specified message. Unless a **$SIGNATURE** environment variable or `signature:` profile entry exists, the **send** command places the sender's address in the `From:` field.

The **send** command puts the current date in the `Date:` field. If the **dist** command calls the **send** command, the **send** command adds `Resent-` to the `From:`, `Date:`, and `Message-ID:` fields.

After successful delivery, the **send** command removes messages from active status by renaming them. The system renames messages by prefacing the current message number with a , (comma). Inactive files are unavailable to the Message Handler (MH) package. However, system commands can still manipulate inactive files. Until you use the **send** command again, you can retrieve an inactive file.

Flags

Item	Description
-alias *File*	Specifies a mail alias file to be searched. Three MH profile entries are required to use MH aliases: `ali: -alias Aliases` `send: -alias Aliases` `whom: -alias Aliases` where `Aliases` is the file to be searched. The default alias file is **/etc/mh/MailAliases**.
-draft	Uses the current draft message if no file is specified. Without this flag and when no file is specified, the **send** command asks the user if the current draft message is the one to use.
-draftfolder +*Folder*	Specifies the draft folder that contains the draft message to be sent. The **-draftfolder +***Folder* flag followed by a *Message* parameter is the same as specifying the **-draftmessage** flag.

Item	Description
-draftmessage *Message*	Specifies the message to be sent. You can use one of the following message references as the value of the *Message* parameter: ***Number*** Number of the message. **cur or . (period)** Current message. This is the default. **first** First message in a folder. **last** Last message in a folder. **next** Message following the current message. **prev** Message preceding the current message.
-filter *File*	Uses the format instructions in the specified file to reformat copies of the message sent to the recipients listed in the `Bcc:` field.
-format	Puts all recipient addresses in a standard format for the delivery transport system. This flag is the default.
-forward	Adds a failure message to the draft message and returns it to the sender if the **send** command fails to deliver the draft. This flag is the default.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-msgid	Adds a message-identification component (such as `Message-ID:`) to the message.
-nodraftfolder	Undoes the last occurrence of the **-draftfolder +***Folder* flag. This flag is the default.
-nofilter	Removes the `Bcc:` header field from the message for recipients listed in the `To:` and `cc:` fields. The flag then sends the message with minimal headers to recipients listed in the `Bcc:` field. This flag is the default.
-noformat	Prevents alteration of the format of the recipient addresses.
-noforward	Prevents return of the draft message to the sender if delivery fails.
-nomsgid	Prevents addition of a message-identification component. This flag is the default.
-nopush	Runs the **send** command in the foreground. This flag is the default.
-noverbose	Prevents display of information during the delivery of the message to the **sendmail** command. This flag is the default.
-nowatch	Prevents display information during delivery by the **sendmail** command. This flag is the default.
-push	Runs the **send** command in the background. The **send** command does not display error messages on the terminal if delivery fails. Use the **-forward** flag to return messages to you that are not delivered.

Item	Description
-verbose	Displays information during the delivery of the message to the **sendmail** command. This information allows you to monitor the steps involved in sending mail.
-watch	Displays information during the delivery of the message by the **sendmail** command. This information allows you to monitor the steps involved in sending mail.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Draft-Folder:`	Sets the default folder for drafts.
`mailproc:`	Specifies the program used to post failure notices.
`Path:`	Specifies the user's MH directory.
`postproc:`	Specifies the program used to post messages.
`Signature:`	Sets the mail signature.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Examples

To send a draft message that is in your **$HOME/Mail/draft** file, enter:

```
send
```

The system responds with a message similar to the following:

```
Use "/home/david/Mail/draft"?
```

If you enter `yes`, the draft message is sent, and you are returned to the shell prompt. In this example, the name of the **$HOME** directory is **/home/david**.

Files

Item	Description
$HOME/.mh_profile	Specifies the MH user profile.
/usr/bin/send	Contains the **send** command.

sendbug Command

Purpose

Mails a system bug report to a specified address.

Syntax

sendbug [*Address*]

Description

The **sendbug** command is a shell script to assist the user in composing and mailing bug reports in the correct format.

The **sendbug** command starts the editor specified by the **EDITOR** environment variable on a temporary copy of the bug report format outline. The default editor is vi.

Fill out the appropriate fields in the bug report format outline and exit the editor. The **sendbug** command mails the completed report to the address specified by the *Address* parameter. The default address is POSTMASTER.

Files

Item	Description
/usr/lib/bugformat	Contains the bug report outline.

sendmail Command

Purpose

Routes mail for local or network delivery.

Syntax

sendmail [**-ba** | **-bd** | **-bD** | **-bh** | **-bH** | **-bi** | **-bm** | **-bp** | **-bs** | **-bv** | **-bt** [**-Ac** *File*] [**-C** *File*] [**-D** *Log File*] [**-d** *Value*]] [**-B** *Type*] [**-F** *FullName*] [**-f** *Name*] [**-G**] [**-h** *Number*] [**-i**] [**-M***x Value*] [**-n**] [**-N** *Dsn*] [**-O** *Option*=*Value*] [**-o** *Option* [*Value*]] [**-p** *Protocol*] [**-q** [*Time*]] [**-qG**name] [**-qI***Substr*] [**-qR***Substr*] [**-qS***Substr*] [**-R** *Return*] [**-r** *addr*] [**-t**] [**-V** *Envid*]] [**-v**] [**-X** *LogFile*] *Address*

Note: The *Address* parameter is optional with the **-bd**, **-bi**, **-bp**, **-bt**, and **-q** [*Time*] flags.

Description

Note:

- Starting with AIX 7 with 7200-04, the **sendmail** command uses Sendmail version 8.15.2. The **sendmail** command can be run by a new smmsp user and smmsp group instead of the root user for enhanced security.
- In Sendmail v8.7, and later, name resolution ordering is Domain Name System (DNS), Network Information Services (NIS), Network Interface Services (NIS), then local. If you want to override this default order, specify an order in the `/etc/netsvc.conf` file or specify the *NSORDER* environment variable.

The **sendmail** command receives formatted text messages and routes the messages to one or more users. Used on a network, the **sendmail** command translates the format of a message's header information to match the requirements of the destination system. The program determines the network of the destination system by using the syntax and content of the addresses.

The **sendmail** command can deliver messages to:

- Users on the local system
- Users connected to the local system by using the TCP/IP protocol
- Users connected to the local system by using the Basic Networking Utilities (BNU) command protocol

Use the **sendmail** command only to deliver pre-formatted messages. The **sendmail** command is not intended as a user interface routine; other commands provide user-friendly interfaces.

The **sendmail** command reads standard input for message text. The **sendmail** command sends a copy of the message to all addresses listed whenever it reads an end of the message character. The end of the message character is either an end-of-file (Ctrl-D) control sequence or a single period on a line.

Details about the **sendmail** command are explained in the following subsections:

sendmail Mail Filter API (Milter)

The **sendmail** Mail Filter API provides access to mail messages as they are being processed so that third-party programs can filter meta-information and content. Filters that are developed using the sendmail Mail Filter API use threads, so it might be necessary to alter the per-process limits in your filter. For example, if your filter is frequently used, use the **setrlimit** subroutine to increase the number of open file descriptors.

Specifying filters in sendmail configs

Use the key letter **X** (for external) to specify filters. The following are three example filters:

```
Xfilter1, S=local:/var/run/f1.sock, F=R
```

```
Xfilter2, S=inet6:999@localhost, F=T, T=C:10m;S:1s;R:1s;E:5m
```

```
Xfilter3, S=inet:3333@localhost
```

You can specify filters in your `.mc` file. The following filter attaches to a UNIX-domain socket in the /var/run directory:

```
INPUT_MAIL_FILTER(`filter1', `S=local:/var/run/f1.sock, F=R')
```

The following filter uses an IPv6 socket on port 999 of localhost:

```
INPUT_MAIL_FILTER(`filter2', `S=inet6:999@localhost, F=T, T=C:10m;S:1s;R:1s;E:5m')
```

The following filter uses an IPv4 socket on port 3333 of localhost:

```
INPUT_MAIL_FILTER(`filter3', `S=inet:3333@localhost')
```

sendmail mail filter flags

R
: Reject connection if filter is not available.

T
: Temporarily fail connection if filter is not available.

If neither F=R or F=T is specified, the **sendmail** command passes the message as if the filter is not present. The separator is a comma (**,**).

sendmail mail filter timeouts

You can override the default sendmail timeouts with T=*x*. There are four fields in the T= statement:

C
Timeout for connecting to a filter (if 0, use system timeout).

S
Timeout for sending information from the MTA to a filter.

R
Timeout for reading reply from the filter.

E
Overall timeout between sending end-of-message to filter and waiting for the final acknowledgment.

The separator between each entry is a semicolon (**;**).

The default values are:

- T=C:0m;S:10s;R:10s;E:5m

The InputMailFilters option determines which filters are invoked and how the filters are sequenced:

```
InputMailFilters=filter1, filter2, filter3
```

This is set automatically according to the order of the INPUT_MAIL_FILTER commands in your .mc file. You can also reset the value by setting confINPUT_MAIL_FILTERS in your .mc file. This option calls the three filters in the order the filters were specified.

You can define a filter without adding it to the input filter list by using `MAIL_FILTER()` instead of `INPUT_MAIL_FILTER()` in your .mc file.

Note: If InputMailFilters is not defined, no filters will be used.

Using the sendmail configuration files

In AIX 7 with 7200-03 and earlier, the **sendmail** command uses a single configuration file, `/etc/mail/sendmail.cf`, to set operational parameters and to determine how the command parses addresses. Starting with AIX 7 with 7200-04, the **sendmail** command supports the Mail Submission Program mode (MSP_mode) by using the `/etc/mail/submit.cf` configuration file. The **sendmail** command in the MSP mode does not require root privileges. Therefore, the **sendmail** command in MSP mode is more secure as compared to the previous version. The **sendmail** command uses the `sendmail.cf` configuration file when the **sendmail** command runs as mail server daemon in the Mail Transmission Agent (MTA) mode. For information about the security consideration of the **sendmail** command, see http://www.sendmail.org/~ca/email/doc8.12/SECURITY.

The sendmail configuration files are described as follows:

/etc/mail/sendmail.cf

This configuration file is used when the **sendmail** command runs as mail server daemon in the MTA mode. By default, the `sendmail.cf` file uses the mail queue in the `/var/spool/mqueue` directory. On system boot, the **sendmail** MTA daemon is started in the `/etc/rc.tcpip` directory by default. To manually start the **sendmail** MTA daemon, enter the following command:

```
# startsrc -s sendmail -a " -bd -q30m"
```

/etc/mail/submit.cf

This configuration file is used by the **sendmail** command to operate in the MSP mode. By default, the `submit.cf` file uses the system mail queue in the `/var/spool/clientmqueue` directory. The `sendmail` command operates in the MSP mode under the following scenarios:

- When the **sendmail** command is run at command line or is called by another mail facility (such as the **mail** command) to send email.
- When the **sendmail** command is invoked as client-queue runner. The **sendmail** client-queue runner identifies the undelivered messages in the `/var/spool/clientmqueue` directory and

submits the messages to the `sendmail` MTA daemon for delivery. Enter the following command to run the `sendmail` command as a queue runner in MSP mode manually:

```
# /usr/lib/sendmail -Ac -q 30m
```

You can also set the **sendmail** MTA daemon to start automatically whenever the system boots by editing the `/etc/rc.tcpip` file. For instructions about editing the `/etc/rc.tcpip` file, see Starting the sendmail daemon during system boot.

Restarting and refreshing the sendmail processes

The configuration files that are used by the **sendmail** command are text files that you can edit by using any text editor. After modifying any of these configuration files, you must restart or refresh the MTA daemon and the MSP for the changes to take effect.

The current process ID of the **sendmail** command is stored in the `/etc/mail/sendmail.pid` file. Run the following **kill** command to allow the **sendmail** command reread the newly edited configuration files:

```
#kill -15 `head -1 /etc/mail/sendmail.pid`
```

If the `srcmstr` command is running, you can run the **refresh** command to build the configuration database, the aliases database, and the NLS database again:

```
#refresh -s sendmail
```

If you started the `sendmail` MSP manually, and if the sendmail process is not controlled by the `srcmstr` command, you can stop the sendmail process by using the following **kill** command:

```
# kill <pid of the sendmail: Queue runner >
```

Migrating to AIX 7 with 7200-04

If you are running AIX 7 with 7200-03, or earlier, and if you configure the **sendmail** command, when you migrate to AIX 7 with 7200-04, the **sendmail** command runs as an MTA daemon. Back up the `sendmail.cf` configuration file before starting the migration operation by running the following command:

```
# cp /etc/mail/sendmail.cf /etc/mail/sendmail.cf.old
```

After the migration operation is complete, the previous `sendmail.cf` configuration file is transferred to the new `sendmail.cf` configuration file.

Complete the following steps after you migrate your AIX operating system to AIX version 7.2.4, or later:

1. If you cannot find the `sendmail.cf` file after migration, restore the backup file by running the following command:

   ```
   # cp /etc/mail/sendmail.cf.old /etc/mail/sendmail.cf
   ```

2. Restart and refresh the `sendmail` processes by running the following command:

   ```
   # startsrc -s sendmail -a " -bd -q30m"
   ```

   ```
   # refresh -s sendmail
   ```

   These commands run the **sendmail** command as MTA.

The **sendmail** command rereads the databases and continues the operation with the `sendmail.cf` configuration file.

Defining Aliases

The**sendmail** command allows you to define aliases to use when the **sendmail** command handles the local mail. Aliases are alternative names that you can use in place of elaborate network addresses. You can also use aliases to build distribution lists.

Define aliases in the **/etc/mail/aliases** file. This file is a text file you can edit. The **sendmail** command uses a database version of this file. Before any changes made to the **/etc/mail/aliases** file become effective, you must build a new alias database by running the **sendmail-bi** command or the **newaliases** command.

Berkeley DB support is available on AIX for Sendmail 8.11.0. Sendmail will continue to read the aliases in the DBM format until the aliases database gets rebuilt. Once rebuilt, Sendmail will read the aliases in the Berkeley DB format and store them in the **/etc/mail/aliases.db** file.

Note: When defining aliases in the **/etc/mail/aliases** file, use only lowercase characters for nested aliases. Uppercase characters on the right-hand side of an alias are converted to lowercase before being stored in the aliases database. In the following example, mail sent to `testalias` fails, because TEST is converted to `test` when the second line is stored.

```
TEST: user@machine
testalias: TEST
```

Every system must have a user or user alias designated as the **postmaster** alias. The default **postmaster** alias is a root file. You can assign this alias to a different user in the **/etc/mail/aliases** file. The **postmaster** alias allows other users outside your system to send mail to a known ID and to get information about mailing to users on your system. Also, users on your system can send problem notifications to the **postmaster** ID.

The **sendmail** command first opens a database in the format of hash-style aliases file. If it fails or if the NEWDB support was not compiled, the command opens a NDBM database. If that fails, the **sendmail** command reads the aliases source file into its internal symbol table.

Flags

Item	Description
-Ac *File*	Specifies the **sendmail** command to choose an alternative configuration file based on the operative mode. If you specify **-bm**, **-bs**, or **-t** flags, the **sendmail** command uses the `submit.cf` configuration file. If any other flags are specified, and for compatibility with earlier versions, the **sendmail** command uses the `sendmail.cf` configuration file. If you do not specify the *file* variable, by default, the **sendmail** command uses the `submit.cf` configuration file.
-B *Type*	Sets the body type to *type*. Current legal values are 7BI or 8BITMIME. **Note:** The **-b** flag is mutually exclusive.
-ba	Starts the **sendmail** command in ARPANET mode. All input lines to the command must end with a carriage return and a line feed (CR-LF). The **sendmail** command generates messages with a CR-LF at the end and looks at the `From:` and `Sender:` fields to find the name of the sender.
-bd	Starts the **sendmail** command as a daemon running in the background as a Simple Mail Transfer Protocol (SMTP) mail router.
-bD	Starts the **sendmail** command as a daemon running in the foreground as a Simple Mail Transfer Protocol (SMTP) mail router.
-bh	Prints the persistent host status database.
-bH	Purges the persistent host status database.

Item	Description
-bi	Builds the alias database from information defined in the **/etc/mail/aliases** file. Running the **sendmail** command with this flag is the same as running the **/usr/sbin/newaliases** command.
-bm	Delivers mail in the usual way. (This is the default.)
-bp	Prints a listing of the mail queue. Running the **sendmail** command with this flag is the same as running the **/usr/sbin/mailq** command.
-bs	Uses the simple mail transfer protocol (SMTP) as described in RFC821 to collect mail from standard input. This flag also includes all of the operations of the **-ba** flag that are compatible with SMTP.
-bt	Starts the **sendmail** command in address test mode. This mode allows you to enter interactive addresses and watch as the **sendmail** command displays the steps it takes to parse the address. At the test-mode prompt, enter a rule set or multiple rule sets separated by commas and an address. Use this mode for debugging the address parsing rules in a new configuration file.
-bv	Starts the **sendmail** command with a request to verify the user IDs provided in the *Address* parameter field of the command. The **sendmail** command responds with a message telling which IDs can be resolved to a mailer command. It does not try to collect or deliver a message. Use this mode to validate the format of user IDs, aliases, or mailing lists.
-C *File*	Starts the **sendmail** command using an alternate configuration file specified by the *File* variable. Use this flag together with **-bt** to test a new configuration file before installing it as the running configuration file.
-D *Log File*	Sends the debugging output to the specified log file. The **-D** option must be before the **-d** option.
-d *Value*	Sets the debugging value to the value specified by the *Value* variable. The only valid value is 21.*n*, where *n* is any nonzero integer. This produces information regarding address parsing and is typically used with the **-bt** flag. Higher values of *n* produce more verbose information. Root permissions are required for this flag.
-F *FullName*	Sets the full name of the sender to the string provided in the *FullName* variable.
-f *Name*	Sets the name of the from person (the envelope sender of the mail). This address may also be used in the From: header if that header is missing during initial submission. The envelope sender address is used as the recipient for delivery status notifications and may also appear in a Return-path: header. This flag should only be used by trusted users (normally root, daemon, and uucp) or if the person you are trying to become is the same as the person you are. Otherwise, an X-Authentication-Warning header is added to the message.
-G	Relay (gateway) submission of a message. For example, when the **rmail** command calls the **sendmail** command.
-h *Number*	Sets the hop count to the value specified by the *Number* variable. The hop count is the number of times that the message has been processed by an SMTP router (not just the local copy of the **sendmail** command). The mail router increments the hop count every time the message is processed. When it reaches a limit, the message is returned with an error message in order to prevent infinite loops in the mail system.
-i	Ignores dots alone on lines by themselves in incoming messages. This should be set if you are reading data from a file.
-L	Sets the identifier used in syslog messages to the supplied tag.
-M*x Value*	Sets marco *x* to the specified *value*.

Item	Description
-N *Dsn*	Sets delivery status notification conditions to DSN. The delivery status notification conditions can be: `never` for no notifications or for a comma separated list of the values, `failure` for notification if delivery failed, `delay` for notification if delivery is delayed, and `success` for notification when the message is successfully delivered.
-n	Prevents the **sendmail** command from interpreting aliases.
-O *Option*=*Value*	Sets *Option* to the specified *Value*. Use for long-form option names.
-o *Option* [*Value*]	Sets the *Option* variable. If the option is a valued option, you must also specify a value for the *Value* variable. **Note:** For valid values, see **Options for the sendmail Command in the sendmail.cf** file in *Performance Tools Guide and Reference*.
-p *Protocol*	Sets the sending protocol. It is recommended that you set this. You can set *Protocol* in the form *Protocol*:*Host* to set both the sending protocol and the sending host. For example, `-pUUCP:uunet` sets the sending protocol to UUCP and the sending host to uunet. Some existing programs use **-oM** flag to set the **r** and **s** macros, which is equivalent to using the **-p** flag.
-qI*Substr*	Limits process jobs to those containing *Substr* as a substring of the queue ID.
-qGname	Processes jobs in a queue group called by name only.
-qR*Substr*	Limits process jobs to those containing *Substr* as a substring of one of the recipients.
-qS*Substr*	Limits process jobs to those containing *Substr* as a substring of the sender.
-q [*Time*]	Processes saved messages in the queue at the intervals specified by the *Time* variable. If the *Time* variable is not specified, this flag processes the queue at once.
-R *Return*	Sets the amount of the message to be returned if the message bounces. The *Return* parameter can be `full` to return the entire message or `hdrs` to return only the headers.
-r *addr*	An obsolete form of **-f**.
-t	Sends the message to the recipients specified in the `To:`, `Cc:`, and `Bcc:` fields of the message header, as well as to any users specified on the command line.
-V *Envid*	Sets the original envelope ID. This is propagated across SMTP to servers that support DSNs and is returned in DSN-compliant error messages.
-v	Starts the **sendmail** command in verbose mode. The **sendmail** command displays messages regarding the status of transmission and the expansion of aliases.
-X *LogFile*	Logs all traffic in and out of **sendmail** in *LogFile* for debugging mailer problems. Use this flag sparingly, since it produces a lot of data very quickly.

You can also set or remove the **sendmail** configuration processing options. The person responsible for the mail system uses these options. To set these options, use the **-o** flag on the command line or the **O** control line in the configuration (**/etc/mail/sendmail.cf**) file.

Exit Status

The **sendmail** command returns exit status values. These exit values are defined in the **/usr/include/sysexits.h** file. The following table summarizes the meanings of these return values:

Item	Description
EX_CANTCREAT	The **sendmail** command cannot create a file that the user specified.
EX_CONFIG	An error was found in the format of the configuration file.
EX_DATAERR	The input data was incorrect in some way.
EX_IOERR	An error occurred during I/O.
EX_NOHOST	The **sendmail** command could not recognize the specified host name.
EX_NOINPUT	An input file (not a system file) did not exist or was not readable.
EX_NOPERM	The user does not have permission to perform the requested operation.
EX_NOUSER	The **sendmail** command could not recognize a specified user ID.
EX_OK	The **sendmail** command successfully completed.
EX_OSERR	A temporary operating system error occurred. An example of such an error is a failure to create a new process.
EX_OSFILE	A system file error occurred. For example, a system file (such as **/etc/passwd**) does not exist, cannot be opened, or has another type of error preventing it from being used.
EX_PROTOCOL	The remote system returned something that was incorrect during a protocol exchange.
EX_SOFTWARE	An internal software error occurred (including bad arguments).
EX_TEMPFAIL	The **sendmail** command could not create a connection to a remote system. Try the request again later.
EX_UNAVAILABLE	A service or resource that the **sendmail** command needed was not available.
EX_USAGE	The command syntax was not correct.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Auditing Events:

Event	Information
SENDMAIL_Config	Configuration event
SENDMAIL_ToFile	File-creation event

Example

Run the following command to display the sendmail version:

```
echo \$Z | sendmail -d0
```

The system responds with a message similar to the following:

```
Version AIX5.2/8.11.6p2
 Compiled with: LDAPMAP MAP_REGEX LOG MATCHGECOS MIME7TO8 MIME8TO7
             NAMED_BIND NDBM NETINET NETINET6 NETUNIX NEWDB NIS NISPLUS
                QUEUE SCANF SMTP USERDB XDEBUG

============ SYSTEM IDENTITY (after readcf) ============
      (short domain name) $w = dodgers
```

```
   (canonical domain name) $j = dodgers.usca.ibm.com
          (subdomain name) $m = usca.ibm.com
               (node name) $k = dodgers
========================================================

Recipient names must be specified
# oslevel -r
5200-02
#
```

Files

Item	Description
/usr/sbin/sendmail	Contains the **sendmail** command.
/usr/sbinmailq/	Contains the mail queue.
/usr/sbin/newaliases	Contains the alias database.
/usr/sbin/mailstats	Contains statistics found in the **/usr/lib/sendmail.st** file.
/etc/mail/aliases	Contains the text version of the **sendmail** command aliases.
/etc/mail/aliases.db	Contains Berkeley DB formatted database for aliases.
/etc/mail/aliases.dir	Contains DBM formatted database for aliases.
/etc/mail/aliases.pag	Contains DBM formatted database for aliases.
/etc/mail/sendmail.cf	Contains the text version of the **sendmail** configuration file.
/etc/mail/submit.cf	Contains the text version of the **sendmail** configuration file. If this file exists, this file is considered as the default configuration file.
/etc/sendmail.st	Contains mail routing statistics information.
/usr/lib/smdemon.cleanu	Maintains aging copies of the log file found in the **/var/spool/mqueue** directory.
/var/spool/mqueue	Contains the temporary files and the log file associated with the messages in the mail queue.
/usr/bin/uux	Contains the mailer command to deliver Basic Networking Utilities (BNU) mail.
/usr/bin/bellmail	Contains the mailer command to deliver local mail.

setclock Command

Purpose

Sets the time and date for a host on a network.

Syntax

/usr/sbin/setclock [*TimeServer*]

Description

The **/usr/sbin/setclock** command gets the time from a network time server, and if run by a user with root user authority, sets the local time and date accordingly.

The **setclock** command takes the first response from the time server, converts the calendar clock reading found there, and displays the local date and time. If the **setclock** command is run by the root user, it calls the standard workstation entry points to set the system date and time.

If no time server responds or if the network is not operational, the **setclock** command displays a message to that effect and leaves the current date and time settings of the system unchanged.

Note: Any host running the **inetd** daemon can act as a time server.

Parameter

Item	Description
TimeServer	The host name or address of a network host that services TIME requests. The **setclock** command sends an Internet TIME service request to a time server host. If the *TimeServer* name is omitted, the **setclock** command sends the request to the default time server. The default time server in a DOMAIN environment is specified by the name server. Otherwise the default time server is specified in the **/etc/hosts** file.

Examples

1. To display the date and time using the time server host specified in the **/etc/hosts** file, enter:

   ```
   setclock
   Sat Mar 11 15:31:05 1988
   ```

   The **setclock** command displays the proper date and time.
2. To set the date and time, enter:

   ```
   su root
   setclock host1
   Thu Jan 12 15:24:15 1990
   ```

   You must use the **su** command or log in as the root user before setting the time from the time server in `host1`.

setea Command

Purpose

Writes or deletes a named extended attribute to a file.

Syntax

setea **-n** *Name* [**-l**]{ **-v** *Value* | **-d** | **-f** *EAFile* } *FileName* ...

Description

The **setea** command writes or deletes a named extended attribute to a file. The file must be in a file system which supports named extended attributes, such as JFS2 using **v2** extended attribute format.

Note: To prevent naming collisions, JFS2 has reserved the 8-character prefix `(0xf8)SYSTEM(0xF8)` for system-defined extended attributes. Avoid using this prefix for naming user-defined extended attributes.

This command is not used to set ACLs. To set ACLs, use the **aclput** command.

Flags

Item	Description
-d	Specifies to delete the named extended attribute from the file.
-f *EAFile*	*EAFile* specifies a file which contains the EA value. If an extended attribute matching the specified name already exists for the *FileName*, then the value will be changed to the value specified.

Item	Description
-l	Specifies to write or delete the extended attribute from the symbolic link itself rather than the file to which it is pointing.
-n *Name*	Specifies the name of the extended attribute to be written.
-v *Value*	Specifies the value of the named extended attribute. If an extended attribute matching the specified name already exists for the file, then the value will be changed to the value specified. Value is treated as a character string. It should be enclosed in quotes if it contains spaces.
FileName ...	Specifies the file or files from which to write or delete the extended attribute.

Exit Status

Item	Description
0	Successful completion.
Positive integer	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To create an extended attribute with a name of `Approver` and a value of `Grover` for file `design.html`, enter:

```
setea -n Approver -v Grover design.html
```

2. To modify an extended attribute named `Approver` to new value of `Joon` for file `design.html`, enter:

```
setea -n Approver -v Joon design.html
```

3. To remove an extended attribute named `Approver` from file `design.html`, enter:

```
setea -n Approver -d design.html
```

4. To create an extended attribute with a name of `Approver` and a value of `Zach` for the symbolic link `design.html`, enter:

```
setea -n Approver -v Zach -l design.html
```

Location

/usr/sbin

setgroups Command

Purpose

Resets a session's process group set.

Syntax

setgroups [**-**] [**-a** *GroupSet*] [**-d** *GroupSet*] [**-r** [*Group*]] [*GroupSet*]

Description

The **setgroups** command, by default, displays the user's current group set and process group set for the current shell. A user's group set is defined in the user database files. When given a flag and a *GroupSet* parameter, this command resets the process group set as listed by the *GroupSet* parameter. The *GroupSet* parameter is a comma-separated list of group names. The available groups are defined in the user database files.

You can also use the **setgroups** command to add or delete groups from the current group set. Using the **-r** flag, you can reset the real group ID. If you specify the *Groupset* parameter but no flags, the **setgroups** command resets all the groups and makes the first group in the list the real group. The **setgroups** command does not change the security characteristics of the controlling terminal.

When you run the **setgroups** command, the system always replaces your shell with a new one. The command replaces your shell regardless of whether the command is successful or not. For this reason, the command does not return error codes.

The **setgroups -r** command is identical to the **newgrp** command.

Flags

Item	Description
-a *GroupSet*	Adds the groups specified by the *GroupSet* parameter to the current session. The number of groups in the new set must not exceed **NGROUPS_MAX** groups, a value defined in the **limits.h** file. The real group ID is not changed.
-d *GroupSet*	Removes the groups specified by the *GroupSet* parameter from the current session. If the real group is removed, the next group listed in the current set becomes the real group.
-r *Group*	Resets the real group for the current process. If you do not specify a *Group* parameter and the current real group is not the primary group, the **-r** flag removes the current real group and resets the real group to the original primary group. If you specify a *Group* parameter, this behaves identically to the **newgrp** command.
-	Re-initializes the group set of the session to its original login state.

Security

Access Control: This command should be a general user program. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the root user with the **setuid** (SUID) bit set.

Files Accessed:

Mode	Files
r	**/etc/passwd**
r	**/etc/group**

Auditing Events:

Item	Description
Event	**Information**

Item	Description
USER_SetGroups	realgroup, groupset

Examples

1. As user sah, you can display your current group membership and process group set, by entering:

   ```
   setgroups
   ```

   Output similar to the following appears:

   ```
   sah:

      user groups = staff,payroll
      process groups = staff,payroll
   ```

2. To add the finance group to the process group of the current session, enter:

   ```
   setgroups -a finance
   ```

3. To set your real group to finance, enter:

   ```
   setgroups finance,staff,payroll
   ```

   This sets finance as the real group. The staff and payroll groups make up the supplementary group list.

4. To delete the payroll group from the current process group set, enter:

   ```
   setgroups -d payroll
   ```

5. To change the process group set back to your default set, enter:

   ```
   setgroups -
   ```

   This resets your current session to its original state just after you log in.

Files

Item	Description
/usr/bin/setgroups	Contains the **setgroups** command.
/etc/group	Contains basic group attributes.
/etc/passwd	Contains basic user attributes.

setkst Command

Purpose

Sets the entries in the kernel security tables (KST).

Syntax

setkst [**-q**] [**-b** | **-l** | **-t** *table1*, *table2*,...]

Description

The **setkst** command reads the security databases and loads the information from the databases into the kernel security tables. By default, all of the security databases are sent to the KST. Alternatively, you can specify a specific database using the **-t** flag. If only the authorization database is the only one

you specified, the role and privileged command databases are updated in the KST because they are dependent on the authorization database.

The **setkst** command checks the tables before updating the KST. If any severe error in the database is found, the **setkst** command warns the user by sending message to the **stderr**, and exits without resetting the KST. If a minor error is found in the database, a warning message is displayed, and the entry is skipped.

The **setkst** command is only functional if the system is operating in enhanced Role Based Access Control (RBAC) mode. If the system is not in enhanced RBAC mode, the command displays an error message and ends.

Flags

Item	Description
-b	Loads the KST with the information that is stored in the backup binary file on the system. If information in the binary file cannot be loaded, the tables are regenerated from the security databases.
-l	Reads the `loglevel` attribute value from the `syslog` stanza in the `/etc/secvars.cfg` file and updates the `loglevel` attribute value to the kernel. The valid values for the `loglevel` attribute are as follows: `all`, `crit`, and `none`. Any invalid value for the `loglevel` attribute are ignored by the `setkst` command.
-q	Specifies quiet mode. Warning messages that occur are not displayed when the security databases are parsed.
-t *table1*, *table2*	Sends the specified security databases to the KST. The parameter for the **-t** flag is a comma-separated list of security databases. Values for this flag are as follows: **auth** Authorizations database **role** Role database **cmd** Privileged command database **dev** Privileged device database **dom** Domains **domobj** Domain objects

Security

The **setkst** command is a privileged command. Only users that have the following authorization can run the command successfully.

Item	Description
aix.security.kst.set	Required to run the command.

Files Accessed

File	Mode
/etc/security/authorizations	r

File	Mode
/etc/security/privcmds	r
/etc/security/privdevs	r
/etc/security/roles	r
/etc/security/domains	r
/etc/security/domobjs	r
/etc/secvars.cfg	r

Examples

1. To send all of the security databases to the KST, enter the following command:

```
setkst
```

2. To send the **role** and **privileged** command databases to the KST, enter the following command:

```
setkst -t role,cmd
```

3. To send the domain object and domain databases to the KST, enter the following command:

```
setkst -t domobj,dom
```

setmaps Command

Purpose

Sets terminal maps or code set maps.

Syntax

To use setmaps with no input or output map file designation, type the following:

setmaps [**-v**] [**-c** | **-h**]

To select a file from the default directory as the code set map file, type the following:

setmaps [**-v**] **-s** **-i** *MapName*

To select a designated file as the code set map file, type the following:

setmaps [**-v**] **-s** **-I** *File1*

To select a file from the default directory as the input or output terminal map file, type the following:

setmaps [**-v**] [**-D**] [**-k** *KeyName*] [**-d** *DirectoryPath*] { **-i** | **-o** } *MapName*

To select files from the default directory as the input or output terminal map files, type the following:

setmaps [**-v**] [**-D**] [**-d** *DirectoryPath*] **-t** *MapName*

To select a designated file as the input or output terminal map file, type the following:

setmaps [**-v**] [**-D**] [**-k** KeyName] { **-I** | **-O** } *File1*

To load the default terminal map file for later use, type the following:

setmaps [**-v**] [**-D**] [**-k** KeyName] [**-r**] **-l** *File2*

To load a designated terminal map file for later use, type the following:

setmaps [**-v**] [**-D**] [**-k** KeyName] [**-r**] **-L** *File1*

Description

> **Note:** If this command is run without root user authority, the code set map is not loaded, only debugged.

The **setmaps** command handles terminal and code set maps. The **-s** flag must be used for code set maps. The operating system uses input and output terminal maps to convert internal data representations to the ASCII characters supported by asynchronous terminals. If you enter the **setmaps** command with no flags, it displays the names of the current input and output terminal maps.

A terminal map is a text file containing a list of rules that associate a pattern string with a replacement string. This file normally resides in the **/usr/lib/nls/termmap** directory. The operating system uses an input map file to map input from the keyboard to an application and an output map file to map output from an application to the display.

Terminal mapping works as follows:

1. The system collects characters in a buffer until a pattern specified by a rule in the map file matches a substring in the buffer.
2. The system then constructs and returns the replacement string specified by the rule.

This processing continues with the remaining characters in the buffer.

The rules of a terminal map can test and change the state of the pattern processor. The state is identified by a single-byte character, conventionally a digit (0 through 9). The state is reset to 0, the initial state, whenever the system loads a new map or flushes the terminal input or output buffer (such as when it processes a KILL or INTR character or when a program issues an **ioctl** system call). A terminal map can use states to detect multibyte escape sequences, among other tasks. You can test for state *x* by specifying @x in a pattern. You can set the state to *x* by including @x in the replacement string.

The **setmaps** command, when using the **-s** flag, assigns a code set map to the standard input device. The operating system uses code set maps to determine the number of bytes of memory a character requires and the number of display columns it requires.

Flags

Item	Description
-c	Clears all mappings on this terminal.
-d *DirectoryPath*	Causes the *DirectoryPath* variable to be used as the path to the directory that contains the *MapName* variable. Specifying this flag and variable overrides the **/usr/lib/nls/termmap** directory.
-D	Produces a debug program printout of the specified map on the standard output device before loading the map. When using this to run the debug program on new maps, do not run with root user authority until the map is fully debugged to prevent the map from actually being loaded.
-h	Prints the usage information of the **setmaps** command (used with the **-v** flag for advanced users).
-i *MapName*	Selects the **/usr/lib/nls/termmap/MapName.in** file as the input map. When used with the **-s** flag, this flag selects the **/usr/lib/nls/csmap/MapName** file as the terminal code set map file.
-I *File1*	Selects the contents of the *File1* variable as the input map. The file specified by the *File1* variable can be either a full path name or a path name relative to the current working directory. When used with the **-s** flag, this flag selects the contents of the *File1* variable as the terminal code page map file.
-k *KeyName*	Associates the contents of the *KeyName* variable with the map being selected. This key name overrides the default key, which is normally set to the value of the *MapName* variable.

Item	Description
-l *File2*	Loads the **/usr/lib/nls/termmap/File2** file for later use. The *File2* variable includes the full path name and suffix (if any) of the map file. **Note:** You must have root user authority to specify this flag.
-L *File1*	Loads the specified map for later use. The *File1* variable includes the full path name and suffix (if any) of the map file. **Note:** You must have root user authority to specify this flag.
-o *MapName*	Selects the **/usr/lib/nls/termmap/MapName.out** file as the terminal output map.
-O *File1*	Selects the contents of the *File1* variable as the terminal output map. The *File1* variable includes the full path name and suffix (if any) of the map file.
-r	Forces reloading of the specified map, even if it is already loaded. Terminals using the old map continue to do so until they are logged off or until their maps are explicitly reset. If you do not specify this flag, a map is loaded only if it has not already been loaded into the kernel. **Note:** You must have root user authority to specify this flag.
-s	Treats any map as a code set map.
-t *MapName*	Selects the **/usr/lib/nls/termmap/MapName.in** file as the terminal input map and the **/usr/lib/nls/termmap/MapName.out** file as the terminal output map.
-v	Selects verbose output.

All maps loaded must have unique names. Use the **-k** flag to eliminate naming conflicts. Only the **-i**, **-o**, and **-t** flags implicitly add a suffix. Other flags specifying map names should include a suffix if appropriate. If a requested map name is already loaded in the kernel, that map is used even if the path information provided on the command line implies a different map.

To reset the code set map to its original state, the **/usr/lib/nls/csmap/sbcs** code set map should be used.

Examples

1. To display the current map settings for this terminal, enter:

```
setmaps
```

2. To clear all mapping for the current terminal, enter:

```
setmaps -c
```

3. To set up mapping (both input and output maps) for an `ibm3161-C` terminal, enter:

```
setmaps -t ibm3161-C
```

4. To load the `vt220` input map into the kernel as the `fred` map, enter:

```
setmaps -k fred -i vt220
```

5. To gather debug output for a new map called `bob` in a file called `bob.dump`, enter:

```
setmaps -D -L /tmp/bob > bob.dump
```

6. To set up a code set map conforming to the IBM-943 code page for this terminal, enter:

```
setmaps -s -i IBM-943
```

7. To set up a code set map from the file myEUC for this terminal, enter:

```
setmaps -s -I myEUC
```

Files

Item	Description
/**usr**/**bin**/**setmaps**	Contains the **setmaps** command.
/usr/lib/nls/termmap/*.in	Contains input map files.
/usr/lib/nls/termmap/*.out	Contains output map files.
/usr/lib/nls/csmap/sbcs	Contains code set map for a single-byte code page.
/usr/lib/nls/csmap/IBM-943	Contains code set map for the IBM-943 code page.
/usr/lib/nls/csmap/IBM-eucJP	Contains code set map for the IBM-eucJP code page.

setrunmode Command

Purpose

Sets the run mode of the system.

Syntax

setrunmode { **-c** | **-o** }

Description

The **setrunmode** command sets the run mode of the system. A run mode is either the CONFIGURATION mode or the OPERATIONAL mode.

Flags

Item	Description
-c	Specifies the CONFIGURATION Mode.
-o	Specifies the OPERATIONAL mode.

Security

Only users that have the following authorization can run the command successfully:

Item	Description
aix.mls.system.mode	Required to set the run mode.

Examples

1. To set the system in the CONFIGURATION mode, enter the following command:

```
setrunmode –c
```

2. To set the system in the OPERATIONAL mode, enter the following command:

```
setrunmode –o
```

Files

Item	Description
/usr/sbin/setrunmode	Contains the **setrunmode** command.

setsecattr Command

Purpose

Sets the security attributes of a command, a device, a privileged file, a process, or a domain-assigned object.

Syntax

setsecattr [**-R** *load_module*]{ **-c** | **-d** | **-p** | **-f** | **-o**} *Attribute = Value* [*Attribute = Value* ...] *Name*

Description

The **setsecattr** command sets the security attributes of the command, device, or process that is specified by the *Name* parameter. The command interprets the *Name* parameter as either a command, a device, a privileged file, or a process based on whether the **-c** (command), **-d** (device), **-f** (privileged file), or **-p** (process) flag is specified.

If you configure the system to one of the following values specified by the *Name* parameter, the system performs in the order that is specified by the **secorder** attribute of the corresponding database stanza in the **/etc/nscontrol.conf** file:

- Uses databases from multiple domains
- Sets security attributes for a privileged command
- Sets security attributes for a privileged device
- Sets security attributes for a privileged file
- Sets security attributes for a domain-assigned object

Only the first matching entry is modified. Duplicate entries from the remaining domains are not modified. Use the **-R** flag to modify the entry from a specific domain. If no matching entry is found in any of the domains, a new entry for the *Name* parameter is created in the first domain. Use the **-R** flag to add the entry to a specific domain.

To set a value for an attribute, specify the attribute name and the new value with the *Attribute=Value* parameter. To clear an attribute, specify the `Attribute=` for the *Attribute=Value* pair. To make incremental changes to attributes, whose values are lists, specify the *Attribute=Value* pairs as `Attribute=+Value`, or `Attribute=-Value`. If you specify the `Attribute=+Value`, the value is added onto the existing value for the attribute. If you specify the `Attribute=-Value`, the value is removed from the existing value for the attribute.

Flags

Item	Description
-c	Specifies that the security attributes of a command on the system are to be set. If the command name that you specified using the *Name* parameter is not in the privileged command database, a command entry is created in the **/etc/security/privcmds** privileged command database. If an attribute is being cleared and is the only attribute set for the command, the command is removed from the privileged command database. Modifications made to the privileged command database are not used until the database is sent to the kernel security tables using the **setkst** command.
-d	Specifies that the security attributes of a device on the system are to be set. If the device name you specify using the *Name* parameter is not in the privileged device database, a device entry is created in the **/etc/security/privdevs** privileged device database. If an attribute is being cleared and is the only attribute set for the device, the device is removed from the privileged device database. Modifications made to the privileged device database are not used until the database is sent to the kernel security tables using the **setkst** command.
-f	Specifies that the security attributes of a privileged file on the system are to be set. Changes requested through the *Attribute=Value* pairs are made in the **/etc/security/privfiles** privileged file database. If the specified file is not in the privileged file database, a file entry is created in the database. If an attribute is being cleared and is the only attribute set for the command, the command is removed from the privileged file database.
-o	Specifies that the security attributes of an object on the system are to be set. If the object name that you specified using the *Name* parameter is not in the domain object database, an object entry is created in the `/etc/security/domobjs` domain object database. If an attribute is being cleared and is the only attribute set for the object, the object entry is removed from the domain object database. Modifications made to the domain object database are not used until the database is sent to the kernel security tables using the **setkst** command.
-p	Specifies that the numeric process identifier (PID) of an active process on the system are to be set. Changes that you specify with the *Attribute=Value* pairs immediately affects the state of the specified active process. Modifications are not saved in a database.
-R *load_module*	Specifies the loadable module to use for security attribute modification.

Parameters

Item	Description
Attribute = Value	Sets the value of a security attribute for the object. The list of valid attribute names are dependent on the object type as specified using the **-c**, **-d**, **-p**, and **-o** flags. Use the following attributes for the privileged command database (**-c**) flag: **accessauths** Specifies access authorizations. Specifies a comma-separated list of authorization names. You can specify a total of sixteen authorization. A user with any of the authorizations that you specified can run the command. This attribute has three special additional values: ALLOW_OWNER, ALLOW_GROUP, and ALLOW_ALL that allows a command owner, a group, or all users to run the command without checking for access authorizations. **authprivs** Specifies authorized privileges. Specifies a list of authorizations and privilege pairs that grant additional privileges to the process. The authorization and its corresponding privileges are separated by an equal sign (=), individual privileges are separated by a plus sign (+), and authorization or privilege pairs are separated by a comma (,), as shown in the following examples: `auth=priv+priv+...,auth=priv+priv+...,...` You can specify a maximum of sixteen pairs of authorizations or privileges.Specifies roles, the users of which need to be authenticated before command can be executed successfully. Specifies a comma separated list of roles. Each role should be authenticated by different users such as no user can perform the authentication for more than one role at a time. **authroles** Specifies the user roles that need to be authenticated before the command can run successfully. If listing multiple roles, separate each role with a comma. For example: `authroles=so,isso` Each role must be authenticated by different users. For example, no one user can perform the authentication for more than one role. **innateprivs** Specifies the innate privileges. Specifies a comma-separated list of privileges that are assigned to the process when the command is run. **inheritprivs** Specifies inheritable privileges. Specifies a comma-separated list of privileges that are passed to child processes. **euid** Specifies the effective user ID to assume when the command is run. **egid** Specifies the effective group ID to assume when the command is run.

Item	Description
	ruid Specifies the real user ID to assume when the command is run. Only valid value is 0. This attribute value will be ignored if the command provides access to all users by specifying the special value ALLOW_ALL in its **accessauths** attribute. **secflags** Specifies the file security flags. Specifies a comma-separated list of security flags. Use the following values for this flag: **FSF_EPS** Causes the maximum privilege set to be loaded into the effective privilege set when the command is run. Use the following attributes for the privileged device database (**-d**) flag: **readprivs** Specifies a comma-separated list of privileges that a user or a process must have for read access to the device. You can specify a maximum of eight privileges. The user or process must have one of the listed privileges to read from the device. **writeprivs** Specifies a comma-separated list of privileges that a user or a process must have for write access to the device. You can specify a maximum of eight privileges. The user or process must have one of the listed privileges to write to the device.

Item	Description
	Use the following attributes for the privileged file (**-f**) flag: **readauths** Specify the read access authorizations. Specify a comma-separated list of authorization names. A user with any of the authorizations can read the file. **writeauths** Specify the write access authorizations. Specify a comma-separated list of authorization names. A user with any of the authorizations can read or write the file. Use the following attributes for the privileged process (**-p**) flag: **eprivs** Specify the effective privilege set. Specify a comma-separated list of privileges that are to be active for the process. The process might remove the privileges from this set and add the privileges from the maximum privilege set to its effective privilege set. **iprivs** Specifies the inheritable privilege set. Specifies a comma-separated list of privileges that are passed to child processes' effective and maximum privilege sets. The inheritable privilege set is a subset of the limiting privilege set. **mprivs** Specify a maximum privilege set. Specify a comma-separated list of privileges that the process can add to its effective privilege set. The maximum privilege set is a superset of the effective privilege set. **lprivs** Specify the limiting privilege set. Specify a comma-separated list of privileges that make up the maximum possible privilege set for a process. The limiting privilege set is a superset of the maximum privilege set. **uprivs** Specify the used privilege set. Specify a comma-separated list of privileges that are used during the life of the process. This set is mainly used by the **tracepriv** command.

Item	Description
	Use the following attributes for the domain-assigned object database (**-o**) flag: **domains** Specify a comma-separated list of domains the objects belong to. **conflictsets** Specify a comma-separated list of domains that are excluded from accessing the object. **objtype** Specify the type of the object. Valid values are device, netint, netport and file. **secflags** Specify the security flags for the object. Valid values are: • **FSF_DOM_ANY**: This value specifies that a process can access the object if it has any of the domains given in the domains attribute. • **FSF_DOM_ALL**: Specifies that a process can access the object only if it has all the domains as specified in the domains attribute. This is the default value if no secflags is specified. The **FSF_DOM_ANY** and **FSF_DOM_ALL** are mutually exclusive flags.
Name	Specify the object to modify. The *Name* parameter is interpreted according to the flags that you specify. One name must be indicated for processing at a time.

Security

The **setsecattr** command is a privileged command. It is owned by the root user and the security group, with the mode set to 755. You must have assume a role with at least one of the following authorizations to run the command successfully. For trusted process, the auditing system will not log any object auditing events for the respective process. However, users can capture events using event auditing.

Item	Description
aix.security.cmd.set	Required to modify the attributes of a command with the **-c** flag.
aix.security.device.set	Required to modify the attributes of a device with the **-d** flag.
aix.security.file.set	Required to modify the attributes of a device with the **-f** flag.
aix.security.proc.set	Required to modify the attributes of a process with the **-p** flag.
aix.security.dobject.set	Required to modify the attributes of a process with the **-o** flag.

File Accessed

Item	Description
File	**Mode**
/etc/security/privcmds	rw
/etc/security/privdevs	rw
/etc/security/privfiles	rw
/etc/security/domobjs	rw

Examples

1. To set an authorized privilege pair for the `/usr/sbin/mount` command, enter the following command:

```
setsecattr -c authprivs=aix.fs.manage.mount=PV_FS_MOUNT /usr/sbin/mount
```

2. To incrementally add the PV_AU_WRITE and PV_DAC_W privileges to the existing set of writing privileges for the `/dev/mydev` device, enter the following command:

```
setsecattr -d writeprivs=+PV_AU_WRITE,PV_DAC_W /dev/mydev
```

3. To set a read authorization for the **/etc/security/user** file, enter the following command:

```
setsecattr -f readauths=aix.security.user.change /etc/security/user
```

4. To incrementally remove the PV_DAC_R privilege from the effective privilege set of an active process, enter the following command:

```
setsecattr -p eprivs=-PV_DAC_R 35875
```

5. To set the access authorizations for the `/usr/sbin/mount` command in LDAP, enter the following command:

```
setsecattr -R LDAP -c accessauths=aix.fs.manage.mount /usr/sbin/mount
```

6. To set the domains on the network interface **en0**, enter the following command:

```
setsecattr –o domains=INTRANET,APPLICATION conflictsets=INTERNET
objtype=netint secflags=FSF_DOM_ANY en0
```

setsecconf Command

Purpose

Loads the system security flag settings into the kernel.

Syntax

setseconf { **-c** | **-o** } [*Attribute* = *Value* ...]

Description

The **setsecconf** command loads the system security flag settings into the kernel. If you specify any attributes, the values of these attributes are stored and used when the system is restarted. This command can change the setting of the flags for the CONFIGURATION and OPERATIONAL modes of the system, but these flags can be changed only when the system is in the CONFIGURATION mode.

Flags

Item	Description
-c	Specifies the CONFIGURATION mode.
-o	Specifies the OPERATIONAL mode.

Parameters

Item	Description
Attribute	You can specify the following attributes: **root** Specifies whether the root user can log in to the system. If enabled, the root user can log in to the system. If disabled, the root user cannot log in to the system. The value of this flag cannot be changed in Trusted AIX systems. For more information, see the information in the "Disabling the root user" topic. **tnet** Specifies the Advanced Security Network. If enabled, all of the data packets are labeled. **tlwrite** Specifies whether to enforce the write access checks on the integrity labels (TLs). If enabled, TLs are checked on write, remove, and rename operations. If disabled, TLs can be set, but are ignored on write access checks. **tlread** Specifies whether to enforce the read access checks on the integrity labels (TLs). If enabled, TLs are checked on read operations. If disabled, TLs can be set, but are ignored on read access checks. **traceauth** Specifies if authorization tracing is enabled. If enabled, the authorizations used in a process are traced and logged in a process credential. The **lssecattr** command is used to display used authorizations. If disabled, no authorizations are traced in a system. By default, this flag is disabled. This flag is only meaningful in the operational mode. **sl** Specifies whether to enforce the Mandatory Access Control (MAC) flag. If enabled, MAC is enforced. If not enabled, sensitivity labels (SLs) can be configured, but not used to determine the access to files and other objects. **tlib** Specifies whether to recognize and enforce the Trusted Computing Base (TCB). If enabled, the TCB flag on file system objects is recognized and enforced. If disabled, the TCB on objects is ignored and all objects are treated as if they are not TCB objects.
Value	Specifies a value that is either **enable** or **disable**.

Security

The **setsecconf** command is a privileged command. Only users that have the following authorization can run the command successfully:

Item	Description
aix.mls.system.config.write	Required to set the system configuration flags.

Exit Status

The **setsecconf** command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To turn on the trusted network and turn off the integrity read system flags for the CONFIGURATION mode run, enter the following command:

```
setsecconf -c tnet=enable tlread=disable
```

2. To turn on the integrity write system flag for the OPERATIONAL mode run, enter the following command:

```
setsecconf -o tlwrite=enable
```

Files

Item	Description
/usr/sbin/setsecconf	Contains the **setsecconf** command.

setsenv Command

Purpose

Resets the protected state environment of a user.

Syntax

setsenv [**-**] *NewEnvironment*

Description

The **setsenv** command resets your protected state environment while you are logged in. The protected state environment is defined as a set of variables. These variables are kept in the kernel and can be modified only by a **SETUINFO** system call. The **setsenv** command uses the variables specified by the *NewEnvironment* parameter. This parameter consists of *EnvironmentVariable*=*Value* definitions separated by a blank space. For information on environment variables, see **environment** File.

You cannot reset the following environment variables with the **setsenv** command:

Item	Description
NAME	Your last authenticated user name. This corresponds to the real user ID of the current process.
TTY	The name of the terminal on which you logged in. This corresponds to the initial controlling terminal for the process. This variable cannot be set for processes initiated without a *full login*. A full login is a login initiated by the **getty** command.
LOGNAME	The name under which you logged in, if the current session was started from a terminal login program. If the session was not started from a terminal, this variable is not set.

If you enter the **setsenv** command without any defined variables, it displays the current protected state. The **setsenv** command does not change the security characteristics of the controlling terminal.

When you run the **setsenv** command, it replaces your current shell and gives you a new one. The command replaces your shell regardless of whether it completed successfully or not. For this reason, the command does not return error codes.

Flags

Item	Description
-	Reinitializes the environment as if the user had just logged in to the system. Otherwise, the environment is not changed.

Security

Access Control: This command should be a standard user program. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the root user with the **setuid** (SUID) bit set.

Files Accessed:

Mode	File
r	**/etc/environment**
r	**/etc/security/environ**

Auditing Events:

Event	Information
USER_SetEnv	new environment string

Examples

1. To display the current environment variables, enter:

   ```
   setsenv
   ```

2. To add the PSEUDO=tom protected environment variable, enter:

   ```
   setsenv PSEUDO=tom
   ```

   This example sets a user name for the **PSEUDO** protected environment variable.

Files

Item	Description
/usr/bin/setsenv	Specifies the path to the **setsenv** command.
/etc/environment	Contains environment information for each user.
/etc/security/environ	Contains privileged environment information for each user.

setsyslab Command

Purpose

Sets the minimum and maximum sensitivity labels of the system.

Syntax

setsyslab

Description

The **setsyslab** command sets the system minimum sensitivity label (SL), maximum SL, minimum integrity label (TL), and maximum TL. The values of the SL and TL are taken from the **/etc/security/enc/LabelEncodings** label encodings file.

Security

The **setsyslab** command is a privileged command. Only users that have the following authorization can run the command successfully:

Item	Description
aix.mls.system.label.write	Required to set system labels.

Files Accessed:

Item	Description
Mode	**File**
r	**/etc/security/enc/LabelEncodings**

Examples

1. To set system labels, enter the following command:

```
setsyslab
```

Files

Item	Description
/usr/sbin/setsyslab	Contains the **setsyslab** command.
/etc/security/enc/LabelEncodings	System default label encodings file.

settime Command

Purpose

Updates access and modification times of a file.

Syntax

settime [[*MMddhhmm*[*yy*]] | [**-f** *ReferenceFile*]] *File* ...

Description

settime updates the argument files with the current access and modification times by default. The file is not created if it does not exist. The **settime** command silently continues its operation if the file does not exist.

Note: Any dates beyond and including the year 2038 are not valid for the **settime** command.

Flags

Item	Description
-f *ReferenceFile*	Use the corresponding time of *ReferenceFile* instead of the current time.

Parameters

Item	Description
MMddhhmm[*yy*]	Time is specified for the **settime** command in the format *MMddhhmm* or *MMddhhmmyy*, where *MM* is a two-digit representation of the month, *dd* is a two-digit representation of the day of the month, *hh* is a two-digit representation of the hour, *mm* is a two-digit representation of the minute, and *yy* is a two-digit representation of the year.
File	Specifies the name of a file or a space separated list of files.

Exit Status

0
The command completed successfully.

>0
An error occurred.

The return code from **settime** is the number of specified files for which the times could not be successfully modified.

Examples

1. To update the access and modification times of the file "infile" to the current time, enter:

```
settime infile
```

2. To update the access and modification times of "infile" to be the same as "reffile", enter:

```
settime -f reffile infile
```

3. To update the access and modification times of multiple files, enter:

```
settime file1 file2 file3
```

4. To update the access and modification times of a file to April 9th 2002 with time 23:59, enter:

```
settime 0409235902 infile
```

Files

Item	Description
/usr/bin/settime	Contains the **settime** command.

settxattr Command

Purpose

Sets the security attributes.

Syntax

settxattr { **-f** | **-m** | **-p** | **-q** | **-s** } *Attribute* = *Value* ... *Name*

Description

The **settxattr** command sets Trusted AIX security attributes of the file, process, shared memory, message queue, or semaphore that is specified by the *Name* parameter. The command interprets the *Name* parameter as either a file, a process, a shared memory, a message queue, or a semaphore based on whether the **-f** (file), **-p** (process), **-m** (shared memory), **-q** (message queue), or the **-s** (semaphore) flag is specified.

To set a value for an attribute, specify the attribute name and the new value with the *Attribute=Value* parameter. All of the attributes are applied to extended attributes (EA) of the file system for file system objects and user credentials for processes.

Flags

Item	Description
-f	Specifies the security attributes of a file. The *Name* parameter specifies the path to this file on the system.
-p	Specifies the security attributes of a process. The *Name* parameter specifies the numeric process identifier (PID) of an active process on the system. Changes requested through the *Attribute=Value* pairs immediately affect the state of the specified active process.
-m	Specifies the security attributes of a shared memory. The *Name* parameter specifies the numeric shared memory identifier on the system.
-q	Specifies the security attributes of a message queue. The *Name* parameter specifies the numeric message queue identifier on the system.
-s	Specifies the security attributes of a semaphore. The *Name* parameter specifies the numeric semaphore identifier on the system.

Parameters

Item	Description
Attribute = Value	Specifies the value of a security attribute for the object. The list of valid attribute names are dependent on the object type as specified through the **-f**, **-m**, **-p**, **-q**, and **-s** flags. Use the following file security attributes for the (**-f**) flag: **sl** Specifies the Sensitivity Label (SL). Specifies the SL to apply labels for regular files. This attribute is not valid for directories, devices, or terminal devices (TTYs). **maxsl** Specifies the Maximum Sensitivity Label. The value that you specify for this attribute must dominate the existing Minimum Sensitivity Label. This attribute is valid only for directories, devices, and TTYs. **minsl** Specifies the Minimum Sensitivity Label. The value that you specify for this attribute must be dominated by the existing Maximum Sensitivity Label. This attribute is valid only for directories, devices, and TTYs. **tl** Specifies the Integrity Label. Specify this attribute to apply labels to a file. **secflags** Specifies the Trusted AIX file security flags. Specify this attribute as a comma-separated list of security flags. You can specify the following flags: • FSF_APPEND • FSF_AUDIT • FSF_MAC_EXMPT • FSF_TLIB • FSF_TLIB_PROC Use the following process security attributes for the **-p** flag: **effsl** Effective Sensitivity Label. Specify this attribute to apply labels on an active process. The **effsl** attribute must dominate the existing Minimum Sensitivity Label. **maxcl** Maximum Sensitivity Clearance Label. Specify this attribute to apply labels on an active process. The **maxsl** attribute must dominate the existing Effective Sensitivity Label. **mincl** Minimum Sensitivity Clearance Label. Specify this attribute to apply label on an active process. The **mincl** attribute must be dominated by the existing Effective Sensitivity Label. **efftl** Effective Integrity Label. Specify this attribute to apply labels on an active process. The **efftl** attribute must dominate the existing Minimum Integrity Label. **maxtl** Maximum Integrity Label Specify this attribute to apply labels on an active process. The **maxtl** attribute must dominate the existing Effective Integrity Label. **mintl** Minimum Integrity Label. Specify this attribute to apply labels on an active process. The **mintl** attribute must be dominated by the existing Effective Integrity Label. Use the following security attributes for the message queue (**-q**) flag, the shared memory (**-m**) flag, and the semaphore (**-s**) flag: **sl** Specifies the Sensitivity Label (SL). Specify this attribute to apply labels to a message queue, shared memory, or semaphore object. **tl** Specifies the Integrity Label (TL). Specify this attribute to apply labels to a message queue, shared memory, or semaphore object.

Security

The **settxattr** command is a privileged command. It is owned by the root user and the security group, with the mode set to 755. To run the command successfully, users must have at least one of the following authorizations:

Item	Description
aix.mls.label.sl.upgrade	Required to assign an SL higher than the existing SL of filesystem objects.
aix.mls.label.tl.upgrade	Required to assign a TL higher than the existing TL of filesystem objects.
aix.mls.label.sl.downgrade	Required to assign an SL lower than the existing SL of filesystem objects.
aix.mls.label.tl.downgrade	Required to assign a TL lower than the existing TL of filesystem objects.

Item	Description
aix.mls.proc.sl.upgrade	Required to assign an effective SL higher than the existing effective SL of the process.
aix.mls.proc.tl.upgrade	Required to assign an effective TL higher than the existing effective TL of the process.
aix.mls.proc.sl.downgrade	Required to assign an effective SL lower than the existing effective SL of the process.
aix.mls.proc.tl.downgrade	Required to assign an effective TL lower than the existing effective TL of the process.
aix.mls.label.outsideaccred	Required to assign labels outside the accreditation range.

File Accessed:

Item	Description
Mode	**File**
r	**/etc/security/enc/LabelEncodings**

Examples

1. To apply labels to a regular file called `regfile`, enter the following command:

   ```
   settxattr -f sl=SECRET tl=SECRET regfile
   ```

2. To apply labels to a directory called `dirname`, enter the following command:

   ```
   settxattr -f maxsl="TS ALL" minsl="SEC ALL" tl=TS dirname
   ```

3. To apply labels to a message queue IPC object with the `0` message queue ID, enter the following command:

   ```
   settxattr -q sl=SECRET tl=SECRET 0
   ```

4. To apply labels to a shared memory IPC object with the 3145728 shared memory ID, enter the following command:

   ```
   settxattr -m sl=SECRET tl=SECRET 3145728
   ```

5. To apply labels to a semaphore IPC object with the three shared memory IDs, enter the following command:

   ```
   settxattr -s sl=SECRET tl=SECRET 3
   ```

setuname Command

Purpose

Sets the node name of the system.

Syntax

setuname [**-t**] **-n** *Node*

Description

The **setuname** command is used to set the node name of the system. The **-n** option must be specified. Only users with root authority can set the node name. The change can be made temporary by using the **-t** option. The node name will be modified only on the current running kernel if a temporary change is requested. The nodename set temporarily will not persist after a reboot. Without the **-t** option the node name is changed permanently in the ODM database.

Flags

Item	Description
-n *Node*	Specifies that the node name has to be changed. This option is required. *Node* is the primary node name for the host. This can be the UUCP communications network name for the system.
-t	Temporary change. No attempt will be made to make the change permanent. The original name will be restored after reboot.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Examples

1. To temporarily change the node name to "orion", enter:

```
setuname -t -n orion
```

2. To permanently change the node name to "orion", enter:

```
setuname -n orion
```

Files

Item	Description
/usr/bin/setuname	Contains the setuname command.

sh Command

Purpose

Invokes the default shell.

Syntax

Refer to the syntax of the **ksh** command. The **/usr/bin/sh** file is linked to the Korn shell.

Description

The **sh** command invokes the default shell and uses its syntax and flags. The shell linked to the **/usr/bin/sh** path is the default shell. The standard configuration of the operating system links the **/usr/bin/sh** path to the Korn shell.

Flags

Refer to the flags for the Korn shell (**ksh** command).

Files

Item	Description
/usr/bin/sh	Contains the **sh** command.

shconf Command

Purpose

Manages the system hang detection parameters.

Syntax

shconf **-d**

shconf **-R** **-l** *Name*

shconf {**-D** [**-O**] | **-E** [**-O**]} [**-H**] **-l** *Name*

shconf **-l** *Name* [**-a** *Attribute=Value*] ...

Description

The **shconf** command is used to display or specify the parameters of the priority problem detection and lost I/O detection.

For the priority problem, the user can specify five actions described below and for each action, the user can specify the priority level to check, the time out while no process or thread executes at a lower or equal priority, the terminal device for the warning action, and the getty action:

Item	Description
pp_cmd	Launches a command specified by the path parameter.
pp_errlog	Logs an error in error log.
pp_login	Launches a getty at the highest priority on the serial line specified by the terminal device parameter (term).
pp_reboot	Reboots the system.
pp_warning	Displays a warning message on the console specified by the terminal device parameter (term).

For lost I/O, the user can specify the actions listed below and **errlog**, which is automatic when lost I/O detection is enabled. There is a unique timeout which applies to all enabled actions.

Item	Description
lio_warning	Displays a warning message on the console specified by the terminal device parameter (term).
lio_reboot	Creates a system dump and reboots the system.

Note: The `shconf` command only supports the `tty` and `console` terminal types.

Flags

Item	Description
-d	Displays if priority problem detection and lost I/O detection are enabled or not.
-R	Restore the default values for a specified name of detection.
-a *Attribute=Value*	Specifies the attribute value pairs used for changing specific attribute values.
-D	Displays the default values for a specified name of detection.
-E	Displays the effective values for a specified name of detection.
-H	Displays the headers above the column output. When used together, the **-O** flag overrides the **-H** flag.
-l *Name*	Specifies the detection name.
-O	Displays all attribute names separated by colons and, on the second line, displays all the corresponding attribute values separated by colons. The attribute values are current values when the **-E** flag is also specified and default values when the **-D** flag is specified. This flag cannot be used with the **-a** flag.

Files

Item	Description
/usr/sbin/shconf	Contains the **shconf** command.

shell Command

Purpose

Executes a shell with the user's default credentials and environment.

Syntax

shell

Description

The **shell** command re-initializes a user's login session. When the command is given, the port characteristics of the process's controlling terminal are reset and all access to the port is revoked. The **shell** command then resets the process credentials and environment to the defaults established for the user and executes the user's initial program. All credentials and environment are established according to the login user ID of the invoking process.

If the **shell** command is invoked on the trusted path and the user's **tpath** attribute in the **/etc/security/user** file does not have a value of **always**, the trusted environment of the terminal is not maintained.

Note: The **shell** command does not reset the login ID of the user.

Security

Access Control: The command should be **setuid** to the root user to reset the user's process credentials, and grant execute (x) access to all users. The command should have the **trusted computing base** attribute.

Files Accessed:

Mode	File
r	**/etc/passwd**
r	**/etc/group**
r	**/etc/security/audit/config**
r	**/etc/security/environ**
r	**/etc/security/limits**
r	**/etc/security/user**

Auditing Events:

Event	Information
USER_Shell	portname

Examples

To re-initialize your session to your default credentials and environment after using the trusted shell (**tsh**), enter:

```
shell
```

Files

Item	Description
/usr/bin/shell	Contains the **shell** command.
/etc/security/user	Contains the extended attributes of users.
/etc/passwd	Contains user IDs.
/etc/group	Contains group IDs.
/etc/security/audit/config	Contains the audit configuration information.
/etc/security/environ	Defines the environment attributes for users.
/etc/security/limits	Defines process resource limits for each user.

show Command

Purpose

Shows messages.

Syntax

show [**+***Folder*] [**-draft** | *Messages*] [**-header** | **-noheader**] [**-showproc** *CommandString* | **-noshowproc**]

Description

The **show** command displays the contents of messages. If standard output is not a display, the **show** command lists each message with a one-line header and two separation lines. By default, the **show** command displays the current message in the current folder.

The **show** command invokes a listing program to create the list. The default listing program is **/usr/bin/more**. You can define your own default with the `showproc:` entry in your **$HOME/.mh_profile** file. If you set the `showproc:` entry to `mhl`, the **show** command calls an internal **mhl** routine instead of the **mhl** command. You can also specify the program to perform a listing in the *CommandString* parameter of the **-showproc** flag.

The **show** command passes any flags it does not recognize to the listing program. Thus, you can specify flags for the listing program, as well as for the **show** command.

If the `Unseen-Sequence:` entry is present in your **$HOME/.mh_profile** file and the entry is not empty, the **show** command removes each of the messages shown from each sequence named by the profile entry. If several messages are specified, the last message shown becomes the current message.

Flags

Item	Description
-draft	Shows the *UserMhDirectory***/draft** file if it exists.
+*Folder*	Specifies a folder. The current folder is the default.
-header	Displays a one-line description of the message being shown. The description includes the folder name and message number. If you show more than one message, this flag does not produce message headers. The **-header** flag is the default.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
Messages	Specifies the messages to show. You can specify several messages, a range of messages, or a single message. Use the following references to specify messages: ***Number*** Number of the message. ***Sequence*** A group of messages specified by the user. Recognized values include: **all** All messages in a folder. **cur or . (period)** Current message. This is the default. **first** First message in a folder. **last** Last message in a folder. **next** Message following the current message. **prev** Message preceding the current message.

Item	Description
-noheader	Prevents display of a one-line description of each message.
-noshowproc	Uses the **/usr/bin/cat** command to perform the listing. This is the default.
-showproc *CommandString*	Uses the specified command string to perform the listing.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Current-Folder:`	Sets the default current folder.
`Path:`	Specifies the user's MH directory.
`showproc:`	Specifies the program used to show messages.
`Unseen-Sequence:`	Specifies the sequences used to keep track of the unseen messages.

Examples

1. To display the contents of the current message in the current folder one screen at a time, enter:

   ```
   show
   ```

   If the message continues for more than one screen, press the Enter key until you have read the entire message.
2. To see the contents of all the messages in the current folder, enter:

   ```
   show all
   ```

   If the messages continue for more than one screen, press the Enter key until you have read all the messages.
3. To see the contents of message 5 in the `meetings` folder, enter:

   ```
   show  +meetings 5
   ```
4. To see the contents of all the messages belonging to the `weekly` sequence in the `meeting` folder, enter:

   ```
   show  +meeting weekly
   ```

Files

Item	Description
$HOME/.mh_profile	Specifies the MH user profile.
*UserMhDirectory***/draft**	Contains the current message draft.
/usr/bin/show	Contains the **show** command.

showmount Command

Purpose

Displays a list of all clients that have remotely mounted file systems.

Syntax

/usr/bin/showmount [**-a**] [**-d**] [**-e**] [*Host*]

Description

The **showmount** command displays a list of all clients that have remotely mounted a file system from a specified machine in the *Host* parameter. This information is maintained by the **mountd** daemon on the *Host* parameter. This information is saved in the **/etc/rmtab** file in case the server crashes. The default value for the *Host* parameter is the value returned by the **hostname** command.

Note: If a client crashes, its entry will not be removed from the list until the client reboots and starts the **umount -a** command.

Note: The **showmount** command returns information maintained by the **mountd** daemon. Because NFS Version 4 does not use the **mountd** daemon, **showmount** will not return information about version 4 mounts.

Flags

Item	Description
-a	Prints all remote mounts in the format *HostName*:*Directory*, in which *HostName* is the name of the client and *Directory* is a directory pathname that has been remotely mounted.
-d	Lists only directories that have been remotely mounted by clients.
-e	Prints the list of exported directories.

Examples

1. To display a list of all remote directories that are mounted by a host, enter the following command:

   ```
   /usr/bin/showmount -a zeus
   ```

   In this example, the `showmount` command produces a list of all of the remote directories mounted by the clients on the host machine named `zeus`.
2. To display a list of only the directories that are mounted by a client on the host, enter the following command:

   ```
   /usr/bin/showmount -d athena
   ```

   In this example, the `showmount` command produces a list of all remote directories mounted by the client machines on the host named `athena`.
3. To print a list of all directories that are exported from a machine, enter the following command:

   ```
   /usr/bin/showmount -e zeus
   ```

   In this example, the `showmount` command produces a list of all remote directories that are exported by the host machine named `zeus` except the ones that are exported only with NFS version 4.

Files

Item	Description
/etc/rmtab	Contains information about the current state of all exported directories.
/etc/xtab	Lists currently exported directories.

shutacct Command

Purpose

Turns off processing accounting.

Syntax

/usr/sbin/acct/shutacct [**"***Reason***"**]

Description

The **shutacct** command turns off process accounting and calls the **acctwtmp** command to add a record stating the reason to the **/var/adm/wtmp** file. The **shutacct** command is invoked by the **shutdown** command.

> **Note:** It is necessary to place quotation marks around the *Reason* value in the **/var/adm/wtmp** file.

Variables

Item	Description
Reason	Specifies the reason for accounting system shutdown. This value is optional.

Security

Access Control: This command should grant execute (x) access only to members of the adm group.

Files

Item	Description
/usr/sbin/acct	The path to the accounting commands.
/var/adm/wtmp	The login and logout history file.

shutdown Command

Purpose

Ends system operation.

Syntax

shutdown [**-d**] [**-F**] [**-h**] [**-i**] [**-k**] [**-l**] [**-m**] [**-p**] [**-r**] [**-t** *mmddHHMM* [*yy*]] [**-u**] [**-v**] [**+***Time* [*Message*]]

Description

The **shutdown** command halts the operating system. Only a user with root user authority can run this command. During the default shutdown, users are notified (by a **wall**command) of the impending system shutdown with a message. However, shutdown is not complete until the user receives a shutdown completion message. Do not attempt to restart the system or turn off the system before the shutdown completion message is displayed; otherwise, file system damage can result.

> **Note:** The `halt completed` message is not displayed on the tty from which shutdown is invoked if it is connected to the system through a multiport adapter.

As shutdown time approaches, warning messages are displayed on the terminals of all users on the system.

After the specified number of seconds (60 by default), the system stops the accounting and error logging processes and writes an entry to the error log. The **shutdown** command then runs the **killall** command to end any remaining processes and runs the **sync** command to flush all memory resident disk blocks. Finally, it unmounts the file systems and calls the **halt** command.

> **Note:** Users who have files open on the node that is running the **shutdown** command, but who are not logged in to that node, are not notified about the shutdown.

If you request a complete halt to the operating system, the **shutdown** command stops all processes, unmounts all file systems, and calls the **halt** command.

The system administrator can place local customized shutdown procedures in a shell script named **/etc/rc.shutdown**. This script runs at the beginning of the shutdown if it exists. If the script runs but fails with a non-zero return code, the shutdown stops.

> **Attention:** If you are bringing the system down to maintenance mode, you must run the **shutdown** command from the **/** (root) directory to ensure that it can cleanly unmount the file systems.
>
> **Note:** By default, if issued on models having a power supply capable of software control, the **shutdown** command turns off the system.

Flags

Item	Description
-d	Brings the system down from a distributed mode to a multiuser mode.
-F	Does a fast shutdown, bypassing the messages to other users and bringing the system down as quickly as possible. The **+***Time* [*Message*] options are ignored if the **-F** flag is specified.
-h	Halts the operating system completely; same as the **-v** flag.
-i	Specifies interactive mode. Displays interactive messages to guide the user through the shutdown.
-k	Allows the administrator to broadcast the **shutdown** warning messages *without* causing the system to shut down. When the **-k** flag is used, no other shutdown activity occurs except for sending messages. For example, no processes are killed, no activity is logged in **/etc/shutdown.log** if the **-l** flag is specified, and if an **/etc/rc.shutdown** script exists it does not run.
-l	Creates/appends the **/etc/shutdown.log** file that contains information about the filesystems, daemons, user login, licensing services, network interfaces being brought down. The file may be used for diagnostic and debugging purposes in the event of shutdown failures. **Note:** Ensure that there is enough disk space for the **shutdown** command to log the entries while using this flag.

Item	Description
-m	Brings the system down to maintenance (single user) mode.
-p	Halts the system without a power down. This is used by uninterruptible power supply (UPS). **Note:** The **-p** flag will have no effect if used in combination with flags not requiring a permanent halt. Power will still be turned off if other operands request a delayed power-on and reboot
-r	Restarts the system after being shutdown with the **reboot** command.
-t *mmddHHMM* [*yy*]	Shuts down the system immediately and then restarts the system on the date specified by *mmddHHMM* [*yy*] where ***mm*** Specifies the month. ***dd*** Specifies the day. ***HH*** Specifies the hour. ***MM*** Specifies the minute. ***yy*** Specifies the year. The **shutdown -t** flag cannot be used with the **-v** or **-h** option. **Note:** This option is only supported on systems that have a power supply which automatically turns power off at shutdown and an alarm to allow reboot at a later time. Systems without this capability may hang or may reboot immediately after shutdown.
-u	This flag is used by diagnostics to update the flash-memory and reboot.
-v	Halts the operating system completely.

Parameters

Item	Description
+*Time*	Specifies the time at which the **shutdown** command stops the system. An immediate shutdown is indicated by the word now displayed on the screen. A future time can be specified in one of two formats: +number or hour:minute. The first form brings the system down in the specified number of minutes and the second brings the system down at the time of day indicated (as a 24-hour clock). If the *Message* parameter is specified, the *Time* parameter must also be specified.
Message	Specifies the message

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To turn off the machine, enter:

   ```
   shutdown
   ```

   This shuts down the system, waiting 1 minute before stopping the user processes and the **init** process.
2. To give users more time to finish what they are doing and bring the system to maintenance mode, enter:

   ```
   shutdown  -m +2
   ```

   This brings the system down from multiuser mode to maintenance mode after waiting 2 minutes.

Files

Item	Description
/usr/sbin/shutdown	Contains the **shutdown** command.

sisraidmgr Command

Purpose

Uses and maintains a Peripheral Component Interconnect-X (PCI-X) SCSI Redundant Array of Independent Disks (RAID) controller.

Syntax

`sisraidmgr` [`-A` -l hdisk# [-z *pdisk*] [-f]]

`sisraidmgr` [`-B` -l adptr# -b ioa_opt [-r raid_level]]

`sisraidmgr` [`-C` [-r raid_level -s stripe_size (in KB) -z *pdisk_list*]]

`sisraidmgr` [`-D` -l adptr# [-d *hdisk* | -e *serial_num*]]

`sisraidmgr` [`-F` [-z *pdisk_list*]]

`sisraidmgr` [`-H` [-z *pdisk_list*]]

`sisraidmgr` [`-I` [-z *pdisk_list*]]

`sisraidmgr` [`-L` -l adptr# [-a display_opt [-v sisarray_opt -p pdisk_opt -j jbod_opt]]]

`sisraidmgr` [`-L` -l hdisk# [-a display_opt [-v sisarray_opt -p pdisk_opt]]]

`sisraidmgr` [`-L` -l pdisk# [-p pdisk_opt]]

`sisraidmgr` [`-M` -l adptr# -o *cmd_opt*]

`sisraidmgr` [`-P` -z drive_list { *pdisks* | *hdisks* }]

`sisraidmgr` [`-R` -z *pdisk_list*]

`sisraidmgr` [`-S` -l adptr#]

`sisraidmgr` [`-U` -z *pdisk_list*]

`sisraidmgr` [`-W` -l adptr# -o *cmd_opt*]

`sisraidmgr` [`-X` -l adptr# -o *cmd_opt*]

`sisraidmgr` [`-Y` -l hdisk# [-x cmd_opt]]

Description

The `sisraidmgr` command is used to create, delete, and maintain RAID arrays on a PCI-X SCSI RAID controller.

Attention: See the *PCI-X SCSI RAID Controller Reference Guide for AIX* and become familiar with the storage management concepts before you run the **sisraidmgr** command.

Attention: The *System Management Interface Tool (SMIT)* **smit pxdam** fast path is the preferred method to manage a PCI-X SCSI RAID Controller.

Attention: Service tasks require special training and must not be performed by nonservice personnel.

Flags

Item	Description
`-A`	Adds a device to an existing array. The performance is not optimal when you use this option because the included device does not contain parity, and the data is not restriped. **-l lname** The logical name of the array. **-z *pdisks*** The drives to be included. **-f** The option to force the include operation in the situation where the disks to be included might not be known; that is, they might be 0.
`-B`	Lists information about what the adapter supports. **-l lname** The logical name of the adapter. **-b ioa_support_opt** **1** Displays supported RAID levels for the **lname** option. This is the default option. **2** Displays supported stripe size for the **lname** and **raid_level** options. **3** Displays the minimum number of devices for the **raid_level** option. **4** Displays the maximum number of devices for the **raid_level** option. **5** Displays the minimum multiple number of devices for the **raid_level** option. **-r raid_level** Shows supported stripe sizes for this RAID level.
`-C`	Creates a RAID array. **-r raid_level** { 0, 5, or 10 (RAID 1+0) } **-s stripe_size (in KB)** If not specified, the default (64 KB) is used. **-z *pdisk_list*** Lists pdisks to include in the new array. For example, 'pdisk2 pdisk3 pdisk4' must be connected to the same adapter.
`-D`	Deletes a RAID array. **-l lname** The logical name of the adapter. **-d *hdisk*** The name of the array to be deleted. **-e *serial_num*** The serial number of the array to be deleted. Use this option only if the array name is unknown.
`-F`	Formats the pdisks for recovery. (format 522-byte formatted disks). **-z *drive_list*** A list of pdisks to format.
`-H`	Adds a hot spare device. **-z pdisk_list** A list of pdisks to be made hot spare devices.

Item	Description
-I	Removes a hot spare device. **-z pdisk_list** A list of pdisks to be removed from being hot spare devices.
-L	Lists advance function information. **-l lname** The device for which information is displayed. It can be a RAID adapter (*sisioa0*), a RAID array (*hdisk8*), or a physical disk (*pdisk5*). **-a display_opt** **0** Displays all configuration information for the **lname** option. This is the default option. **1** Displays only the logical device information for the **lname** option. **2** Displays only the physical device information for the **lname** option. **-v sisarray_opt** **0** Displays all arrays. This is the default. **1** Displays only arrays that are candidates for the **Delete Array** option. **2** Displays only arrays that are candidates for the **Rsync Protection** option. **3** Displays only arrays that are candidates for including additional devices. **4** Displays only ODM arrays that have no adapter information. **-p pdisk_opt** **0** Displays all pdisks. This is the default. **1** Displays only pdisks that are candidates for the **Prepare** option. **2** Displays only pdisks that are candidates for the **Start RAID** option. **3** Displays only pdisks that are candidates for the **Add Hot Spare** option. **4** Displays only pdisks that are candidates for the **Remove Hot Spare** option. **5** Displays only pdisks that are candidates to be added to an existing array. **6** Displays only pdisks that are candidates for the **Rebuild** option. **7** Displays only pdisks that are candidates for the **Recovery Format** option. **8** Displays only ODM pdisks that have no adapter information. **9** Displays only pdisks that are candidates for the **Unprepare** option (522 - 512). **10** Displays only pdisks that, if prepared, would be candidates to be added to an existing array. **-j jbod_opt** **0** Displays no JBOD hdisks. This is the default option. **1** Displays all JBOD hdisks. **2** Displays only JBOD hdisks that are candidates for the **Prepare** option (512 - 522).

Item	Description
-M	Maintains the rechargeable battery. **-l lname** The logical name of the adapter. **-o cmd_option** The command options follow: **0** Displays rechargeable battery information. **1** Forces a rechargeable battery error. **2** Starts caching after concurrent battery replace. **3** Queries candidates for concurrently starting batteries.
-P	Prepares devices; that is, creates array candidates physical disks. **-z drive_list** A list of either JBOD hdisks, pdisks, or both to become an array candidate.
-Q	Sets or Clears pdisk error suppression attributes. **-z pdisk_list** A list of pdisks for attributes to be applied or cleared. **-o cmd_option** A 1-byte hexadecimal string that specifies which error suppression bits to turn on or off.
-R	Rebuilds devices; that is, reconstructs a degraded array. **-z pdisk_list** A list of pdisks to be rebuilt.
-S	Displays the adapter link status. **-l lname** The logical name of the adapter.
-U	Creates stand-alone physical disks. **-z drive_lists** A list of pdisks to be formatted to stand-alone disks.
-W	Reclaims cache storage. **-l lname** The logical name of the adapter. **-o cmd_option** The command options follow: **0** Queries to determine whether a reclaim operation is needed. **1** Queries to determine whether permission for unknown data loss is needed. **2** Performs reclaim cache storage. **3** Performs reclaim cache storage and allows unknown data loss.
-X	Changes adapter assignment. **-l lname** The logical name of the adapter. **-o cmd_option** The command options follow: **0** Displays only. **1** Preferred as primary adapter. **2** No preferred operating preferences. **3** Preferred as primary adapter. This value runs the **cfgmgr** command.
-Y	Resynchronizes array protection. **-l lname** The logical name of the array.

Exit Status

This command returns the following exit values:

Item	Description
0	The **sisraidmgr** command completed the operation successfully.
>0	The **sisraidmgr** command detected an error.

Security

Privilege Control: Only the root user and members of the system group should have execute (x) access to this command.

Examples

1. Display usage information:

```
# sisraidmgr -h
```

2. Views disk array configuration on a PCI-X SCSI RAID controller named sissas0:

```
# sisraidmgr -L -l sissas0 -j3
```

3. Prepares 512 byte formatted drives (hdisk3 and hdisk4) for use in a disk array:

```
# sisraidmgr -P -z 'hdisk3 hdisk4'
```

4. Creates a RAID 0 array with stripe size of 256K on the prepared disks (pdisk2 and pdisk5):

```
# sisraidmgr -C -r 0 -s 256 -z 'pdisk2 pdisk5'
```

5. Deletes the RAID array hdisk3 on controller sissas0:

```
# sisraidmgr -D -l sissas0 -d hdisk3
```

Files

Item	Description
`/usr/bin/sisraidmgr`	Contains the `sisraidmgr` command.

sissasraidmgr Command

Purpose

Maintains and uses a Serial Attached SCSI (SAS) Redundant Array of Independent Disks (RAID) controller.

Syntax

`sissasraidmgr` -A -l hdisk# [-z *pdisk* [-f]]

`sissasraidmgr` -B -l adptr# -b ioa_opt [-r raid_level]

`sissasraidmgr` -C [-r raid_level -s stripe_size (in KB) -z *pdisk_list*]

`sissasraidmgr` -D -l adptr# [-d *hdisk* | -e *serial_num*]

`sissasraidmgr` -E -l adptr# [-d *hdisk* -o *cmd_opt*]

`sissasraidmgr` -F -z *pdisk_list*

`sissasraidmgr` -G -l hdisk# -r raid_level [-s stripe_size (in KB) -z *pdisk_list*]

`sissasraidmgr -H [ -z pdisk_list ]`

`sissasraidmgr -I [ -z pdisk_list ]`

`sissasraidmgr -J -z drive_list -o cmd_opt`

`sissasraidmgr -L -l adptr# [ -a display_opt [ -v sisarray_opt -p pdisk_opt -j jbod_opt ] ]`

`sissasraidmgr -L -l hdisk# [ -a display_opt [ -v sisarray_opt -p pdisk_opt ] ]`

`sissasraidmgr -L -l pdisk# [ -p pdisk_opt ]`

`sissasraidmgr -M -l adptr# -o cmd_opt`

`sissasraidmgr -P -z drive_list ( pdisks | hdisks )`

`sissasraidmgr -Q -z pdisks } [ -o cmd_opt ]`

`sissasraidmgr -R -z pdisk_list`

`sissasraidmgr -S -l adptr# [ -o cmd_opt ]`

`sissasraidmgr -T -l adptr# [ -o cmd_opt ]`

`sissasraidmgr -T -l device# [ -o cmd_opt ]`

`sissasraidmgr -U -z pdisk_list`

`sissasraidmgr -W -l adptr# -o cmd_opt`

`sissasraidmgr -X -l adptr# -o cmd_opt`

`sissasraidmgr -Y -l hdisk#`

`sissasraidmgr -Z -l adptr# -o cmd_opt`

Description

The `sissasraidmgr` command is used to create, delete, and maintain RAID arrays on a Peripheral Component Interconnect-X (PCI-X) or PCI Express (PCIe) SAS RAID controller.

Attention: See the *Power Systems SAS RAID Controllers for AIX* reference guide and become familiar with the storage management concepts before you run the **sissasraidmgr** command.

Attention: The *System Management Interface Tool (SMIT)* **smit sasdam** fast path is the preferred method to manage a SAS RAID controller.

Attention: Service tasks require special training and must not be performed by nonservice personnel.

Flags

Item	Description
-A	Add a device to an existing array. The performance is not optimal when using this option because the included device does not contain parity, and the data is not restriped. **-l lname** The logical name of the array. **-z *pdisks*** The drives to be included. **-f** The option to force the include operation in the situation where the disks to be included might not be known; that is, they might be 0.

Item	Description
-B	Lists information about what the adapter supports. **-l lname** The logical name of the adapter. **-b ioa_support_opt** **1** Displays supported RAID levels for the **lname** option. This is the default option. **2** Displays supported stripe size for the **lname** and **raid_level** option. **3** Displays the minimum number of devices for the **raid_level** option. **4** Displays the maximum number of devices for the **raid_level** option. **5** Displays the minimum multiple number of devices for the **raid_level** option. **6** Displays supported migration RAID levels for the **lname** option. **7** Displays supported migration stripe size for the **lname** and **raid_level** options. **8** Displays the minimum number of migration include devices for the **raid_level** option. **9** Displays the maximum number of migration include devices for the **raid_level** option. **10** Displays minimum multiple migration include devices for the **raid_level** option. **11** Displays the minimum percentage of the total array capacity that is allowed in one tier for the **raid_level** option. **12** Displays the minimum number of devices per tier for the **raid_level** option. **-r raid_level** Shows supported stripe sizes for this RAID level.
-C	Creates a RAID array. **-r raid_level** { 0, 5, 6, 10 (RAID 1+0), 5T2, 6T2, or 10T2} **-s stripe_size (in KB)** Specifies the stripe size. If not specified, the default (64 KB) is used. **-z *pdisk_list*** Lists pdisks to include in the new array. For example, 'pdisk2 pdisk3 pdisk4' must be connected to the same adapter.
-D	Deletes a RAID array. **-l lname** The logical name of the adapter. **-d *hdisk*** The name of the array to be deleted. **-e *serial_num*** The serial number of the array to be deleted. Use this option only if the array name is unknown.
-E	Manages HA access characteristics of a RAID array. **-l lname** The logical name of the adapter. **-d *hdisk*** The name of the array. **-o *cmd-opt*** The command options follow: **1** Displays the current and preferred HA access states. **2** Sets the preference to optimized on the **lname** option. **3** Sets the preference to nonoptimized on the **lname** option. **4** Clears preferences.

Item	Description
-F	Formats the pdisks for recovery (format RAID formatted disks). **-z *drive_list*** A list of pdisks to format.
-G	Migrates the RAID array to a new RAID level. **-l lname** The logical name of the array. **-r raid_level** { 0, 5, 6, 10 (RAID 1+0), 5T2, 6T2, or 10T2} **-s stripe_size (in KB)** Specifies the stripe size. If not specified, the default (64 KB) is used. **-z pdisk_list** A list of pdisks to be included in the new array, if any.
-H	Adds a hot spare device. **-z pdisk_list** A list of pdisks to be made hot spare devices.
-I	Removes a hot spare device. **-z pdisk_list** A list of pdisks to be removed from being hot spare devices.
-J	Optimizes JBOD workload. **-z drive_list** A list of JBOD hdisks to optimize. **-o *cmd_opt*** The command options: **1** Optimizes for the I/O response time. **2** Optimizes for the I/O operation per second.

Item	Description
-L	Lists advance function information. **-l lname** The device for which information is displayed. It can be a RAID adapter (*sisioa0*), a RAID array (*hdisk8*), or a physical disk (*pdisk5*). **-a display_opt** **0** Displays all configuration information for the **lname** option. This is the default option. **1** Displays only logical device information for the **lname** option. **2** Displays only physical device information for the **lname** option. **3** Displays only the physical device information for the **lname** option that is not under an adapter in the secondary mode. **-v sisarray_opt** **0** Displays all arrays. This is the default. **1** Displays only arrays that are candidates for the **Delete Array** option. **2** Displays only arrays that are candidates for the **Rsync Protection** option. **3** Displays only arrays that are candidates for including additional devices. **4** Displays only ODM arrays that have no adapter information. **5** Displays only arrays that are candidates for migration to a new RAID level. **-p pdisk_opt** **0** Displays all pdisks. This is the default. **1** Displays only pdisks that are candidates for the **Prepare** option. **2** Displays only pdisks that are candidates for the **Start RAID** option. **3** Displays only pdisks that are candidates for the **Add Hot Spare** option. **4** Displays only pdisks that are candidates for the **Remove Hot Spare** option. **5** Displays only pdisks that are candidates to be added to an existing array. **6** Displays only pdisks that are candidates for the **Rebuild** option. **7** Displays only pdisks that are candidates for the **Recovery Format** option. **8** Displays only ODM pdisks that have no adapter information. **9** Displays only pdisks that are candidates for the **Unprepare** option. **10** Displays only pdisks that, if prepared, would be candidates to be added to an existing array. **11** Displays only pdisks under their main path (primary or only path). **12** Displays only pdisks that are candidates for including during the migration of an existing array. **-j jbod_opt** **0** Displays no JBOD hdisks. This is the default. **1** Displays all JBOD hdisks. **2** Displays only JBOD hdisks that are candidates for the **Prepare** option. **3** Displays all JBOD devices.

Item	Description
-M	Maintains rechargeable battery. **-l lname** The logical name of the adapter. **-o cmd_option** The command options follow: **0** Displays rechargeable battery information. **1** Forces a rechargeable battery error. **2** Starts caching after concurrent battery replacement. **3** Queries candidates for concurrently starting batteries..
-P	Prepares devices; that is, creates array candidates physical disks. **-z drive_list** A list of either JBOD hdisks, pdisks, or both to become an array candidate.
-Q	Sets or clears pdisk error suppression attributes. **-z pdisk_list** A list of pdisks for attributes to be applied or cleared. **-o *cmd_option*** A 1-byte hexadecimal string that specifies which error suppression bits to turn on or off.
-R	Rebuilds devices; that is, reconstructs a degraded array. **-z pdisk_list** A list of pdisks to be rebuilt. **-o *cmd_opt*** Command option for adapter: **0** Displays HA link status. This is the default. **1** Displays HA and AWC link status.
-S	Displays the adapter link status. **-l lname** The logical name of the adapter.
-T	Displays SAS path information for the adapter. **-l lname** The logical name of the adapter. **-o *cmd_opt*** The command option for the adapter follow: **0** Displays the summary path window. This is the default. **1** Displays all path information for all attached devices. **2** Graphically displays paths for all attached devices. **16** Displays I/O Adapter SAS addresses.
-T	Displays SAS path information for the attached devices. **-l lname** The logical name of the device (pdisk or hdisk). **-o *cmd_opt*** The command option for the adapter follow: **0** Graphically displays path information for device. **1** Displays path information data for a selected device.
-U	Creates stand-alone physical disks. **-z drive_lists** A list of pdisks to be formatted to stand-alone disks.

Item	Description
-W	Reclaims cache storage. **-l lname** The logical name of the adapter. **-o cmd_option** The command options follow: **0** Queries to determine whether a reclaim operation is needed. **1** Queries to deternmine whether permission for unknown data loss is needed. **2** Performs reclaim cache storage. **3** Performs reclaim cache storage, and allows unknown data loss.
-X	Changes adapter assignment. **-l lname** The logical name of the adapter. **-o cmd_option** The command options follow: **0** Displays only **1** Preferred as primary adapter. **2** No preferred operating preferences. **3** Preferred as primary adapter. This value runs the **cfgmgr** command. **4** Displays AWC preferred role information. **10** Sets the dual initiator mode to be the default. **11** Sets the dual initiator mode to the JBOD HA single path. **256** Clears HA access states. **512** Preserves HA Access states. **1024** Enables the default behavior of the IOA cache. **2048** Disables the IOA cache. **Note:** The clear, preserve, enable, and disable options can be paired (ORed) with options 1, 2, or 3, or they can be used as stand-alone options.
-Y	Resynchronizes array protection. **-l lname** The logical name of the array.
-Z	Shows the SAS controller physical resources. **-l lname** The logical name of the adapter. **-o *cmd_option*** The command options follow: **0** Shows the physical location. This is the default. **1** Shows physical information. **Note:** Enter the same options as the **-L** flag to filter the output.

Exit Status

This command returns the following exit values:

Item	Description
0	The **sissasraidmgr** command completed the operation successfully.
>0	The **sissasraidmgr** command detected an error.

Security

Privilege Control: Only the root user and members of the system group should have execute (x) access to this command.

Examples

1. Displays usage information:

```
# sissasraidmgr -h
```

2. Views disk array configuration on a SAS RAID controller named sissas0:

```
# sissasraidmgr -L -l sissas0 -j3
```

3. Prepares JBOD drives (hdisk3 and hdisk4) for use in a disk array:

```
# sissasraidmgr -P -z 'hdisk3 hdisk4'
```

4. Creates a RAID 0 array with a stripe size of 256 KB on the prepared disks (pdisk2 and pdisk5):

```
# sissasraidmgr -C -r 0 -s 256 -z 'pdisk2 pdisk5'
```

5. Deletes the RAID array hdisk3 on controller sissas0:

```
# sissasraidmgr -D -l sissas0 -d hdisk3
```

6. Optimizes the RAID array hdisk1 on sissas2, which is also the primary controller:

```
# sissasraidmgr -E -l sissas2 -d hdisk1 -o 2
```

7. Optimizes hdisk2 on sissas3, which is the secondary controller:

```
# sissasraidmgr -E -l sissas2 -d hdisk2 -o 3
```

8. Show SAS physical paths to a drive pdisk3:

```
# sissasraidmgr -T -l pdisk3 -o 1
```

Files

Item	Description
`/usr/bin/sissasraidmgr`	Contains the `sissasraidmgr` command.

size Command

Purpose

Displays the section sizes of the Extended Common Object File Format (XCOFF) object files.

Syntax

size [**-d** | **-o** | **-x**] [**-f**] [**-V**] [**-X** {**32** | **64** | **32_64** | **d64** | any}] [*File ...*]

Description

The **size** command writes to standard output the number of bytes required by all sections, along with their sum for each XCOFF file. If the **-f** flag is specified, the section name follows the section size.

Note: When no file is passed as an input to the **size** command, the **a.out** file is considered as the default.

Flags

The output is in decimal notation unless you change the output with the following flags:

Item	Description
-d	Writes in decimal notation.
-f	Writes the section name in parenthesis following the section size.
-o	Writes in octal notation.
-x	Writes in hexadecimal notation.
-X *mode*	Specifies the type of object file **size** should examine. The *mode* must be one of the following: **32** Processes only 32-bit object files **64** Processes only 64-bit object files **32_64** Processes both 32-bit and 64-bit object files **d64** Examines discontinued 64-bit XCOFF files (magic number == U803XTOCMAGIC). **any** Processes all of the supported object files. The default is to process 32-bit object files (ignore 64-bit objects). The *mode* can also be set with the **OBJECT_MODE** environment variable. For example, **OBJECT_MODE=64** causes **size** to process any 64-bit objects and ignore 32-bit objects. The **-X** flag overrides the **OBJECT_MODE** variable.
-V	Prints the version number of the **size** command.

Examples

1. To display the size of the **a.out** file in decimal, enter:

   ```
   size
   ```

   This displays the size in bytes of the executable **a.out** file. The size of each section of the object file is given, followed by the total:

   ```
   3720 + 1752 + 4152 = 9624
   ```

2. To display the size of an object file in octal, enter:

   ```
   size -o driver.o
   ```

   This displays the size of the **driver.o** object file in octal.

3. To display the size of several object files in hexadecimal, enter:

   ```
   size -x *.o
   ```

This displays in hexadecimal the size of each file ending with **.o** in the current directory.

skctl Command

Purpose

Handles alterations in the storage protection keys attributes.

Syntax

skctl [**-D**]

skctl [**-u**] <*nukeys/off*>] [**-k** *on/off/default*]

skctl [**-v** [*now*|*default*|*boot*]

Description

The **skctl** command is a privileged command used on a system that supports storage protection keys. The **skctl** command can change the number of user-space storage keys, disable user-space storage keys, enable/disable kernel storage key state, and display the default, current, and next boot storage keys attributes.

Note: You must run **/usr/sbin/bosboot** command after changing the storage keys attributes, and then reboot the system for the change to take effect.

Flags

Item	Description
-u	Alters the number of user-space keys or disables user-space keys. The flag must be **off** or a number between **2** and the maximum number of hardware storage keys.
-k	Enables/disables kernel keys.
-v	Displays the default, current, and next boot storage keys attributes.
-D	Resets the storage protection keys attributes to default.

skulker Command

Purpose

Cleans up file systems by removing unwanted files.

Syntax

skulker

Description

> **Attention:** Because the **skulker** command is run by a root user, and its whole purpose is to remove files, it has the potential for unexpected results. Before installing a new **skulker** command, test any additions to its file removal criteria by running the additions manually using the **xargs -p** command. After you have verified that the new **skulker** command removes only the files you want removed, you can install it.

The **skulker** command is used for periodically purging obsolete or unneeded files from file systems. Candidate files include files in the **/tmp** directory, files older than a specified age, and the following file types: ***.bak**, **a.out**, **core**, proof, galley, **...***, **ed.hup**, and files that are more than one day old.

The **skulker** command is normally invoked daily, often as part of an accounting procedure run by the **cron** command during off-peak periods. Modify the **skulker** command to suit local needs following the patterns shown in the distributed version. Local users should be made aware of the criteria for automatic file removal.

The **find** command and the **xargs** command form a powerful combination for use in the **skulker** command. Most file selection criteria can be expressed conveniently with **find** expressions. The resulting file list can be segmented and inserted into **rm** commands using the **xargs** command to reduce the overhead that would result if each file were deleted with a separate command.

slattach Command

Purpose

Attaches serial lines as network interfaces.

Syntax

/usr/sbin/slattach *TTYName* [*BaudRate DialString* [*DebugLevel*]]

Description

The **/usr/sbin/slattach** command assigns a TTY line to a network interface.

The **slattach** command is run by the **/etc/rc.net** file during system startup to automatically configure any Serial Line Internet Protocol (SLIP) network interfaces defined by the System Management Interface Tool (SMIT). SLIP interfaces can also be configured manually as shown in the examples section.

For a directly connected SLIP interface, broken connections are retried automatically without manual intervention. For a SLIP interface connected by modem, broken connections must be manually redialed. If a user supplies a dial string in the **slattach** command line, the user must re-enter the command and dial string to restore a broken connection.

To detach the interface, run the **ifconfig** *Interface* **down** command after terminating the **slattach** command. The *Interface* parameter is the name shown by the **netstat** command.

If configuring a slip interface from the command line, the **/usr/sbin/ifconfig** command must be invoked for the slip interface with the appropriate parameters and the slip tty line discipline must also be available in order for this command to succeed. To check if the slip tty line discipline is already loaded, run the command `strinfo -m | grep slip`. If no output is shown, the module has not yet been loaded. Load the module by issuing the command `strload -m /usr/lib/drivers/slip`.

Note:

1. After the SLIP interface has been configured with **ifconfig**, any user who has permission on the TTY may issue the **slattach** command.
2. You must configure the tty devices used by the **slattach** command before establishing a connection. You may also need to make an entry for the tty device in the BNU **/usr/lib/uucp/Devices** file.
3. Sample shell script, **/usr/sbin/slipcall**, provides a simplified interface for invoking **slattach** and connecting to remote systems. **slipcall** is useful for connecting to dial-in SLIP networks that require a user to login before activating the SLIP tty line discipline. The basic configuration of **slipcall** will connect to other operating systems with **sliplogin** configurations and derive the local and remote internet addresses and network mask assigned by the called system. It then configures the local interface with the remote system's specified values.

Parameters

Item	Description
BaudRate	Sets the speed of the connection. The default speed is 9600.

Item	Description
DebugLevel	Sets the level of debug information desired. A number from 0 through 9 may be specified. A value of 0 specifies no debug information; a value of 9 specifies the most debug information. The default value is 0.
DialString	Specifies a string of expect/respond sequences using the Basic Networking Utility (BNU)/AIX to AIX Copy Program (UUCP) **chat** syntax.
TTYName	Specifies a TTY line. This string is in the form `ttyxx` or `/dev/ttyxx`.

Examples

1. To attach the SLIP network interface to the `tty1` port with a direct connection, issue the following command:

   ```
   slattach /dev/tty1
   ```

   This command attaches `tty1` to a network interface to be used by the SLIP.
2. To attach the SLIP network interface to `tty1` using a modem connection, issue the following command:

   ```
   slattach /dev/tty1 9600 '""AT OK \pATF1 OK \pATDT34335 CONNECT""'
   ```

Files

Item	Description
/etc/uucp/Devices	Lists definitions of devices used for remote connections.

sleep Command

Purpose

Suspends execution for an interval.

Syntax

sleep *Seconds*

Description

The **sleep** command suspends execution of a process for at least the interval specified by the *Seconds* parameter. The amount of time specified in the *Seconds* parameter can range from 1 to **MAXINT** (2,147,483,647) seconds.

Exit Status

This command returns the following exit values:

Item	Description
0	The execution was successfully suspended for at least *Seconds* seconds, or a **SIGALRM** signal was received.
>0	An error occurred.

Examples

1. To run a command after a certain amount of time has passed, enter:

```
(
echo  "SYSTEM SHUTDOWN IN 10 MINUTES!" | wall
sleep 300; echo "SYSTEM SHUTDOWN IN 5 MINUTES!" | wall
sleep 240; echo "SYSTEM SHUTDOWN IN 1 MINUTE!" | wall
sleep 60; shutdown
)&
```

   This command sequence warns all users 10 minutes, 5 minutes, and 1 minute before the system is shut down.

2. To run a command at regular intervals, enter:

```
while true
do
 date
 sleep 60
done
```

   This shell procedure displays the date and time once a minute. To stop it, press the Interrupt key sequence.

slibclean Command

Purpose

Removes any currently unused modules in kernel and library memory.

Syntax

slibclean

Description

The **slibclean** command unloads all object files with load and use counts of 0. It can also be used to remove object files that are no longer used from both the shared library region and in the shared library and kernel text regions by removing object files that are no longer required.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/usr/sbin/slibclean	Contains the **slibclean** command.

sliplogin Command

Purpose

Converts a standard-input terminal line into a Serial Line Internet Protocol (SLIP) link to a remote host.

Syntax

sliplogin [*LoginName*]

Description

The **sliplogin** command configures a standard-input terminal line into a Serial Line Internet Protocol (SLIP) link to a remote host; that is, the command attaches a serial line network interface.

Note: User requires root authority to attach a network interface.

The **sliplogin** command searches the **/etc/slip.hosts** file for a loginname entry that matches the value of the *LoginName* parameter. If a matching entry is found, **sliplogin** configures the line appropriately for SLIP (that is, for 8-bit transparent input/output) and converts it to SLIP line discipline. Then, **sliplogin** invokes the applicable login shell script which initializes the SLIP interface with the local and remote Internet Protocol (IP) addresses, netmask, and optional arguments associated with the loginname entry in the **/etc/slip.hosts** file.

The usual initialization script file is **/etc/slip.login**. However, in order to accommodate special initialization needs of a particular host, a script file named **/etc/slip.login.*userlogin*** (where *userlogin* corresponds to the loginname entry in the **/etc/slip.hosts** file) can be created. The **sliplogin** command uses the **/etc/slip.login.*userlogin*** script file when it exists, instead of the **/etc/slip.login** script file.

To deinitialize the SLIP interface, the **sliplogin** command uses either the **/etc/slip.logout** script file or the **/etc/slip.logout.*userlogin*** script file, if one of them exists, with preference given to the latter. The **/etc/slip.logout** script file is given the same arguments as the **/etc/slip.login** script file; the **/etc/slip.logout.*userlogin*** script file is given the same arguments as the **/etc/slip.login.*userlogin*** script file. In its default form, the **/etc/slip.logout** script file deletes all routes through the network interface for the specified SLIP unit. Additional processes to be done when the SLIP interface is disconnected can be added to either logout script file.

Note:

1. The interface automatically deactivates when the remote connection terminates or when the **sliplogin** command dies.
2. Use the **slattach** command to access a remote system that has a SLIP link configured. Use the sample shell script file **/usr/sbin/slipcall** to invoke the **slattach** command with the proper parameters needed to call a remote system and configure the local interface with the appropriate values assigned by the remote system.
3. When using **sliplogin** as a user's login shell on a tty device, then this tty port used needs to be enabled for login. (This differs from the configuration when using **slattach** instead of **sliplogin** as a SLIP server process.

/etc/slip.hosts File

The **/etc/slip.hosts** file is the configuration file containing the names of preconfigured sliplogin users and the IP addresses to be assigned to the local and remote interface when the user logs in. **sliplogin** searches this file for matching *LoginName* entries. This file has the following format:

- Comments (lines starting with a #) and blank lines are ignored.
- Other lines must start with a *loginname* argument, and the fields should contain whatever is appropriate for the **slip.login** file that is executed for that name.
- Arguments are separated by white space and follow normal sh(1) quoting conventions. However, the *loginname* argument cannot be quoted. Usually lines have the following form:

  ```
  loginname local_address remote_address netmask opt_args
  ```

  where *local_address* and *remote_address* are the IP host names or addresses of the local and remote ends of the SLIP line, and *netmask* is the appropriate IP netmask. These arguments are passed directly to the **ifconfig** command. *Opt_args* are optional arguments used to configure the line.

- This implementation of **sliplogin** allows the **/etc/slip.hosts** file to contain multiple entries for a single SLIP user with differing addresses. This enables multiple SLIP interfaces to be activated by the **sliplogin** command for the same user name. When user entries are retrieved from the **/etc/slip.hosts** file, only entry addresses meeting the following criteria are selected.

  The entry is ignored if a `slip.hosts` entry specifies a local address which is already in use on another non-SLIP interface on the local system.

  The entry is ignored if the remote address specified in an **/etc/slip.hosts** entry is already in use on any other interface.

/etc/slip.login File

The **/etc/slip.login** or **/etc/slip.login.*userlogin*** file is the setup script invoked by the **sliplogin** command to initialize the user's network interface. The **/etc/slip.login.*userlogin*** file is invoked if it exists, where the value of the *LoginName* parameter of the **sliplogin** command corresponds to a loginname entry in the **/etc/slip.hosts** file. If this file cannot be accessed, the **/etc/slip.login** file is invoked instead. The login script file contains the following parameters:

Item	Description
slipunit	Specifies the unit number of SLIP interface assigned to this line. For example, 0 for sl0 (sl0 is s, lowercase L, zero.)
speed	Specifies the speed of the line.
args	Specifies the arguments from the **/etc/slip.hosts** file entries, in order, starting with *loginname*.

/etc/slip.logout File

The **/etc/slip.logout** or **/etc/slip.logout.*userlogin*** file is the setup script invoked by **sliplogin** to deinitialize the user's network interface. The **/etc/slip.logout.*userlogin*** file is invoked if it exists, where the value of the *LoginName* parameter of **sliplogin** corresponds to a loginname entry in the **/etc/slip.hosts** file. If this file cannot be accessed, the **/etc/slip.logout** file is invoked instead.

Flags

Item	Description
</dev/ttyx	Redirects the command to the **ttyx** device if the user is already logged into a **tty** device and wants to configure their terminal as a SLIP line.

Parameters

Item	Description
LoginName	Specifies the desired login name. The default is the current login name.

Examples

The normal use of the **sliplogin** command is to create an **/etc/passwd** entry for each legal, remote SLIP site with **sliplogin** as the shell for the entry. For example,

```
foo:!:2010:1:slip line to foo:/tmp:/usr/sbin/sliplogin
```

An entry must then be added to the **/etc/slip.hosts** file. The entry should resemble the following example:

```
foo 1.1.1.1 1.1.1.2 0xffffff00 normal
```

where *loginname* = `foo`, *local_address* = `1.1.1.1`, *remote_address* = `1.1.1.2`, *netmask* = `0xffffff00`, and `opt_args = normal.` (The optional argument `normal` indicates which SLIP mode to activate.)

Diagnostics

The **sliplogin** command logs various information to the system log daemon (**syslogd**). The messages are listed here, grouped by severity levels.

Error Severity	
Message	**Description**
ioctl (TCGETS): *reason*	The **ioctl** subroutine failed to get the line parameters for the reason indicated.
ioctl (TCSETS): *reason*	The **ioctl** subroutine failed to set the line parameters for the reason indicated.
ioctl (TIOCGETD): *reason*	The **ioctl** subroutine failed to get the current **tty** discipline for the reason indicated.
/etc/slip.hosts: *reason*	The **/etc/slip.hosts** file could not be opened for the reason indicated.
Check of flags for interface *xxx* failed. Errno is reason.	An attempt to check the status of the indicated interface to avert possible addressing conflicts failed for the reason indicated in the **errno** global variable.
Access denied for user - no /etc/slip.login[.***userlogin***] file.	No **/etc/slip.login** or **/etc/slip.login.*userlogin*** script file could be found.
Access denied for user - no /etc/slip.hosts entries available.	No loginname entry in the **/etc/slip.hosts** file matched the *LoginName* value specified in the command.
Access denied - getlogin returned 0.	The user issuing the **sliplogin** command does not have a password entry in the **/etc/passwd** file.
Logout script failed: exit status *xxx* from /etc/slip.logout[.***userlogin***]	An attempt to run the **/etc/slip.logout** or **/etc/slip.logout.*userlogin*** script file failed with the indicated exit status.
No SLIP interface for ttyx. Errno is reason.	No SLIP interface could be located for the **ttyx** device for the reason indicated in the **errno** global variable. Try either running the **ifconfig slx up** command or using SMIT to add a network interface for the **tty** device.
Open /dev/null: *reason*	An attempt to open the **/dev/null** device failed for the reason indicated.
/etc/slip.logout file not found	The **/etc/slip.logout** file could not be located.
sliplogin: cannot add SLIP discipline to ttyx	No SLIP interface exists for the **ttyx** device. Try either running the **ifconfig slx up** command or using SMIT to add a network interface for the **tty** device.
SLIP discipline removal from tty failed. Errno is reason.	An attempt to remove the SLIP discipline from the **tty** device failed for the reason indicated in the **errno** global variable.

Error Severity *(continued)*	
Message	**Description**
tcgetattr: *reason*	An attempt to read the current attributes of the **tty** device failed for the reason indicated.
userlogin login failed: exit status *xxx* from /etc/slip.login[.*userlogin*]	A system call to execute the **/etc/slip.login** or **/etc/slip.login.*userlogin*** script file failed with the indicated exit status.

Information Severity	
Message	**Description**
Attaching SLIP unit *xxx* for *userlogin* on ttyx.	The **sliplogin** command found a loginname entry in the **/etc/slip.hosts** file that matched the *LoginName* value specified in the command, invoked the applicable **/etc/slip.login** or **/etc/slip.login.*userlogin*** file, and is now attaching the indicated network interface.
Closed *userlogin* SLIP unit *xxx* (signal)	The indicated SLIP unit for the indicated *userlogin* was closed because the **sliplogin** command terminated due to a signal.

Notice Severity	
Message	**Description**
Attaching SLIP unit *xxx* for *userlogin*.	The indicated SLIP unit has been successfully attached for the indicated *userlogin*.

Files

Item	Description
/etc/slip.hosts	The configuration file that contains the names of preconfigured sliplogin users and the IP addresses to be assigned to the local and remote interface when the user logs in.
/etc/slip.login or **/etc/slip.login.*userlogin***	The setup script invoked by the **sliplogin** command to initialize the user's network interface.
/etc/slip.logout or **/etc/slip.logout.*userlogin***	The setup script invoked by the **sliplogin** command to deinitialize the user's network interface.

slocal Command

Purpose

Processes incoming mail.

Syntax

slocal [**-verbose** | **-noverbose**] [**-debug**]

Description

The **slocal** command performs a set of actions each time a message is sent to the user. The **slocal** command is not started by the user. The **slocal** command is called by the **sendmail** command.

The **sendmail** command starts the **slocal** command upon encountering the following line in the **$HOME/.forward** files:

```
/usr/lib/mh/slocal
```

For each incoming message, the **slocal** command performs the actions specified in the **.maildelivery** file. If the **slocal** command cannot find the **$HOME/.maildelivery** file, the **slocal** command uses the **/etc/mh/maildelivery** default file. If the delivery request fails, the **slocal** command delivers the message to the **/usr/mail/$USER** file.

Flags

Item	Description
-debug	Provides information for debugging.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.
-noverbose	Does not display information as the system executes commands in the **.maildelivery** file. This flag is the default.
-verbose	Displays information as the system executes commands in the **.maildelivery** file.

Files

Item	Description
/usr/lib/mh/mtstailor	Contains MH command definitions.
/etc/mh//.maildelivery	Contains the default MH instructions for local mail delivery.
$HOME/.maildelivery	Provides the user with MH instructions for local mail delivery.
$HOME/.forward	Contains either the line that starts the **slocal** command or a path to forward mail.
/etc/mh/mh_profile	Contains parameters that customize the MH package.

slp_srvreg Command

Purpose

Manages a service location protocol (SLP) service agent.

Syntax

slp_srvreg **-t** *servicetype* **-u** *URL* [**-a** *attribute*] [**-l** *lifetime*] [**-s** *scopes*] [**-T** *IPAddress*] [**-p** *port*] [**-U**] [**-v**] [**-b** *debuglevel*] [**-6**]

slp_srvreg **-d** *URL* [**-s** *scopes*] [**-T** *IPAddress*] [**-p** *port*] [**-v**] [**-b** *debuglevel*] [**-6**]

slp_srvreg **-D** [**-v**] [**-b** *debuglevel* [**-p** *port*]

slp_srvreg **-k** [**-v**] [**-b** *debuglevel*]

slp_srvreg **-h**

Description

The **slp_srvreg** command manages the service location protocol (SLP) service agent. The **slp_srvreg** command is used to register services for a specified URL with an attribute list in a given scope. The *servicetype* specified with the **-t** flag will override any service type expressed in the URL with the **scheme** service.

To register a service, use the **slp_srvreg** command with the **-u** flag to specify the URL to register.

To deregister a service, use the **slp_srvreg** command with the **-d** flag to specify the URL to deregister.

For both registration and deregistration, use the **-T** flag to specify an IP address to which the registration request will be sent to. If you specify the IP address of the local host (e.g. 127.0.0.1) or if you do not use the **-T** flag, the registration of the service URL is processed locally.

You must specify the **slp_srvreg** command with the **-D** flag to run **slp_srvreg** as a daemon. The **slp_srvreg** command with the **-k** flag kills the **slp_srvreg**.

Restriction: Do not run more than one **slp_srvreg** daemon on the same machine.

Use the **-p** flag to make the **slp_srvreg** agent running as daemon listen on a user specified port instead of the default port number 427. When registering or de-registering with a port specified in the **-p** flag of the **slp_srvreg**, only the service agents or directory agents listening on this port will accept the registration or deregistration.

Requirement: The **-t** and **-u** flags are mandatory for registration.

SLP clients must not expect the SLP service agent to return attribute values using the same case as used during the registration. For example, if a client registers a service with *attribute*=**true**, a query for the attribute might respond with *attribute*=**TRUE**. Any client seeking this information must handle the attribute in a case-insensitive manner.

Note: When the command **slp_srvreg -D -b** debuglevel is used with a debuglevel greater than zero, then **slp_srvreg** is not run as a daemon.

Flags

Item	Description
-a *attribute*	Specifies a comma-separated list of attributes for the services to be registered.
-d *URL*	Specifies the URL for the service to be deregistered.
-D	Specifies to run as a daemon.
-k	Kills the **slp_srvreg** daemon.
-l *lifetime*	Specifies the time after which the service registration needs be renewed. The value of the *lifetime* attribute is specified in number of seconds.
-p *port*	Specifies the port to listen to when running as a daemon. If you do not specify the **-p** flag, the default port 427 is used. If the **slp_srvreg** daemon is listening on a port other than the standard port, the user agent uses this flag to send the new registration data to the correct listener.
-s *scopes*	Specifies the scopes of the services to be registered.
-t *servicetype*	Specifies the service type of the service URL.
-T *IPAddress*	Specifies the IP address that the service registration needs to be sent to.
-u *URL*	Specifies the URL for the service to be registered.

Item	Description
-U	Replaces an existing registration.
-v	Specifies verbose output.
-b level	Specifies the debuglevel (from 0 to 7). A three-bit mask is used: • - 0b001 = 1 is to see the important debug information (errors and main program steps) • - 0b010 = 2 is to see the detailed debug information (detailed program steps) • - 0b100 = 4 is to see the start and stop traces of all functions.
-6	Specifies that IPv6 must be used to resolve any hostname used in URL; if omitted, IPv4 is used to resolve the host names.
-h	Display the help: Command Usage.

Examples

1. To run the command as a daemon on the default SLP port 427, enter the following command:

```
# slp_srvreg –D
```

2. To register the service with the `service:pop3://mail.ibm.com` URL and the `user=Tom, Richard` attributes for two days, enter the following command:

```
# slp_srvreg -v –a “user=Tom, Richard” –u “service:pop3://mail.ibm.com”
             –t “service:pop3” –l 172800
```

3. To register the service with the `service:pop3://mail.ibm.com` URL and the `user=Tom, Richard` attributes for two days for the local host, enter the following command:

```
# slp_srvreg –a “user=Tom, Richard” –u “service:pop3://mail.ibm.com”
             –t “service:pop3”  –l 172800 –T 127.0.0.1
```

4. To register the service with the `service:pop3://mail.ibm.com` URL and the `user=Tom, Richard` attributes for two days for the local host, enter the following command:

```
# slp_srvreg -a “user=Tom, Richard" -u “service:pop3://mail.ibm.com"
             -t “service:pop3"  -l 172800 -T 127.0.0.1
```

5. To deregister the service with the `service:pop3://mail.ibm.com` URL with important and detailled debug traces (0b011 = 3), enter the following command:

```
# slp_srvreg -d “service:pop3://mail.ibm.com" -t “service:pop3" -b 5
```

6. To kill the **slp_srvreg** daemon, enter the following command:

```
# slp_srvreg -k
```

smbcd Daemon

Purpose

Processes General Security Services API (GSSAPI) authentication requests for Server Message Block (SMB) client file system.

Syntax

```
/usr/sbin/smbcd
```

Description

The SMB client file system in the AIX operating system requires Kerberos-based GSSAPI to start the user-authenticated session by using the SMB protocol version 2.1. In the AIX operating system, the GSSAPI is provided by a Userspace Library in the IBM Network Authentication Service (NAS) version 1.16.1.0, or later fileset. This fileset is included in AIX Expansion Pack.

The **smbcd** daemon authenticates the SMB client file system to the required Kerberos-based GSSAPI and later deletes the established authentication context for the SMB client file system based on the type of request. The SMB client file system sends requests to the SMB server to access the remote shares (files and directories) during the mount or unmount operations and to reauthenticate an existing session at regular intervals or after a session expiry.

When the `smbc.rte` fileset is installed, the **smbcd** daemon is configured to operate with the System Resource Controller (SRC) master program. The SRC commands can stop and start the **smbcd** daemon. The **smbcd** daemon is started automatically when the logical partition starts. If the **smbcd** daemon is killed, the SRC master program restarts the **smbcd** daemon automatically. If the **smbcd** daemon is not running, you cannot authenticate or reauthenticate the session with the SMB server. All further attempts to access files in the SMB client file system in an unauthenticated session fails. You can start the **smbcd** daemon directly by running the **startsrc** command.

When the **smbcd** daemon starts, the **smbcd** daemon parses the `/etc/smbc/smbctune.conf` file and updates the kernel with the latest values of the tunable parameters from the file. Thus, the tunable parameter values are preserved between system restart operations.

You can determine the number of concurrent authentications that can be performed by using the **smbc_max_concurrent_mount** tunable parameter in the **smbctune** command. You can query the **smbcd** daemon status to determine the basic information such as the process ID, state, and subsystem.

Files

/var/adm/smbc/unixsock
: UNIX socket file that is used for inter-process communication between the **smbcd** daemon and the `smbclient` kernel extension.

/var/adm/smbc/smbcgssd_krb5cc
: Directory that contains the cache files of Kerberos credentials for various users.

/var/locks/LCK..smbcd
: Lock file for the **smbcd** daemon. The lock operation ensures that the **smbcd** daemon does not run multiple times on repeated invocation.

smbcstat Command

Purpose

Displays statistics information for the mount operation of the Server Message Block (SMB) client file system.

Syntax

```
/usr/sbin/smbcstat [-l] [-s <remote share> | -d <mount point>]
```

Description

The **smbcstat** command displays statistics information for the mount operation of the SMB client file system. You can view statistics information for a specific mounted remote directory, a specific mount point, or all mount points in the logical partition. If you do not specify any options, the **smbcstat** command lists statistics information for all SMB mount points in the logical partition.

Flags

-d *mount point*
Lists statistics information for the SMB mount point that is specified as an argument to this option.

-l
Lists statistics information for all SMB mount points.

-s *remote share*
Lists mount statistics information for the remote mount point that is specified as an argument to this option.

Exit status

0
Indicates success.

>0
Indicates error.

Example

- To list statistics information for all the mount points in the SMB client file system, enter the following command:

```
smbcstat -l
```

For SMB protocol version 2.1, the command displays an output similar to the following example:

```
Share Name      : FVT_SHARE
Protocol Used   : smb2.1
Port            : 445
Mount Name      : /mnt
Server Name     : winexample.com
User Name       : testusr
Domain Name     : SMB_test
Signing State   : enabled
Group Id        : 0
User Id         : 0
Mode            : 0
----------------------------
Total Number of SMB Client Filesystem Mounts   : 1
```

For SMB protocol version 3.0.2, the command displays an output similar to the following example:

```
Share Name      : FVT_SHARE
Protocol Used   : smb3.0.2
Port            : 445
Mount Name      : /mnt
Server Name     : winexample.com
User Name       : testusr
Domain Name     : SMB_test
Signing State   : enabled
Group Id        : 0
User Id         : 0
Mode            : 0
Encryption      : False
Secure Negotiate: False
----------------------------
Total Number of SMB Client Filesystem Mounts   : 1
```

smbctune Command

Purpose

Sets and displays the tunable parameters for the mount operations of the Server Message Block (SMB) client file system.

Syntax

```
smbctune [ -l | -f ] | [ [ -p ] -s smbc_tunable_parameter1=value smbc_tunable_parameter2=value
…]
```

Description

You can run this command with the **-s** flag to set new values for the tunable parameters by specifying the tunable parameters as an argument. When you set new values for the tunable parameters, the **smbctune** command updates the kernel with new values of the tunable parameters. If you use **-p** flag along with the **-s** flag, the **smbctune** command updates the `/etc/smbc/smbctune.conf` file with new values to make tunable parameters persistent. This file is used when logical partition starts and initializes the SMB client subsystem.

When you do not specify the **pver**, **signing**, **secure_negotiate**, and **encryption** parameters with the **mount** command by using the **-o** flag, default values are used from the tunable parameter values in the kernel that are read from the `smbctune.conf` file or set by using the **smbctune** command.

Flags

-f
Displays the tunable parameter values that are specified in the `/etc/smbc/smbctune.conf` file.

-l
Displays the tunable parameter values that are specified in the kernel.

-p
Makes the tunable parameter values persistent by writing values to `/etc/smbc/smbctune.conf` file. Should be used only along with -s.

-s *smbc_tunable_parameter=value*
Sets specified values to specified tunable parameters. You can update multiple tunable parameters in a single command. You must not specify any negative values.

smbc_max_concurrent_mount
Specifies the maximum number of concurrent authentications that the **smbcd** daemon can perform at a time. This value indicates the threshold limit for the number of concurrent mount operations that can be processed at a time. When you set a new value to this tunable parameter, the **smbctune** command refreshes the **smbcd** daemon so that the corresponding value in the `userspace` daemon changes accordingly. Valid values for this tunable parameter can be in the range `0-INT_MAX-1`. The default value is 0.

smbc_request_timeout
Specifies the amount of time, in seconds, to wait before a request from the SMB client system to the SMB server times out. When this tunable parameter is set to 0, the requests do not expire on the SMB client system. Valid values for this tunable parameter can be in the range `0-INT_MAX-1`. The default value for this tunable parameter is 0.

smbc_max_connections
Specifies the maximum number of SMB client connections that can exist with any number of SMB servers. When this tunable parameter is set to 0, the SMB client system can establish unlimited number of connections with the SMB servers. Valid values for this tunable parameter can be in the range `0-INT_MAX-1`. This tunable parameter is set to 0 by default.

smbc_lookup_cache_size
Specifies the cache memory size of the SMB client lookup file. This lookup file keeps N file identifiers open so that these identifiers can be reused for lookup operations. The contents of this lookup file is similar to a Least Recently Used (LRU) cache memory that is implemented for each mount operation. The size of cache memory is same for each mount operation of the SMB client systems. The size of the cache memory cannot be a negative value. Valid value for this tunable parameter can be in the range 0-100. The default value is 32 cache entries. You can set a higher value if less number of SMB client systems are connecting to the SMB server.

smbc_krb5_lifetime
Specifies the token expiry time. The SMB client system and the SMB server communicate with each other by using tokens. This token is valid for a specific period. Based on the default values of Kerberos authentication, the token is valid for 10 hours. After that time period, the authenticated session between the SMB client system and SMB server expires and you cannot perform any other operation on the mounted remote share. Token expiry time is saved both in the cache memory of the SMB client credentials and in the token. The duration of the token can be renewed by reauthenticating to the SMB server. You can set this tunable parameter to a value less than 10 hours so that you can reauthenticate to the SMB server before 10 hours. Valid values for this tunable parameter can be in the range `0-INT_MAX-1`. The default value is 0.

smbc_krb5_renew_till
Specifies the maximum time in seconds that a token can be reauthenticated to renew the token duration. After this period, the existing token context is deleted, and a new authenticated token context is created and used. Valid values for this tunable parameter can be in the range `0-INT_MAX-1`. The default value is 0. Kerberos authentication follows its own default values.

smbc_file_lease_enable
Enables the leasing function in the SMB client file system. By default, this tunable parameter is enabled and is applicable for each SMB client file system.

The SMB client file system can use the file leasing function only when the file leasing function is enabled in the SMB client and in the SMB server systems, and when the SMB server supports the file leasing function. The file leasing function is preferred over the oplock function. The valid values are 1 and 0. You can change the value of this tunable parameter only when SMB client mount points are not available on the SMB client system.

smbc_oplock_enable
Enables the opportunistic locks (oplock) function for the SMB client system. By default, this tunable parameter is enabled and is applicable for each SMB client system.

The SMB client file system can use the oplock function only when the oplock function is enabled in the SMB server and in the SMB client system, and when the file leasing function is disabled in the SMB server or SMB client system. The valid values are 1 and 0. You can change the value of this tunable parameter only when SMB client mount points are not available on the SMB client system.

Note: When the SMB server or the SMB client system disables the file leasing function and the oplock function, the SMB client file system uses direct I/O operations.

smbc_protocol_version
Specifies the version of the SMB protocol that is used to communicate with the SMB server. The valid values are 2.1, 3.0.2, and auto. The default value is `auto`. For the value, `auto`, the SMB client protocol version 2.1 or version 3.0.2 is used based on the specified version of the SMB server.

smbc_signing
Specifies whether the file system of SMB client requires digital signature for communication. The valid values are `enabled` and `required`. The default value is `enabled`.

smbc_secure_negotiate
Specifies whether file system of SMB client requires secure dialect negotiation capability. Valid values are `desired`, `required`, and `disabled`. Default value is `desired`.

smbc_encryption
Specifies whether SMB client file system requires encryption. Valid values are `desired`, `required`, and `disabled`. Default value is `desired`.

The following table shows few tunable parameters in the `smbctune.conf` file and their peer options for the **mount** command.

Tunable parameter of the smbctune.conf file	Corresponding mount option	Valid tunable parameter values (smbctune.conf file)	Default tunable parameter values (smbctune.conf file)
smbc_protocol_version	pver	2.1, 3.0.2, auto	auto
smbc_signing	signing	enabled, required	enabled
smbc_secure_negotiate	secure_negotiate	desired, required, disabled	desired
smbc_encryption	encryption	desired, required, disabled	desired

Exit status

0
Indicates success.

>0
Indicates error.

Example

- To set the value of the **smbc_max_concurrent_mount** tunable parameter to 20, enter the following command:

```
/usr/sbin/smbctune -s smbc_max_concurrent_mount=20
```

- To update the **smbc_max_concurrent_mount** tunable parameter value, enter the following command:

```
/usr/sbin/smbctune -p -s smbc_max_concurrent_mount=20
```

Note: The command also updates the `smbctune.conf` file, which makes the updates persistent.

smdemon.cleanu Command

Purpose

Cleans up the **sendmail** queue for periodic housekeeping.

Syntax

/usr/lib/smdemon.cleanu

Description

The **smdemon.cleanu** command is a shell procedure that cleans up the **sendmail** command queue and maintains the **/var/spool/mqueue/log** file.

To enable the **smdemon.cleanu** command, you must remove the comment statement by deleting the # character from the beginning of the **smdemon.cleanu** line in the **/var/spool/cron/crontabs/root** file. If the **/var/spool/mqueue** directory does not exist, do not change the **/var/spool/cron/crontabs/root** file.

Be careful that the average size of a log file for each **smdemon.cleanu** session multiplied by the number of log files does not use more space than you need. You can arrange the number of log files to suit your needs.

Note: The **smdemon.cleanu** command is not usually entered on the command line. The command is executed by the **cron** daemon.

Examples

To run the **smdemon.cleanu** procedure automatically, edit the **/var/spool/cron/crontabs/root** file and delete the # (comment character) from the beginning of the **smdemon.cleanu** line as follows:

```
# ulimit 5000; /usr/lib/smdemon.cleanu > /dev/null
```

Files

Item	Description
/var/spool/cron/crontabs/root	Schedules when the **smdemon.cleanu** command will run.
/var/spool/mqueue	Contains the **log** file and temporary files associated with the message in the mail queue.

smit Command

Purpose

Performs system management.

Syntax

smit [**-C** | **-M**] [**-D**] [**-f**] [**-h**] [**-l** *File*] [**-o** *PathName*] [**-p** *Entity/ValueString*] [**-r** *RunMode*] [**-s** *File*] [**-t**] [**-v**] [[**-m** | **-n** | **-d**] *FastPath*] [**-X**] [**-x**]

Description

The **smit** command invokes the System Management Interface Tool (SMIT). SMIT is an interactive interface application designed to simplify system management tasks. The **smit** command displays a hierarchy of menus that can lead to interactive dialogues. SMIT builds and runs commands as directed by the user. Because SMIT runs commands, you need the authority to execute the commands that SMIT runs.

SMIT creates two files, the **smit.script** file and the **smit.log** file. Invoking the **smit** command with the **-s** *PathName* flag saves the **smit.script** file in the file specified by the *PathName* parameter. If the **-s** flag is not specified, the script information is saved in the **$HOME/smit.script** file. Invoking the **smit** command with the **-l** *PathName* flag saves the **smit.log** file in the file specified by the *PathName* parameter. If the **-l** flag is not specified, the log information is recorded in the **$HOME/smit.log** file. You must have write permission for the directory in which you have requested the **smit** file to be written or the **smit.script** file and **smit.log** file are not created. SMIT does not overwrite the **smit.log** file or the **smit.script** file. The files are appended when possible.

The **smit.script** file automatically records the commands with the command flags and parameters used. The **smit.script** file can be used as an executable shell script to duplicate system configuration. SMIT creates the **smit.log** file, which contains additional detailed information that can be used by programmers in extending the SMIT system. The **smit.log** file is affected by the **-D**, **-l**, **-t**, and **-v** flags.

The **smit** command takes you to the top level of the menu hierarchy if you do not use the *FastPath* parameter. To enter the menu at lower levels, use the *FastPath* parameter. All commands run by SMIT can be used as *FastPaths*. The *FastPath* parameter will assist you as you become familiar with the commands. For example, you can enter: `smit chuser` to go directly to the dialog from which you can change user characteristics.

Note: User access to SMIT panels may be controlled with the `smitacl.user` or `smitacl.group` commands.

SMIT requires access to the following files:

Item	Description
sm_menu_opt	SMIT database
sm_name_hdr	SMIT database
sm_cmd_hdr	SMIT database
sm_cmd_opt	SMIT database
smit.log	SMIT log file
smit.script	SMIT script file
/usr/lpp/msg/.../smit.cat	Message Catalog

Note: If any of these files are corrupt, or exist on an NFS server and that server goes down, SMIT may fail to respond.

Flags

Item	Description
-C	Starts SMIT using an ASCII (also called Curses) interface.
-D	Sets the debug mode; sets **-t** and **-v** flags.
-d *FastPath*	Identifies that the *FastPath* is the name of a dialogue.
-f	Allows standard in and standard out from SMIT to be redirected.
-h	Displays the command usage message.
-l *File*	Redirects the **smit.log** file to the specified *File*.
-M	Starts SMIT using a windows (also called Motif) interface.
-m *FastPath*	Identifies that the *FastPath* is the name of a menu.
-n *FastPath*	Identifies that the *FastPath* is the name of a selector.
-o *PathName*	Specifies a directory *PathName* of an alternate repository for SMIT objects. The default directory is **/etc/objrepos**.
-p *Entity/ValueString*	This flag only applies to the smit windows version. Allows nameselects and dialogs to be filled in from the command line. Also allows you to operate on multiple entities simultaneously. You can set the environment variables ENTITY_SEP and VALUE_SEP to override the default comma and semicolon separators. You can enter *Entity/ValueString* in any of the following formats: **"***Entity1***:***Val1***,***Val2***...** **;** *Entity2***:***Val1***,***Val2***...** **;** **..."** or **"***Val1***,***Val2***...** **;** *Val1***,***Val2***...** **;** **..."**

Item	Description
-r *RunMode*	This flag only applies to smit windows version. Specifies the mode to run msmit in. You can enter the following values for *RunMode*: **1** Exit msmit when **done** is clicked in the output window. **2** Exit msmit when **ok** is clicked in a dialog. Print the dialog options upon exit. Do not run the command. **3** Run msmit silently, print the dialog options. Do not run the command. **4** Exit msmit when **ok** is clicked in the dialog. Print the commands upon exit. Do not run the command.
-s *File*	Redirects the **smit.script** file to the specified *File*.
-t	Records detailed trace information in the **smit.log** file.
-v	Records the command strings for intermediate and target task commands run by SMIT, and also records their output in the **smit.log** file.
-x	Does not run any **command_to_execute**, but still logs them for later execution.
-X	Does not run any **command_to_discover**, **command_to_list**, **command_to classify** or **command_to_execute**.

Examples

1. To display the main menu in the overall system management hierarchy, enter:

   ```
   smit
   ```

2. To change the characteristics of a user, enter:

   ```
   smit chuser
   ```

   The **chuser** command is an example of a *FastPath* parameter. The **smit** command and the *FastPath* parameter **chuser** takes you directly to the dialog, `Change User Attributes`, which guides you through changing the characteristics of a user.

3. To make the **smit.script** file executable for duplicate configuration, enter:

   ```
   chmod +x smit.script
   ```

   Then, to duplicate your configuration, enter:

   ```
   smit.script
   ```

   The **smit.script** file can be edited to create slight variations in the configuration commands, or to use only subsets of the commands. The **smit.script** file should be renamed or copied to prevent SMIT from modifying it.

   > **Note:** SMIT runs commands under the Korn shell (**/usr/bin/ksh**). Some command strings in the **smit.script** file may require this environment to run correctly.

Files

Item	Description
/usr/bin/smit	Contains the **smit** command.
/etc/objrepos	Specifies the default directory for the SMIT database.
smit.log	Specifies detailed information of your session, with time stamps.
smit.script	Specifies only the target task commands run by SMIT, with time stamps.

smitty Command

Purpose

Provides a Curses-based text interface to perform system management.

Syntax

smitty [**-C**] [**-D**] [**-f**] [**-h**] [**-l** *File*] [**-o** *PathName*] [**-s** *File*] [**-t**] [**-v**] [[**-m** | **-n** | **-d**] *FastPath*] [**-X**] [**-x**]

Description

The **smitty** command invokes the System Management Interface Tool (SMIT). SMIT is an interactive interface application designed to simplify system management tasks. The **smitty** command displays a hierarchy of menus that can lead to interactive dialogues. SMIT builds and runs commands as directed by the user. Because SMIT runs commands, you need the authority to execute the commands that SMIT runs.

Note: The **smitty** command is identical to **smit -C**.

SMIT creates two files, the **smit.script** file and the **smit.log** file. Invoking the **smitty** command with the **-s** *PathName* flag saves the **smit.script** file in the file specified by the *PathName* parameter. If the **-s** flag is not specified, the script information is saved in the **$HOME/smit.script** file. Invoking the **smitty** command with the **-l** *PathName* flag saves the **smit.log** file in the file specified by the *PathName* parameter. If the **-l** flag is not specified, the log information is recorded in the **$HOME/smit.log** file. You must have write permission for the directory in which you have requested the **smit** files to be written or the **smit.script** file and **smit.log** file are not created. SMIT does not overwrite the **smit.log** file or the **smit.script** file. The files are appended when possible.

The **smit.script** file automatically records the commands with the command flags and parameters used. The **smit.script** file can be used as an executable shell script to duplicate system configuration. SMIT creates the **smit.log** file, which contains additional detailed information that can be used by programmers in extending the SMIT system. The **smit.log** file is affected by the **-D**, **-l**, **-t**, and **-v** flags.

The **smitty** command takes you to the top level of the menu hierarchy if you do not use the *FastPath* parameter. To enter the menu at lower levels, use the *FastPath* parameter. All commands run by SMIT can be used as *FastPaths*. The *FastPath* parameter will assist you as you become familiar with the commands. For example, you can enter: `smitty chuser` to go directly to the dialog from which you can change user characteristics.

SMIT requires access to the following files:

Item	Description
sm_menu_opt	SMIT database
sm_name_hdr	SMIT database
sm_cmd_hdr	SMIT database

Item	Description
sm_cmd_opt	SMIT database
smit.log	SMIT log file
smit.script	SMIT script file
/usr/lpp/msg/.../smit.cat	Message Catalog

Note: If any of these files are corrupt, or exist on an NFS server and that server goes down, SMIT may fail to respond.

Flags

Item	Description
-C	Starts SMIT using a Curses-based text interface. This is the default for the **smitty** command.
-D	Sets the debug mode; sets **-t** and **-v** flags.
-d *FastPath*	Identifies that the *FastPath* is the name of a dialogue.
-f	Allows standard in and standard out from SMIT to be redirected.
-h	Displays the command usage message.
-l *File*	Redirects the **smit.log** file to the specified *File*.
-m *FastPath*	Identifies that the *FastPath* is the name of a menu.
-n *FastPath*	Identifies that the *FastPath* is the name of a selector.
-o *PathName*	Specifies a directory *PathName* of an alternate repository for SMIT objects. The default directory is **/etc/objrepos**.
-s *File*	Redirects the **smit.script** file to the specified *File*.
-t	Records detailed trace information in the **smit.log** file.
-v	Records the command strings for intermediate and target task commands run by SMIT, and also records their output in the **smit.log** file.
-x	Does not run any **command_to_execute**, but still logs them for later execution.
-X	Does not run any **command_to_discover**, **command_to_list**, **command_to classify** or **command_to_execute**.

Examples

1. To display the main menu in the overall system management hierarchy, enter:

   ```
   smitty
   ```

2. To change the characteristics of a user, enter:

   ```
   smitty chuser
   ```

   The **chuser** command is an example of a *FastPath* parameter. The **smitty** command and the *FastPath* parameter **chuser** takes you directly to the dialog, `Change User Attributes`, which guides you through changing the characteristics of a user.

   Note: The **smitty chuser** command should be used to modify only local users.

3. To make the **smit.script** file executable for duplicate configuration, enter:

```
chmod +x smit.script
```

Then, to duplicate your configuration, enter:

```
smit.script
```

The **smit.script** file can be edited to create slight variations in the configuration commands, or to use only subsets of the commands. The **smit.script** file should be renamed or copied to prevent SMIT from modifying it.

> **Note:** SMIT runs commands under the Korn shell (**/usr/bin/ksh**). Some command strings in the **smit.script** file may require this environment to run correctly.

Files

Item	Description
/usr/bin/smitty	Contains the **smitty** command.
/etc/objrepos	Specifies the default directory for the SMIT database.
smit.log	Specifies detailed information of your session, with time stamps.
smit.script	Specifies only the target task commands run by SMIT, with time stamps.

smrsh Command

Purpose

Restricted shell for sendmail.

Syntax

`smrsh` **-c** *command*

Description

The `smrsh` command is intended as a replacement for the `sh` command in the prog mailer in `sendmail` configuration files. The `smrsh` command limits the programs that can be run using the `sendmail` command syntax. This improves overall system security. `smrsh` limits the set of programs that a programmer can execute, even if `sendmail` runs a program without going through an alias or forward file.

The `smrsh` command requires that programs be in the **/var/adm/sm.bin** directory. This allows system administrators to choose which programs can be run by the `smrsh` command. The `smrsh` command also rejects any commands with the following characters on the command line to prevent end-run attacks: ,, <, >, |, ;, &, $, \r (<RETURN>), or \n (<NEWLINE>) on the command line to prevent end run attacks.

- ,
- <
- >
- |
- ;
- &
- $
- \r (<RETURN>)
- or \n (<NEWLINE>)

Initial pathnames on programs are stripped, so forwarding to **/usr/ucb/vacation**, **/usr/bin/vacation**, **/home/server/mydir/bin/vacation**, and **vacation** all actually forward to **/var/adm/sm.bin/vacation**. System administrators should be conservative about populating **/var/adm/sm.bin**. Reasonable additions are utilities such as `vacation(1)` and `procmail`. Never include any shell or shell-like programs (for example, `perl`) in the **sm.bin** directory. This does not allow the execution of arbitrary programs, but does not restrict the use of shell or perl scripts in the **sm.bin** directory (using the **#!** syntax).

Flags

-c *command*
Runs the program specified by *command*.

Location

/usr/sbin/smrsh
Default location of smrsh command.

Files

/var/adm/sm.bin
Directory for restricted programs.

smtctl Command

Purpose

The **smtctl** command controls the enabling and disabling of processor simultaneous multithreading mode.

Syntax

smtctl [**-m off** | **on** [**-w boot** | **now**]]

smtctl [**-t #SMT** [**-w boot** | **now**]]

smtctl [**-m suspend** [**-w boot**]]

smtctl [**-m limit** [**-t #SMT**][**-w boot**]]

smtctl [**-m recommended** [**-w boot** | **now**]]

Description

This command is provided for privileged users and applications to control utilization of processors with simultaneous multithreading support. The simultaneous multithreading mode allows processors to have thread level parallelism at the instruction level. This mode can be enabled or disabled for all processors either immediately or on subsequent boots of the system. This command controls the simultaneous multithreading options.

Each individual Simultaneous Multi-threading (SMT) thread of a physical processor core is treated as an independent logical processor by AIX. The AIX operating system limits the combination of physical processor cores assigned and SMT modes in order to maintain symmetry across all of the physical processor cores assigned to AIX. Due to this limitation, the number of logical processor is less than or equal to 1024 for AIX 7.1 and 256 for AIX 6.1.

The POWER8 processors are capable of SMT-8 which means up to 128 cores can be used in SMT-8 mode which yields 1024 logical processors. A lower SMT mode must be used for AIX users to be able to use more than 128 POWER8 cores.

Number of thread

When booting a P8 Logical Partition (LPAR), the default number of SMT thread is 4. To increase the default number of SMT threads dynamically, enter:

```
smtctl -m on
smtctl -t 8
```

The change to SMT-8 is effective immediately and reboot is not required. If you want the setting to persist after rebooting, then you must rebuild the boot image with the **bosboot** command. The default SMT-4 is intended for better performance for an existing applications that are not designed or compiled for more than 4 threads.

Number of cores

If you have allocated more than 128 cores to an LPAR, by default it uses 128 cores. This is to ensure that AIX limit of maximum 1024 logical processors is not exceeded if SMT-8 is enabled (128 cores * SMT8 = 1024 total). If you want LPAR to use more than 128 cores, then you need to run a sequence of following AIX commands to establish a limit to the number of SMT threads that are available per core.

```
smtctl -m limit -t 4
bosboot -a
shutdown -Fr
```

Upon rebooting, AIX negotiates with the firmware to allow up to 256 cores because the operating system's limit of 1024 processors will not be exceeded with the specified limit of 4 SMT threads. You can exceed 256 cores if you run the **smtctl** command as stated above, but with a limit of 2 instead of 4. The following command suspends SMT capability allowing more cores.

```
smtctl -m suspend
bosboot -a
shutdown -Fr
```

Flags

Item	Description
-m off	Set the simultaneous multithreading mode to `disabled`. This option cannot be used with the **-t** flag.
-m on	Set the simultaneous multithreading mode to `enabled`. By using the **-m** flag, the maximum number of threads supported per processor is enabled. This option cannot be used with the **-t** flag.
-t #SMT	Set the number of the simultaneous threads per processor. The value may be set to one to disable simultaneous multi-threading. The value may be set to two for systems that support 2-way simultaneous multi-threading and the value may be set to four, for the systems that support 4-way simultaneous multi-threading.
-w boot	Makes the simultaneous multithreading mode change effective on next and subsequent reboots if you run the **bosboot** command before the next system reboot.
-w now	Makes the simultaneous multithreading mode change immediately but will not persist across reboot. If the **-w boot** or the **-w now** option is specified, the mode change is made immediately and will persist subsequent reboots if you run the **bosboot** command before the next system reboot.

Item	Description
-m limit	Limits the number of simultaneous multithreading threads to two, or the specified value if the -t flag is used and enables more processor nodes, if available, effective at the next reboot (running bosboot is required to rebuild the boot image). This limit cannot be dynamically changed during run time, and you must reboot to change the operating state.
-m suspend	Suspends the simultaneous multithreading capability, and enables more processor nodes, if available, effective at the next reboot (running bosboot is required to rebuild the boot image). This limit cannot be dynamically changed during run time, and you must reboot to change the operating state.
-m recommended	Sets the number of threads to a value that provides the best performance for the most common types of workloads that are based on the underlying physical processor type. This setting takes place immediately. You can also specify **next boot** to start using the new value.

If no options are specified then the following simultaneous multithreading settings will be reported:

Item	Description
SMT Capability	Indicator that the physical or virtual processors are capable of simultaneous multithreading.
SMT Mode	Current runtime simultaneous multithreading mode of `disabled` or `enabled`.
SMT Boot Mode	Current boot time simultaneous multithreading mode of `disabled` or `enabled`.
SMT Threads	Number of simultaneous multithreading threads per physical or virtual processor.
SMT Bound	Indicator that the simultaneous multithreading threads are bound on the same physical or virtual processor.
SMT Thread Capability	Maximum number of simultaneous multi-threading threads per physical or virtual processor supported by the system.

Exit Status

Item	Description
0	Successfully completed the requested operation.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Examples

1. To enable simultaneous multithreading for the current boot cycle, enter:

```
smtctl -m on -w now
```

The system displays a message similar to the following:

```
smtctl: SMT is now enabled.
```

2. To enable a 2-way simultaneous multithreading on a system that supports up to 4 way, enter:

```
smtctl -t 2 -w now
```

The system displays a message similar to the following:

```
smtctl: SMT is now enabled.
```

3. To view the current simultaneous multithreading mode settings and processor information, enter:

```
smtctl
```

The system displays a message similar to the following:

```
This system is SMT capable.

This system supports up to 4 SMT threads per processor
SMT is currently enabled.

SMT boot mode is set to disabled.


proc0 has 2 SMT threads
Bind processor 0 is bound with proc0
Bind processor 1 is bound with proc0


proc2 has 2 SMT threads
Bind processor 2 is bound with proc2
Bind processor 3 is bound with proc2
```

4. To disable simultaneous multithreading for the current boot cycle and for all subsequent boots, enter:

```
smtctl -m off
```

The system displays a message similar to the following:

```
smtctl: SMT is now disabled. It will persist across reboots if
        you run the bosboot command before the next reboot.
```

Another method to disable simultaneous multi-threading for the current boot cycle and for subsequent boots, enter:

```
smtctl -t 1
```

Note: The boot image must be remade with the **bosboot** command before the next reboot.

Location

/usr/sbin/smtctl

Files

Item	Description
/usr/sbin/smtctl	Contains the **smtctl** command.

snap Command

Purpose

Gathers system configuration information.

Syntax

snap [**-@**] [**-a**] [**-z** *"product_name=prd_name,..." | "class=myclass,.." |* **ALL**] [**-M** *Timeout*][**-A**] [**-b**] [**-B**] [**-c**] [**-C**] [**-D**] [**-f**] [**-F**] [**-g**] [**-G**] [**-i**] [**-k**] [**-l**] [**-L**][**-n**] [**-N**] [**-p**] [**-r**] [**-R**] [**-s**] [**-S**] [**-t**] [**-T** *Filename*][**-u** *user1,...*] [**-w**] [**-X**] [**-Y**][**-o** *OutputDevice*] [**-d** *Dir*] [**-v** *Component*] [-0 *FileSplitSize*] [**-P** *Files*] [*script1 script2 ...* | All | file:*filepath*] [**-U**]

snap -e [**-m** *Nodelist*] [**-d** *Dir*]

snap -z ADD [*"product_name=prod_name" "class=myclass" "command_path=/tmp/myprod_myscript -a"*]

snap -z DELETE [*"product_name=prod_name"*]

Description

The **snap** command gathers system configuration information and compresses the information into a **pax** file. The file may then be written to a device such as tape or DVD, or transmitted to a remote system. The information gathered with the **snap** command might be required to identify and resolve system problems.

Note: Root user authority is required to execute the **snap** command. Use the **snap -o /dev/cd0** command to copy the compressed image to DVD. Use the **snap -o /dev/rmt0** command to copy the image to tape.

Use the **snap -o /dev/rfd0** command to copy the compressed image to diskette. Use the **snap -o /dev/rmt0** command to copy the image to tape.

At least 8 MB of temporary disk space is required to collect all system information, including contents of the error log. If you do not gather all system information with the **snap -a** command, less disk space may be required (depending on the options selected).

Note: If you intend to use a tape to send a snap image to IBM for software support, the tape must be one of the following formats:

- 8 mm, 2.3 GB capacity
- 8 mm, 5.0 GB capacity
- 4 mm, 4.0 GB capacity

Using other formats prevents or delays IBM software support from being able to examine the contents.

The **snap -g** command gathers general system information, including the following:

- Error report
- Copy of the customized Object Data Manager (ODM) database
- Trace file
- User environment
- Amount of physical memory and paging space
- Device and attribute information
- Security user information
- Configuration and tuning parameter information of the system

The output of the **snap -g** command is written to the **/tmp/ibmsupt/general/general.snap** file.

The **snap** command checks for available space in the **/tmp/ibmsupt** directory, the default directory for **snap** command output. You can write the output to another directory by using the **-d** flag. If there is not enough space to hold the **snap** command output, you must expand the file system.

Each execution of the **snap** command appends information to previously created files. Use the **-r** flag to remove previously gathered and saved information.

Flags

Item	Description
-@	Gathers the workload partition information.
-a	Gathers all system configuration information except HACMP specific data. To gather HACMP specific data, run the **snap -e** option. Collection of registered debug data scripts for external products gets executed and their data is also included as part of system configuration and it can be limited for selected products by specifying their names with the **–z** flag. The **-a** option requires at least 8 MB of temporary disk space.
-A	Gathers asynchronous (TTY) information.
-b	Gathers SSA information.
-B	Bypasses collection of SSA adapter dumps. The **-B** flag only works when the **-b** flag is also specified; otherwise, the **-B** flag is ignored.
-c	Creates a compressed **pax** image (**snap.pax.Z** file) of **snap** known component subdirectories in the `/tmp/ibmsupt` directory tree or any other user-defined directory that is specified with the **-d** flag. **Note:** Information that is not gathered with this option must be copied to the **snap** directory tree before using the **-c** flag. If a test case is needed to demonstrate the system problem, copy the test case to the `/tmp/ibmsupt/testcase` directory before compressing the **pax** file. Any directories that are defined by the user must be saved in the `/tmp/ibmsupt/other` directory for the **snap** command to compress them.
-C	Retrieves all the files in the fwdump_dir directory. The files are placed in the "general" subdirectory. The **-C** snap option behaves the same as **-P***.
-D	Gathers dump and **/unix** information. The primary dump device is used. **Note:** 1. If **bosboot -k** was used to specify the running kernel to be other than /**unix**, the incorrect kernel is gathered. Make sure that /**unix** is, or is linked to, the kernel in use when the dump was taken. 2. If the dump file is copied to the host machine, the **snap** command does not collect the dump image in the **/tmp/ibmsupt/dump** directory. Instead, it creates a link in the dump directory to the actual dump image.
-d *AbsolutePath*	Identifies the optional **snap** command output directory (**/tmp/ibmsupt** is the default). You must specify the absolute path.
-e	Gathers HACMP specific information. **Note:** HACMP specific data is collected from all nodes belonging to the cluster. This flag cannot be used with any other flags except **-m** and **-d**.
-f	Gathers file system information.
-F	Gathers flash adapter information.
-g	Gathers the output of the **lslpp -hac** command, which is required to recreate exact operating system environments. Writes output to the **/tmp/ibmsupt/general/lslpp.hac** file. Also collects general system information and writes the output to the **/tmp/ibmsupt/general/general.snap** file.
-G	Includes predefined **Object Data Manager** (ODM) files in general information collected with the **-g** flag.

Item	Description
-i	Gathers installation debug vital product data (VPD) information.
-k	Gathers kernel information
-l	Gathers programming language information.
-L	Gathers LVM information.
-m *Nodelist*	Node name list (separated by commas) to gather HACMP information. **Note:** Currently this flag is only valid with the **-e** flag.
-M *Timeout*	Specifies the maximum time out value in seconds, that the snap frame work waits before it kills one registered external product debug data command. Default time out value is 300 seconds.
-n	Gathers **Network File System** (NFS) information.
-N	Suppresses the check for free space required.
-o *OutputDevice*	Copies the compressed image onto the specified device.
`-0` *FileSplitSize*	Used to enable splitting of the `snap` output files into smaller files. The size of these files is specified as a parameter to the `-0` option and must be specified in megabytes. This flag can only be used when the `-c` flag is specified.
-p	Gathers printer information.
-P *Files*	Retrieves the named *Files* from the `fwdump_dir` directory. If **-P *** is specified, all the files in the directory are gathered. The files are placed in the `general` subdirectory. The **-C** `snap` option behaves the same as **-P***.
-r	Removes **snap** command output from the **/tmp/ibmsupt** directory.
-R	Gathers SCSI RAID information.
-s	Gathers Systems Network Architecture (SNA) information.
-S	Includes security files in general information collected with the **-g** flag.
-t	Gathers **Transmission control protoco**l information.
-T *Filename*	Gathers all the log files for a multi-CPU trace. Only the base file, **trcfile**, is captured with the **-g** flag.
-u *user1,user2...*	Specifies comma separated user names whose shell and System Management Interface Tool (SMIT) history is to be collected.
-v *Component*	Displays the output of the commands executed by the **snap** command. Use this flag to view the specified name or group of files. **Note:** Press the Ctrl-C key sequence to interrupt the **snap** command. A prompt will return with the following options: press the Enter key to return to current operation; press the S key to stop the current operation; press the Q key to quit the **snap** command completely.
-w	Gathers WLM information.
-X	Gathers X.25 (Packet-based Communication Protocol) information.
-Y	Gathers InfiniBand information and saves it in the `/tmp/ibmsupt/IB` directory.

Item	Description
-z	Facilitates debug data collection for external products. • The **ADD** keyword allows external products to register their debug data collection script with the snap framework. • The **DELETE** keyword allows external products to deregister their debug data collection script with the snap framework. When a product name is specified as parameter to the **product_name** attribute, a registered debug data collection command is executed. To collect data for more than one product specify the required product names in the **product_name** attribute. When a class name is specified as parameter to the **class** attribute, registered debug command of all the products in that class are executed. To collect data for more than one class specify the required class names in the **class** attribute. When **ALL** is specified as parameter, registered debug data collection command of all the products in all classes is executed. When any script gets executed, system appends the product name to the list pointed by **SNAPDEBUGDATA** environment variable.
-U	Collects Live kernel update information and save it in the **/tmp/ibmsupt/liveupdate** directory.

Parameters

Arguments

Names of third-party scripts to be executed are specified as parameters to `snap`. A parameter can be a single word or a list of words enclosed in quotes. When parameters are enclosed in quotes, the first parameter in the list represents the name of the script and the subsequent words represent the arguments to pass to the script.

When `All` is specified as a parameter, all the scripts in the script repository are executed. No script parameters may be passed in this case.

If the `file:` keyword is used and is immediately followed by a path to a file, that file is read to get the scripts to execute. Each line in the file represents a script and optional parameters to the script .

snap Scripts

A third-party script must be executable in `/usr/lib/ras/snapscripts`, and must follow the guidelines described below. When called during pass 1, a script must return its size estimation to `snap`. In pass 2, it collects the data and saves it as specified by `snap`.

The script must read and utilize the following environment variables, SNAPDIR, PASSNO, SCRIPTSIZE and SCRIPTLOG.

The scripts or commands can also use **SNAPDEBUGDATA** variable to learn about the debug data collected by snap script. This variable has comma separated name of the products for which the **snap** command collects the data during execution.

All output files must be written to $SNAPDIR. This is the directory where the script should be saving its output. The PASSNO variable contains the `snap` phase during which the script is called. During the first pass, the script should calculate a size estimation for the data it will write during the second pass. It will then write that numeric estimation to the file pointed to by $SCRIPTSIZE. The value saved to the file should be in decimal. `snap` passes the path to a log file where all debug data for the script should be saved. Standard out and standard error should not be redirected by the script, because `snap` will save standard out and standard error to $SNAPDIR**/***ScriptName***.out** and $SNAPDIR**/***ScriptName***.err**, respectively.

The following example shows a snap script:

```
#!/usr/bin/ksh


if [ "$PASSNO" = 1 ]
then
         (( size=99999 ))
        ....
        # this is where code to do the size estimation should go.
        ....
        echo $size > $SCRIPTSIZE

else if [ "$PASSNO" = 2 ]
then
        # debug information should go to $SCRIPTLOG
        echo "Debug Data" >> $SCRIPTLOG

        # .....where the work to collect the data takes place
        # ...

        # The data collected should be written to $SNAPDIR        .
        touch $SNAPDIR/foo_output1
        touch $SNAPDIR/foo_output2
fi
fi
```

Note: To collect information about virtual SCSI devices, run the **snap client_collect,all** command. If you need to collect data from the Virtual I/O server, see the **snap** command page on the Virtual I/O server, which uses different syntax from the **snap** command on AIX.

The following scripts can be run when you run the **snap** command with the **-a** or **-g** flags:

- To run with the **-a** flag: svCollect, client_collect, lsvirt
- To run with the **-g** flag: svCollect, client_collect

Splitting of snap Output

If it is split, snap output might look like the following:

```
% ls -l
total 112048
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xaa
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xab
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xac
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xad
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xae
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xaf
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xag
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xah
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xai
-rw-r--r--   1 lmic     adm          744518 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xaj
```

Executing Third Party Scripts

An external product debug data collection command or script is a standalone executable. The script is registered with the snap framework before it can be used to collect user defined debug data. These scripts can be de-registered as per user discretion.

Following is the ODM class defined in the system.

```
#define DEFAULTSIZE 256
#define DATA_VALUESIZE 1024

class snap_config {
        char product_name[DEFAULTSIZE]; key
        char class[DEFAULTSIZE];key
        char command_path[DATA_VALUESIZE];
        vchar sc_reserved1[DATA_VALUESIZE];
        vchar sc_reserved2[DATA_VALUESIZE];
}
```

product_name
Specify the name of the product. The same name is used for deregistration of the product debug data collection script.

class
Class can be a storage, a network or a database. You can choose appropriate class based on the product or define your own class. Class helps in the classification of the products. Users can contact IBM service personnel to add any other class in the **snap** documentation.

command_path
Path of the command or executable along with its options. **sc_reserved1** and **sc_reserved2** are reserved.

Registration of Third Party Debug Script with Snap framework

Registration can be done in two ways:

1. You can explicitly run **odmadd** command to add the entry. In such case:
   - You must copy the script or executable to `/usr/lib/ras/snapscript/bin/<productname>` directory.

   Points to remember:

   a. You need to enter the command before executing the **odmadd**:

   ```
   export ODMDIR=/usr/lib/objrepos
   ```

   b. After the **odmadd** command completes, you can restore the old value of the **ODMDIR** command.

   c. You can continue running the **snap** command. For example, the content of *myfile* is given below:

   ```
   product_name=myprod
   class=myclass
   command_path=/usr/lib/ras/snapscripts/bin/prod_name/myscript1.sh -t 10
   export ODMDIR=/usr/lib/objrepos
   odmadd myfile
   ```

   Note: Users making direct entry to ODM must take care of duplicate entries as the **snap** command processes only one entry for a particular product name. So, the **odmdelete** command must be executed before the **odmadd** command is invoked.
2. Use the **ADD** keyword with the **–z** flag.

Note:

- 1. If the debug binary is changed or updated, the user must re-register the component to update the snap repository with the latest binary.
- 2. Combination of multiple commands as a part of **command_path** variable is not supported. For example, the following format is not supported:

```
command_path=<path>/ls|<path>/grep myfile
```

- 3. Special characters like, **'**, **<**, **|** are not supported as values to the **command_path** variable.

Deregistration of Third party debug scripts from Snap framework

Deregistration can be done in two ways:

1. Use the **odmdelete** command to deregister the product. For example,

   ```
   export ODMDIR=/usr/lib/objrepos
   odmdelete -o snap_config -q product_name=productname
   ```
2. Use the **DELETE** keyword with the **–z** flag. For example,

   ```
   Snap –z DELETE product_name=productname
   ```

Examples

1. Enter the following command to gather all system configuration information:

   ```
   snap -a
   ```

   The output of this command is written to the **/tmp/ibmsupt** directory.
2. Enter the following command to create a **pax** image of all files contained in the **/tmp/ibmsupt** directory:

   ```
   snap -c
   ```

3. Enter the following command to gather general system configuration information, including the output of the **lslpp -hac** command:

   ```
   snap -g -o /dev/rfd0
   ```

   Output is written to the **/tmp/ibmsupt/general/lslpp.hac** and **/tmp/ibmsupt/general/general.snap** files. This command also writes the system information to a removable diskette.
4. Enter the following command to gather HACMP specific information from nodes node1 and node2 belonging to a single cluster:

   ```
   snap -e -m node1,node2
   ```

   Output is written to the **/tmp/ibmsupt/hacmp** directory.
5. To run the scripts foo1, foo2 and foo3. where foo1 takes no argument, foo2 takes three arguments and foo3 takes one argument, type the following:

   ```
   snap  foo1 "foo2 -x -y 3" "foo3 6"
   ```

   Output is written to `/tmp/ibmsupt/snapscripts/foo1`, `/tmp/ibmsupt/snapscripts/foo2` and `/tmp/ibmsupt/snapscripts/foo3` assuming the destination directory is the default, `/tmp/ibmsupt`.
6. To specify the `All` parameter to run all the scripts, type:

   ```
   snap All
   ```

   Note: No parameters are passed in this case.
7. To specify the path to a file containing the name and optional parameter list of scripts to execute, type:

   ```
   snap file:/tmp/scriptnames
   ```

   A sample input file to execute the scripts from example 5:

   ```
   foo1
   foo2 -x -y 3
   foo6
   ```

8. If splitting of the `snap` output into 4MB files is desired, type:

   ```
   snap -a -c -O 4
   ```

9. To submit only the HACMP `snap -e` data from nodes node1 and node2, enter the following command:

   ```
   snap -e -m node1,node2
   snap -c
   ```

   Submit the <pax.z> file to IBM according to the instructions of the service representative.

10. To submit all of the snap data from nodes node1 and node2, enter the following commands:

```
snap -e -m node1,node2
snap -a
snap -c
```

    Submit the <pax.z> file to IBM according to the instructions of the service representative.

11. To register a debug script present in the /usr/lpp/abc/debug_abc directory of product **abc**, in class **storage** enter the following command:

```
snap –z ADD "product_name=abc" "class=storage" "command_path=/usr/lpp/abc/debug_abc –a"
```

12. To deregister a debug script of product **abc**, enter the following command:

```
snap –z DELETE "product_name=abc"
```

13. To gather debug data of multiple products, enter the following command:

```
snap –z "product_name=abc, product_name=def"
```

Files

Item	Description
/usr/sbin/snap	Contains the **snap** command.
/tmp/ibmsupt	Contains **snap** command output.
/tmp/ibmsupt/general/lslpp.hac	Contains the output of the **lslpp -hac** command, which is required to recreate exact operating system environments.
/tmp/ibmsupt/general/ general.snap	Contains general system information that is collected with the **snap -g** command.
/tmp/ibmsupt/testcase	Contains the test case that demonstrates your system problem.
/tmp/ibmsupt/other	Contains user-defined directory.

snapcore Command

Purpose

Gathers the **core** file.

Syntax

snapcore[**-d** *Dir*] [**-r**] core [*program*]

Description

The **snapcore** command gathers the **core** file, program, and libraries used by the program and compresses the information into a **pax** file. The file can then be downloaded to disk or tape, or transmitted to a remote system. The information gathered with the **snapcore** command is required to identify and resolve a problem with the application.

The **snapcore** command checks for available space in the **/tmp/snapcore** directory, the default directory for **snapcore** command output. You can write the output to another directory by using the **-d** flag. If there is not enough space to hold the **snapcore** command output, you must expand the file system.

Each execution of the **snapcore** command creates a new archive file. The archive file is named **snapcore_$pid.pax**. Use the **-r** flag to remove the previously created archive file. This command uses

$pid (pid of the **snapcore** command) to create a unique name file and preserve any previously created archives.

Specify the full path name for core and program. If the program name is not specified, **snapcore** reads the program name from the **core** file and searches for the location in directories contained in the *PATH* variable.

Flags

Item	Description
-d*Dir*	Identifies the optional **snapcore** command output directory (**/tmp/snapcore** is the default).
-r	Removes **snapcore** command output from the **/tmp/snapcore** directory.

Examples

1. To gather the **core** file, enter the following:
   a. `snapcore <core file name> <program name>`
   b. `snapcore <core file name>`

   Directories contained in the *PATH* variable are searched to find the program file. The **pax** file is created in **/tmp/snapcore** directory.
2. To clean the previously created core archive and create a new one, enter the following:

   ```
   snapcore -r<core file name> <program name>
   ```

   The **pax** file is created in **/tmp/snapcore** directory.
3. To create the **core** file archive in an alternate directory, enter the following:

   ```
   snapcore -d<dir name> <core file name> <program name>
   ```

   The **pax** file is created in **<dirname>/tmp/snapcore** directory.
4. To clean the **/tmp/snapcore** directory, enter the following:

   ```
   snapcore -r
   ```

Files

Item	Description
/usr/sbin/snapcore	Contains the **snapcore** command.
/tmp/snapcore	Contains **core** file archive.

snapshot Command

Purpose

Modify, create or view properties of enhanced journaled file system (JFS2) snapshots.

Syntax

To Create an External Snapshot

snapshot -o snapfrom=snappedFS snapshotLV

snapshot -o snapfrom=*snappedFS* **-o size=***Size*

To Create an Internal Snapshot

snapshot -o snapfrom=*snappedFS* **-n** *snapshotName*

To Delete an External Snapshot

snapshot -d *snapshotLV*

To Delete an Internal Snapshot

snapshot -d -n *snapshotName snappedFS*

To Query a JFS2 File System

snapshot -q [**-c***fieldSeparator*] *snappedFS*

To Query an External Snapshot

snapshot -q [**-c***fieldSeparator*] *snapshotLV*

To Query an Internal Snapshot

snapshot -q -n *snapshotName* [**-c***fieldSeparator*] *snappedFS*

To Modify an External Snapshot

snapshot -o size=*Size snapshotLV*

Note: The **snapshot** command does not support modifying internal snapshots. The size of an internal snapshot is limited by the amount of free space available in the file system itself.

Description

This command provides an interface to JFS2 snapshots.

The maximum number of external snapshots per file system is 15, while the maximum number of internal snapshots per file system is 64.

You cannot have both internal snapshot and external snapshot of a file system at the same time.

Flags

Item	Description
-c *fieldSeparator*	Specifies the output from the snapshot query to be displayed in colon format. The *fieldSeparator* is the character to use to separate the fields of the display.
-d	Deletes the snapshot and any previous snapshots. If the snapshot is an external snapshot, the logical volume containing the snapshot is also deleted unless you specify the **-s** flag. For an external snapshot, the *snapshotLV* parameter specifies the snapshot to delete. For an internal snapshot, the *snappedFS* parameter specifies the file system containing the snapshot to delete. The **-n** flag specifies the name of the snapshot to delete.
-n *snapshotName*	Specifies the access point for the internal snapshot under the *snappedFS*/.*snapshot*/*snapshotName*. If you specify the **-n** flag when creating a snapshot, the file system specified by the *snappedFS* parameter must be enabled for internal snapshots. Otherwise, an error message is displayed and no snapshot is created. To enable a file system to use internal snapshots, specify the **isnapshot** option when you create the file system with the **mkfs** command (**-o isnapshot={yes}**) or the **crfs** command (**-a isnapshot = {yes}**).

Item	Description
-o snapfrom=*snappedFS*	Creates a snapshot of the file system specified by the *snappedFS* parameter. If the **-n** flag is specified, an internal snapshot is created. If the *snapshotLV* parameter is specified, the logic volume must already exist and must be in the same volume group as the file system specified by the *snappedFS* parameter. If the specified logic volume is already in use as a snapshot or a file system known to the **/etc/filesystems** file, the command issues an error message and fails. If the **-n** flag and the *snapshotLV* parameter are not specified, a new logical volume is created for the external snapshot.
-o size=*Size*	Specifies the size of a new logical volume for an external snapshot when you specify this flag with the **-o snapfrom=***snappedFS* flag. Otherwise, this flag increases the size of the external snapshot specified by the *snapshotLV* field to the value of *Size*. This flag is ignored if any flag other given. If the *Size* field is followed by an **M**, the value is treated as megabytes. If the *Size* field is followed by a **G**, the value is treated as gigabytes. If neither **M** nor **G** are used, the value is treated as 512-byte blocks.
-q	Displays information about the specified snapshot or snapshots. Specifies the following flags and options to determine the query as needed: • Specify the **-n** flag to display information about the named internal snapshot belonging to the file system that is specified by the *snappedFS* parameter is displayed. The information includes the file system that the snapshot belongs to, and the time when the snapshot is taken. • Specify the *snapshotLV* parameter to display information about the external snapshot. The information includes the file system that the snapshot belongs to, the time when the snapshot is taken, the size of the snapshot storage object, and the remaining free space. • Specify the *snappedFS* parameter to display information about all of the snapshots for the file system specified by the *snappedFS* parameter. For external snapshots, the information includes each of the snapshots and their storage objects, the time when the snapshot is taken, the size of the snapshot storage objects, and the remaining free space. For internal snapshots, the information includes each of the snapshots and the time when the snapshot is taken.
-s	Retains the specified logical volume for the specified snapshot when the external snapshot is deleted.

Parameters

Item	Description
fieldSeparator	The character to use to separate the fields of the display.
snappedFS	The JFS2 file system to act on for snapshot creation, deletion, or query.
snapshotLV	The logical volume of the external snapshot.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To create a snapshot for the **/home/janet/sb** file system on the **/dev/snapsb** logical volume, enter the following:

   ```
   snapshot -o snapfrom=/home/janet/sb /dev/snapsb
   ```

   This command creates a snapshot for the **/home/janet/sb** file system on the **/dev/snapsb** logical volume, which already exists.
2. To create a snapshot for the **/home/janet/sb** file system, enter the following:

   ```
    snapshot -o snapfrom=/home/janet/sb -o size=16M
   ```

   This command creates a 16-megabyte logical volume and creates a snapshot for the **/home/janet/sb** file system on the newly created logical volume.
3. To view information about all of the snapshots for the **/home/janet/sb** file system, enter the following:

   ```
   snapshot -q /home/janet/sb
   ```

   This command displays each snapshot for the **/home/janet/sb** file system along with the time when the snapshot was taken, the size of the snapshot storage object, and the remaining free space.
4. To increase the size of the snapshot on the **/dev/snapsb** device, enter the following:

   ```
   snapshot -o size=64M /dev/snapsb
   ```

   This command increases the **/dev/snapsb** device to 64 megabytes along with the snapshot contained on the device.
5. To delete the snapshot on the **/dev/snapsb**device, enter the following:

   ```
   snapshot -d /dev/snapsb
   ```

   This command deletes the snapshot contained on the **/dev/snapsb** device and removes the **/dev/snapsb** logical volume .

snapsplit Command

Purpose

To split a `snap` output file into multiple smaller files of arbitrary or specified size.

Syntax

`snapsplit` [`-s` *size*] [`-H` *machinename*] [`-f` *filename*]

`snapsplit` `-u` `-T` *timestamp* [`-H` *machinename*]

Description

The `snapsplit` command is used to split a `snap` output file into smaller files. This command is useful for dealing with very large `snap` files. It breaks the file down into files of a specific size that are multiples of 1 megabyte. Furthermore, it will combine these files into the original file when called with the `-u` option.

The output files are named as following: `snap.`***machinename.timestamp***`.pax.Z.xxx`. *Machinename* is the hostname and *timestamp* is in the format MMDDYYHHMMSS. In addition, xxx represents the extension for the `split` files which is crucial when putting these files back together. The extensions from the start

of the files go in the following order: `xaa, xab, xac, xad, xae ..., xaz, xba, xbb, xbc, xbd, ..., xbz, xca, xcb, xcc, ....`

When performing `ls` on these files, the first file listed would represent the top of the original file and the last file, the end of the original file.

Note that this command should only be used for `snap` files that are `paxed` and compressed. When executed on local system where `snap` output was gathered, the `-H` option need not be used. That flag is provided for the case where user has moved a complete `snap` file to a remote system and wishes to split it. Any machine name may be selected, but it is recommended, to use the machine name where data was collected.

Flags

Item	Description
`-f` *filename*	Input `snapsplit` file. It should be a compressed `pax` file. The default is `snap.pax.Z`.
`-H` *machinename*	Name of the host machine. If none is specified, the default is the current host. Care must be exercised to name `snap` files for the appropriate system.
`-s` *size*	Specifies the size of `snap` output in multiples of 1 MB. The last file will be smaller or equal to this size. *Size* should be entered in megabytes. The default size is 1 MB.
`-T` *timestamp*	Timestamp of the `snapsplit` files to use in restoring the original `snap` output. It is in the format MMDDYYHHMMSS, where MM for month, DD for day, YY for year, HH for hours, MM is for minutes and SS is for seconds.
`-u`	Flag used for rejoining `snapsplit` files. Used with the `-T` flag.

Examples

1. To split the default snap file (`snap.pax.Z` should be in the current directory), enter the following:

   ```
   snapsplit
   ```

   The output of this command is written to current directory.
2. To split file `snap.somefile.pax.Z` from system doe, enter the following:

   ```
   snapsplit -H doe -f snap.somefile.pax.Z
   ```

   Note: The resulting files will be named `snap.doe.MMDDYYHHMMSS.pax.Z`.
3. To restore a file for which the `snap` files (`snap.sue.102303141211.xxx`) are for system sue, and timestamp `102303141211`, type:

   ```
   snapsplit -u -T 102303141211 -H sue
   ```

   Attention: If any one of the `snap` files is missing or has been renamed, the `snap` file created will corrupted.
4. To restore a `snap` file from files with time stamp `102603084512`, and which are for the current system, type:

   ```
   snapsplit -u -T 102603084512
   ```
5. To gather general system configuration information, including the output of the `lslpp -hBc` command, type the following:

   ```
   snap -g -o /dev/rfd0
   ```

   Output is written to the `/tmp/ibmsupt/general/lslpp.hBc` and `/tmp/ibmsupt/general/general.snap` files. This command also writes the system information to a removable diskette.

Files

Item	Description
`/usr/sbin/snapsplit`	Contains the `snapsplit` command.

snmpd Daemon

Purpose

Starts the Simple Network Management Protocol (SNMP) agent as a background process.

Syntax

Refer to the syntax for either the **snmpdv1** daemon or the **snmpdv3** daemon.

Description

/usr/sbin/snmpd is a symbolic link to to either the encrypted or non-encrypted version of the **snmpdv3** daemon which supports SNMP version 3.

Note: The encrypted version of the SNMP version 3 agent is available from the AIX Expansion Pack.

Files

Item	Description
/usr/sbin/snmpd	Contains a symbolic link to either **/usr/sbin/snmpdv1**, **/usr/sbin/snmpdv3e**, or **/usr/sbin/snmpdv3ne**.
/usr/sbin/snmpdv1	Contains the SNMP version 1 agent.
/usr/sbin/snmpdv3e	Contains the encrypted version of the SNMP version 3 agent.
/usr/sbin/snmpdv3ne	Contains the non-encrypted version of the SNMP version 3 agent.

snmpdv1 Daemon

Purpose

Starts the Simple Network Management Protocol (SNMP) version 1 agent as a background process.

Syntax

snmpd [**-c** *ConfigFile*] [**-d** *Level*] [**-f** *LogFile*] [**-S**]

Description

The **snmpd** command starts the SNMP daemon. This command may only be issued by a user with root privileges or by a member of the system group.

The SNMP daemon is a server that supports the standard Simple Network Management Protocol (SNMP) documented by RFC 1157 and the Management Information Base (MIB) as defined in RFC 1155 and RFC 1213. The SNMP daemon provides the following three functions:

- Receiving and authenticating SNMP requests from network monitors.
- Processing requests and returning results to the originating monitor.
- Sending trap notification to all hosts listed in the configuration file.

The SNMP daemon server keeps log messages in a file specified by the *LogFile* variable if the **-f** flag is used or in a log file specified in the configuration file. When the size of the log file exceeds the predefined maximum log file size, the **snmpd** command will rotate the log file by moving the old log file to another file as follows:

- LogFile.3 is deleted.
- LogFile.2 is moved to LogFile.3.
- LogFile.1 is moved to LogFile.2.
- LogFile.0 is moved to LogFile.1.
- LogFile is moved to LogFile.0.
- Logging continues in LogFile.

If logging is not directed from the **snmpd** command line with the **-f** flag, logging can be directed from the configuration file.

Supported set variables are:

- **sysContact**
- **sysName**
- **sysLocation**
- **ifAdminStatus**
- **atPhysAddress**
- **atNetAddress**
- **ipForwarding**
- **ipDefaultTTL**
- **ipRouteDest**
- **ipRouteNextHop**
- **ipRouteType**
- **ipNetToMediaPhysAddress**
- **ipNetToMediaNetAddress**
- **ipNetToMediaType**
- **snmpEnableAuthenTraps**
- **smuxPstatus**
- **smuxTstatus**

See "Understanding SNMP Daemon Support for SET Request Processing" in *AIX Version 6.1 Communications Programming Concepts* for more information on the supported set variables.

The following commands should be issued before the SNMP daemon is started:

- **ifconfig lo0 loopback**
- **startsrc -s inetd**

These commands are normally executed during system startup when the **/etc/rc.net** and **/etc/rc.tcpip** shell scripts are called. (The **snmpd** command can be placed in the **/etc/rc.tcpip** shell script.)

The **snmpd** daemon should be controlled using the System Resource Controller (SRC). Entering **snmpd** at the command line is not recommended.

Manipulating the snmpd Daemon with the System Resource Controller

The **snmpd** daemon is a subsystem controlled by the System Resource Controller (SRC). The **snmpd** daemon is a member of the **tcpip** system group. The **snmpd** daemon is enabled by default and can be manipulated by SRC commands.

Use the following SRC commands to manipulate the **snmpd** daemon:

Item	Description
startsrc	Starts a subsystem, group of subsystems, or a subserver. Issuing the **startsrc** command causes the **snmpd** command to generate a *coldStart* trap.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
refresh	Causes a subsystem or group of subsystems to reread the appropriate configuration file. Issuing a **refresh** command causes the **snmpd** daemon to generate a *warmStart* trap.
traceson	Enables tracing of a subsystem, group of subsystems, or a subserver. If the user issuing the **traceson** command is not the root user, the debugging level will not exceed level 2.
tracesoff	Disables tracing of a subsystem, group of subsystems, or a subserver.
lssrc	Gets the status of a subsystem, group of subsystems, or a subserver. If the user issuing the long status form of the **lssrc** command is not the root user, no community name information is displayed.

Flags

Item	Description
-c *ConfigFile*	Specifies full path and file name of the configuration file for the **snmpd** daemon. This file is read when the **snmpd** daemon starts up and when a **refresh** or **kill -1** signal is issued. If the **-c** flag is not specified, the default configuration file is **/etc/snmpd.conf**. See the **snmpd.conf** file for information on this file format.
-d *Level*	Specifies the level of tracing the **snmpd** command produces. The *Level* value can be one of: **0** All notices, exceptions, and fatal messages **1** Level 0 plus debug messages **2** Level 1 plus a hexadecimal dump of incoming and outgoing packets **3** Level 2 plus an English version of the request and response packets If the **-d** flag is not specified, the debugging level is set to 0.
-f *LogFile*	Specifies the full path and file name into which **snmpd** tracing information is logged. If the **-f** flag is not specified, no information will be logged. See the **snmpd.conf** file for more information on setting logging parameters.
-S	Enable the security option if it's specified. It will prevent the local non-root user from changing the value of MIB variable(s) on the local host.

Examples

1. To start the **snmpd** daemon, enter a command similar to the following:

   ```
   startsrc -s snmpd -a "-f /tmp/snmpd.log"
   ```

   This command starts the **snmpd** daemon and logs information to the **/tmp/snmpd.log** file at debug level 0.
2. To stop the **snmpd** daemon normally, enter:

   ```
   stopsrc -s snmpd
   ```

   This command stops the daemon. The **-s** flag specifies the subsystem that follows to be stopped.

3. To get short status from the **snmpd** daemon, enter:

```
lssrc -s snmpd
```

This command returns the name of the daemon, the process ID of the daemon, and the state of the daemon (active or inactive).

4. To get a long status from the **snmpd** daemon, enter:

```
lssrc -ls snmpd
```

If you are the root user, this long form of the status report lists the configured community names and associated access privileges and views for **snmp** requests. The long form also lists the community names associated with the hosts for trap notification, logging configuration parameters, **snmpd** specific configuration parameters and **smux** configuration parameters.

5. To enable tracing for the **snmpd** daemon, enter the following:

```
traceson -s snmpd
```

This command enables **snmpd** debugging if the **snmpd** daemon is configured for logging.

6. To view the contents of the DHCP Server database files **/etc/dhcpsd.ar** and **/etc/dhcpsd.cr**, enter:

```
lssrc -l -s dhcpsd
```

Files

Item	Description
/etc/services	Contains port assignments for required services. The following entries must be present in the **/etc/services** file if the entries are not already present: **snmp** 161/udp **snmp-trap** 162/udp **smux** 199/tcp **Requirements:** • The snmp port must be 161 as required by RFC 1157. • The snmp-trap port must be 162 as required by RFC 1157. • The smux port must be 199. • The **/etc/services** file is shipped with these entries already in place. • If the **/etc/services** file is being served from a server, these entries must be present in the server's **/etc/services** file.
/etc/snmpd.conf	Specifies the configuration parameters for the **snmpd** agent.
/etc/mib.defs	Defines the Management Information Base (MIB) variables the SNMP agent should recognize and handle.

snmpdv3 Daemon

Purpose

Starts the Simple Network Management Protocol (SNMP) version 3 agent as a background process.

Syntax

snmpd [**-d** *level*] [**-i** *interval*] [**-p** *port*] [**-S**] [**-c** *community*]

Description

The **snmpd** command starts the Simple Network Management Protocol (SNMP) daemon. This command may only be issued by a user with root privileges or by a member of the system group.

The SNMP daemon is a server that supports the all the SNMPv1, SNMPv2c, and SNMPv3 protocols documented by RFCs 1157, RFD 1905, and RFC 2572. It also behaves as a SMUX server as defined by RFC 1227 and as a Distributed Protocol Interface (DPI) version 2.0 agent as defined by RFC 1592. The SNMP daemon provides the following three functions:

- Receiving and authenticating SNMP requests from network monitors.
- Processing requests and returning results to the originating monitor.
- Sending trap notification to all hosts listed in the configuration file.

The SNMP daemon server stores log messages in a file specified by the *LogFile* variable if the **-f** flag is used or stores log messages in a log file specified in the configuration file. The maximum value for number of log files is 4. When the size of the log file exceeds the predefined maximum log file size, the **snmpd** command moves the old log file to another file as follows:

- LogFile.3 is deleted.
- LogFile.2 is moved to LogFile.3.
- LogFile.1 is moved to LogFile.2.
- LogFile.0 is moved to LogFile.1.
- LogFile is moved to LogFile.0.
- Logging continues in LogFile.

The following commands should be issued before the SNMP daemon is started:

- **ifconfig lo0 loopback**
- **startsrc -s inetd**

These commands are normally executed during system startup when the **/etc/rc.net** and **/etc/rc.tcpip** shell scripts are called. (The **snmpd** command can be placed in the **/etc/rc.tcpip** shell script.)

The **snmpdv3** daemon should be controlled using the System Resource Controller (SRC). Entering **snmpd** at the command line is not recommended.

Manipulating the snmpd Daemon with the System Resource Controller

The **snmpdv3** daemon is a subsystem controlled by the System Resource Controller (SRC). The **snmpdv3** daemon is a member of the **tcpip** system group. The **snmpdv3** daemon is enabled by default and can be manipulated by SRC commands.

Use the following SRC commands to manipulate the **snmpd** daemon:

Item	Description
startsrc	Starts a subsystem, group of subsystems, or a subserver. Issuing the **startsrc** command causes the **snmpdv3** command to generate a *coldStart* trap.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
lssrc	Gets the status of a subsystem, group of subsystems, or a subserver.

Flags

Item	Description
-d *level*	Specifies the level of tracing to be started. The valid values for level are 0-255. If the **-d** parameter is not specified, then the default level of 0 is used, meaning no tracing will be done. If the **-d** parameter is specified without a level, then a level of 31 is used, meaning all SNMP requests/responses/traps and DPI activity will be traced. There are 8 levels of tracing provided. Each level selected has a corresponding number. The sum of the numbers associated with each level of tracing selected is the value which should be specified as level. The numbers for the trace levels are: **0** No tracing. This is the default. **1** Trace SNMP responses, requests, and traps. **2** Trace DPI level 1 and DPI level 2. **3** Same as level 1 plus level 2 plus internal trace. **4** Same as trace level 3 plus extended trace.
-i *interval*	Specifies the interval (in minutes) at which dynamic configuration changes to the SNMP agent should be written out to the **/etc/snmpdv3.conf** configuration file. Valid values are 0-10. The default value is 5. This parameter is only relevant when the **/etc/snmpdv3.conf** file is used for SNMPv3 configuration.
-p *port*	Listens for SNMP packets on this port. The default is port 161.
-S	Prevents non-root users from changing the MIB values.
-c *community*	Accepts the requests with the community name that the *community* parameter specifies.

Examples

1. To start the **snmpd** daemon, enter a command similar to the following:

   ```
   startsrc -s snmpd
   ```

   This command starts the **snmpd** daemon at debug level 0.
2. To stop the **snmpd** daemon normally, enter:

   ```
   stopsrc -s snmpd
   ```

   This command stops the daemon. The **-s** flag specifies the subsystem that follows to be stopped.
3. To get status from the **snmpd** daemon, enter:

```
lssrc -s snmpd
```

This command returns the name of the daemon, the process ID of the daemon, and the state of the daemon (active or inactive).

Files

Item	Description
/etc/services	Contains port assignments for required services. The following entries must be present in the **/etc/services** file if the entries are not already present: **snmp** 161/udp **snmp-trap** 162/udp **smux** 199/tcp
/etc/snmpdv3.conf	Specifies the configuration parameters for the **snmpdv3** agent.
/etc/snmpd.boots	Specifies the engineID and the engineBoots for the **snmpdv3** agent.
/etc/mib.defs	Defines the Management Information Base (MIB) variable the SNMP agent should recognize and handle.

snmpevent Command

Purpose

Sends ERRM events to an SNMP agent.

Syntax

`snmpevent` [`-a` *host-name*] [`-c` *community*] [`-h`]

Description

The `snmpevent` script sends a Simple Network Management Protocol (SNMP) trap of an event response resource manager (ERRM) event to a host running an SNMP agent. The agent formats the trap information into an SNMP trap and sends it to the SNMP manager defined in its configuration file. This script is meant to be called by the predefined ERRM response `Generate SNMP trap`. Event or rearm event information is captured and posted by ERRM in environment variables that are generated when an ERRM event or a rearm event occurs.

The `snmpevent` script can also be used as a template to create other user-defined actions. See the *RSCT Administration Guide* to understand how an event response resource runs an action command.

The following message template is sent as a trap when an event or a rearm event occurs and `snmpevent` is the defined response:

```
[ERRM_COND_SEVERITY] [ERRM_TYPE] occurred:
          Condition: [ ERRM_COND_NAME]
          Node:  [ERRM_NODE_NAME]
          Resource: [ERRM_RSRC_NAME]
          Resource Class:  [ERRM_RSRC_CLASS_NAME]
          Resource Attribute: [ERRM_ATTR_NAME]
```

```
            Attribute Type: [ERRM_DATA_TYPE]
            Attribute Value: [ERRM_VALUE]
```

The environment variables have the following definitions:

ERRM_COND_SEVERITY
Specifies the significance of the condition resource that caused the event or rearm event. The valid values are: Critical, Warning, or Informational.

ERRM_TYPE
Specifies the type of event that occurred. The valid values are: event or rearm event.

ERRM_COND_NAME
Specifies the name of the condition resource with the attribute value that changed to cause this event or rearm event.

ERRM_NODE_NAME
Specifies the host name on which this event or rearm event occurred.

ERRM_RSRC_NAME
Specifies the name of the resource with the attribute that changed to cause this event or rearm event.

ERRM_RSRC_CLASS_NAME
Specifies the name of the resource class to which the resource that caused this event or rearm event belongs.

ERRM_ATTR_NAME
Specifies the name of the resource attribute that changed to cause this event or rearm event.

ERRM_DATA_TYPE
Specifies the data type of the resource attribute.

ERRM_VALUE
Specifies the value of the resource attribute that changed to cause this event or rearm event.

The `snmpevent` command captures these environment variable values and formats a generic message that is sent as a trap via a call to the `snmptrap` command.

Flags

-a *host-name*
Specifies the host name of the SNMP agent to which the AIX subagent will connect. By default, the subagent will connect to the SNMP agent running on the local node.

-c
Specifies the SNMP community to be used. This can be any string the SNMP agent will accept. The default is `public`.

-h
Writes this script's usage statement to standard output.

Parameters

log_file
Specifies the name of the file where event information is logged. An absolute path for the *log_file* parameter should be specified.

The *log_file* is treated as a circular log and has a fixed size of 64KB. When *log_file* is full, new entries are written over the oldest existing entries.

If *log_file* already exists, event information is appended to it. If *log_file* does not exist, it is created so that event information can be written to it.

Exit Status

0
The script has run successfully.

1
An error occurred when the script was run.

Restrictions

This script must be run on the node where the ERRM is running.

Standard Output

When the `-h` flag is specified, this script's usage statement is written to standard output.

Examples

1. Suppose the command `/opt/rsct/bin/snmpevent` is an action in the critical-notification response, which is associated with the CSM predefined condition `NodeChanged`. This can be done with the `mkcondresp` command followed by the `startcondresp` command. The `/etc/snmpdv3.conf` file should be configured to where the trap will be sent. In this example, if you want the trap sent to `9.117.16.246`, write the `/etc/snmpdv3.conf` file as follows:

```
VACM_GROUP group1 SNMPv1 public -

VACM_VIEW defaultView        internet                    - included
-VACM_ACCESS  group1 - - noAuthNoPriv SNMPv1  defaultView - defaultView -

NOTIFY notify1 traptag trap -

#TARGET_ADDRESS Target1 UDP 127.0.0.1       traptag trapparms1 - - -
TARGET_ADDRESS Target1 UDP 9.117.16.246       traptag trapparms1 - - -

TARGET_PARAMETERS trapparms1 SNMPv1  SNMPv1  public  noAuthNoPriv -

COMMUNITY public    public     noAuthNoPriv 0.0.0.0     0.0.0.0         -

DEFAULT_SECURITY no-access - -

logging         file=/usr/tmp/snmpdv3.log       enabled
logging         size=0                          level=0

smux            1.3.6.1.4.1.2.3.1.2.1.2         gated_password  # gated

snmpd smuxtimeout=200 #muxatmd
smux 1.3.6.1.4.1.2.3.1.2.3.1.1 muxatmd_password #muxatmd
```

   Then, restart the `snmpd` daemon by first killing the `snmpd` daemon that is currently running and then starting it again:

```
# ps -ef | grep snmpd
    root  4570 12956   1 08:24:32  pts/0  0:00 grep snmpd
    root 13810     1   0 08:11:04      -  0:00 snmpd
# kill -9 13810
# snmpd
```

   Next, change the LParID property of node c175n08 to 12:

```
# chnode c175n08 LParID=12
```

   Now, on the node `9.117.16.158` (the node with the SNMP manager that was specified in the `/etc/snmpdv3.conf` file), the SNMP manager should record something like this:

```
2002-07-15 09:09:25 c174tr1.ppd.pok.ibm.com [9.114.78.17] TRAP, SNMP v1,
community public
        enterprises.ibm Enterprise Specific Trap (1) Uptime: 0:01:45.00
        enterprises.ibm.ibmProd.191.1.6.1.0 = "Informational Event
occurred. Condition=NodeChanged Node=c174tr1.ppd.pok.ibm.com
Resource=c175n08.ppd.pok.ibm.com Resource Class=Node Resource
Attribute=Changed Attributes Attribute Type=CT_CHAR_PTR_ARRAY Attribute
Val={LParID} "
```

   The output varies based on SNMP managers.

Location

/opt/rsct/bin/snmpevent

snmpinfo Command

Purpose

Requests or modifies values of Management Information Base (MIB) variables managed by a Simple Network Management Protocol (SNMP) agent.

Syntax

The get or next Option

snmpinfo [**-m get** | **next**] [**-v**] [**-c** *Community*] [**-d** *Level*] [**-h** *HostName*] [**-o** *ObjectsFile*] ... [**-t** *Tries*] [**-w** *Waittime*] *Variable*. *Instance* ...

The set Option

snmpinfo -m set [**-v**] [**-c** *Community*] [**-d** *Level*] [**-h** *HostName*] [**-o** *ObjectsFile*] ... [**-t** *Tries*] [**-w** *Waittime*] *Variable* . *Instance*= *Value* ...

The dump Option

snmpinfo -m dump [**-v**] [**-c** *Community*] [**-d** *Level*] [**-h** *HostName*] [**-o** *ObjectsFile*] ... [**-t** *Tries*] [**-w** *Waittime*] [*Variable*. *Instance*] ...

Description

The **snmpinfo** command requests or modifies values for one or more MIB variables for an SNMP agent. This command may only be issued by a user with root privileges or by a member of the system group.

If the you specify the **get** option, the **snmpinfo** command requests information about one or more MIB variables from an SNMP agent.

If you specify the **next** option, the **snmpinfo** command requests information from an SNMP agent about the instances following the specified instances. The **next** option makes it possible to obtain MIB values without knowledge of the instance qualifiers.

If you specify the **set** option, the **snmpinfo** command modifies values for one or more MIB variables for an SNMP agent. Only a few MIB variables are designated read-write. The agent that manages the MIB database may take various actions as a side effect of modifying MIB variables. For example, setting the **ifAdminStatus** MIB variable to 2 will normally shut down a network interface. The action taken is determined by the implementation of the SNMP agent that manages the database.

If you specify the **dump** option, the **snmpinfo** command can be used to traverse the entire MIB tree of a given agent. If a group is passed in as the *Variable* parameter, the **snmpinfo** command will traverse that specified path of the MIB tree.

The **snmpinfo** command has a debug facility that will dump debug information for transmitted and received packets. The facility is enabled with the **-d** flag.

Parameters

Item	Description
Value	Specifies the value to which the MIB *Variable* parameter is to be set. A value must be specified for each variable. If a value is not specified, the request packet will be invalid.
Variable	Specifies the name in text format or numeric format of a specific MIB variable as defined in the **/etc/mib.defs** file. If the option to the **-m** flag is **next** or **dump**, the *Variable* parameter may be specified as a MIB group.

Item	Description
Instance	Specifies the instance qualifier for the MIB *Variable* parameter. The *Instance* parameter is required if the option to the **-m** flag is **get** or **set**. The *Instance* parameter is optional if the option to the **-m** flag is **next** or **dump**.

Note:

1. There should be no blank spaces in the *Variable.Instance* parameter sequence.
2. If the *Instance* parameter is not specified, do not place a . (dot) after the *Variable* parameter.

For further information, consult RFC 1213, which defines the Management Information Base (MIB) for network management, and RFC 1157, which defines the SNMP protocol for creating requests for MIB information and formatting responses.

Flags

Item	Description
-c *Community*	Specifies the community name to be used to query the SNMP agent. If the **-c** flag is not specified, the default community name is **public**.
-d *Level*	Specifies the level of I/O debug information. The *Level* value can be one of: **0** No debug information. **1** Port bindings and the number of bytes transmitted and received. **2** Level 1 plus a hexadecimal dump of incoming and outgoing packets. **3** Level 2 plus an English version of the request and response packets. If the **-d** flag is not specified, the default debug level is 0.
-h *HostName*	Specifies the host name of the SNMP agent to be queried. The host name can be an IPv4 address, an IPv6 address, or a host name. If the **-h** flag is not specified, the default host name is the host name of the machine on which the user is currently logged in.
-m *Option*	Specifies the mode by which to access the MIB variables. The *Option* value can be one of: **get** Requests information about the specified MIB variables. **next** Requests the instances following the specified instances. **set** Modifies the specified write access MIB variables. **dump** Dumps the specified section of the MIB tree. **Note:** 1. The option name can be specified by the minimum number of characters required to make it unique. 2. If the **-m** flag is not specified, the default mode is **get**.

Item	Description
-o *ObjectsFile*	Specifies the name of the objects definition file that defines the MIB objects the **snmpinfo** command can request. If the **-o** flag is not specified, the default objects definition file name is **/etc/mib.defs**. See the **mosy** command for information on creating this file. More than one *ObjectsFile* can be referenced with the restriction that files containing parent definitions be specified before files containing child definitions.
-t *Tries*	Specifies the number of times the **snmpinfo** command transmits the SNMP request to the SNMP agent before terminating with the message no SNMP response. If the **-t** flag is not specified, the default number of tries is 3.
-v	Specifies that the output from the **snmpinfo** command be displayed in verbose mode. If the **-v** flag is not specified, the information will not be displayed in verbose mode.
-w	Specifies the wait time in seconds for the response from the **snmpd** agent. If the **-w** flag is not specified, the default wait time is 15 seconds.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Limitation

When the **snmpdv3** daemon encounters a **SMI-v2** data type MIB while processing a SNMPv1 protocol request from **snmpinfo** manager, it will skip the MIB until it encounters a **SMI-v1** data type MIB.

Work around

The **clsnmp** manager should be configured with **SNMPv2** type requests or **SNMPv3** type requests to dump all of the MIB variables with the **snmpdv3** daemon.

Examples

1. To get the values for the MIB variable ifDescr.1, for the interface associated with ifIndex.1 and SysDescr, enter:

   ```
   snmpinfo -m get -v sysDescr.0 ifDescr.1
   ```

   In this example, the **-m get** flag specifies that the **snmpinfo** command should retrieve the value of MIB variables ifDescr.1, (the interface description for the interface associated with the ifIndex.1), and sysDescr.0 (the system description of the local host).
2. To get the value for the MIB variable following the **ipAdEntIfIndex** MIB variable for the host specified by IP address 192.100.154.1, enter:

   ```
   snmpinfo -m next -v 1.3.6.1.2.1.4.20.1.2.192.100.154.1
   ```

   In this example, the **-m next** flag specifies that the **snmpinfo** command should retrieve the information for the MIB variable **ifAdEntIfIndex**.192.100.154.1.
3. To get the value of the first MIB variable in the system group, enter:

   ```
   snmpinfo -m next -v -h giants system
   ```

   In this example, the **-m next** flag specifies that the **snmpinfo** command should retrieve the information for the MIB variable following the system group, which is sysDescr.0; the **-v** flag

indicates verbose mode; the **-h** flag indicates that the agent to be queried is `giants;` the group to retrieve information from is `system`.

4. To set the value of a MIB variable, enter a command similar to the following:

```
snmpinfo -m set -v -h giants -c monitor -t 2 ifAdminStatus.1=2
```

   In this example, the MIB **ifAdminStatus** variable is set to 2, or down, for the interface associated with `ifIndex.1` on the host known as `giants`. The **-c** flag specifies the community for the host. The **-t 2** flag specifies that the **snmpinfo** command will transmit the SNMP request to the SNMP agent 2 times before terminating if no response is received from the SNMP agent.

5. To dump a group of the MIB tree in verbose mode, enter a command similar to the following:

```
snmpinfo -m dump -v interfaces
```

   In this example the `interfaces` group is dumped in verbose mode.

6. To dump the entire MIB tree, enter:

```
snmpinfo -m dump
```

7. To get the values for the `sysName.0` MIB variable, enter:

```
snmpinfo -m get -v -h 2000:1:1:1:209:6bff:feae:6d67 sysName.0
```

   In this example, the **-m** get flag specifies that the **snmpinfo** command should retrieve the value of the `sysName.0` MIB variables. The **-v** flag indicates verbose mode. The **-h** flag indicates that the agent to be queried is an IPv6 address.

Files

Item	Description
/etc/mib.defs	Defines the Management Information Base (MIB) variables the SNMP agent should recognize and handle.

snmpmibd Daemon

Purpose

Starts the **snmpmibd** Distributed Protocol Interface (DPI) version 2 sub-agent daemon as a background process.

Syntax

snmpmibd [**-f** *file*] [**-d** [*level*]] [**-h** *hostname*] [**-c** *community*]

Description

The **snmpmibd** command starts the **snmpmibd** Distributed Protocol Interface (DPI) version 2 (**dpi2**) sub-agent. This command may only be issued by a user with root privileges or by a member of the system group.

The **snmpmibd** daemon complies with the standard Simple Network Management Protocol (SNMP) DPI version 2.0 defined by RFC 1592. It acts as a **dpi2** sub-agent to communicate with the **dpi2** agent through dpiPortForTCP.0 (1.3.6.1.4.1.2.2.1.1.1.0) which is defined in RFC 1592 section 3.1.

The Management Information Base (MIB) is defined by RFC 1155(SMIv1) and RFC 2578(SMIv2).

The specific MIB variables that the **snmpmibd** command is managing are defined by the following RFCs:

RFC 1213
MIB-II

RFC 1229
Extension to the Generic-Interface MIB

RFC 1231
IEEE 802.5 Token Ring MIB

RFC 1398
Ethernet-like Interface Types MIB

RFC 1512
FDDI MIB

RFC 4022
MIB for the Transmission Control Protocol (TCP)

RFC 4113
MIB for the User Datagram Protocol (UDP)

RFC 4292
IP Forwarding Table MIB

RFC 4293
Management Information Base for the Internet Protocol (IP)

Note: The "**system**" and "**snmp**" groups defined in RFC1213 are not implemented by **snmpdmibd** daemon. Instead they are implemented by **snmpdv3** agent.

For the **RFC 4292**, read-only access is provided to the variables.

For the **RFC 4293**, read and write access is provided to the **ipv6IpForwarding** variable and the **ipv6IpDefaultHopLimit** variable. Read-only access is provided to the other MIB variables. Both the server and the agent must use the **SNMP v2c** protocol or later, because some variables defined in this RFC cannot be accessed using the **SNMP v1** protocol.

The **snmpmibd** daemon is normally executed during system startup when **/etc/rc.tcpip** shell script is called.

The **snmpmibd** daemon should be controlled using the System Resource Controller (SRC). Entering **snmpmibd** at the command line is not recommended.

Use the following SRC commands to manipulate the **snmpmibd** daemon:

startsrc
Starts a subsystem, group of subsystems, or a subserver.

stopsrc
Stops a subsystem, group of subsystems, or a subserver.

refresh
Causes a subsystem or group of subsystems to reread the appropriate configuration file.

lssrc
Gets the status of a subsystem, group of subsystems, or a subserver. If the user issuing the long status form of the **lssrc** command is not the root user, no community name information is displayed.

Flags

Item	Description
-c *community*	Uses specified community name. If -c flag is not specified, the default community name is **public**.

Item	Description
-d [*level*]	Specifies tracing/debug level. The levels are: **8** DPI level 1 **16** DPI level 2 **32** Internal level 1 **64** Internal level 2 **128** Internal level 3 Adds the numbers for multiple trace levels. If **-d** flag is specified and the *level* is not specified, the default level is 56. If **-d** flag is not specified, the default level is 0.
-f *file*	A non-default configuration file. If the -f flag is not specified, the default configuration file is **/etc/snmpmibd.conf**. See **/etc/snmpmibd.conf** file for information on this file format.
-h *hostname*	Sends request to specified host. The value of the *hostname* attribute can be an IPv4 address, an IPv6 address, or a host name. If the **-h** flag is not specified, the default destination host is loopback (127.0.0.1).

Examples

1. To start the **snmpmibd** daemon, enter a command similar to the following:

   ```
   startsrc -s snmpmibd -a "-f /tmp/snmpmibd.conf"
   ```

   This command starts the **snmpmibd** daemon and reads the configuration file from **/tmp/snmpmibd.conf**.
2. To stop the snmpmibd daemon normally, enter:

   ```
   stopsrc -s snmpmibd
   ```

   This command returns the name of the daemon, the process ID of the daemon, and the state of the daemon (active or inactive).
3. To get long status from the **snmpmibd** daemon, enter:

   ```
   lssrc -ls snmpmibd
   ```

   If you are the root user, this long form of the status report lists the configuration parameters in **/etc/snmpmibd.conf**.

Files

Item	Description
/etc/snmpmibd.conf	Defines the configuration parameters for snmpmibd command.
/etc/mib.defs	Defines the Management Information Base (MIB) variables the SNMP agent and manager should recognize and handle.

snmptrap Command

Purpose

Generate a notification (trap) to report an event to the SNMP manager with the specified message.

Syntax

snmptrap [**-a** *host*] [**-h** *targethost*] [**-c** *community*] [**-o** *oid*] [**-d**] **-m** *message*

Description

Generate a notification (trap) to report an event to the SNMP manager with the specified message.

Flags

Item	Description
-a *host*	Specifies to connect to the SNMP agent on the specified host. If the **-a** flag is not specified, the default host is the local host. *host* can be an IPv4 address, an IPv6 address, or a host name.
-c *community*	Specifies community name to use. This community must have been set in **/etc/snmpdv3.conf** for SNMP version 3 or in **/etc/snmpd.conf** for SNMP version 1 and have the read access privilege at least to the SNMP agent running on the specified host or local host. If the **-c** flag is not specified, the default community name is "public".
-o *oid*	Specifies the event that generates the trap message. The *oid* specified, it will be used in the trap packet. If the parameter is not specified, the default OID is used in the trap packet. This specified OID is not validated for its correctness.
-d	Enables the debug facility
-h *targethost*	Specifies the target network manger host to which the trap message will be sent. The target host can be an IPv4 address, an IPv6 address, or a host name.The **-h** flag is different from the -a flag. The -a flag specifies a host where the AIX SNMP agent (snmp) must be running and the SNMP agent forwards this trap to network mangers. However, the -h flag does not require the AIX SNMP agent to forward the trap message to network managers, and it sends the trap directly to the network manager. If there are no -h and -a flags, the trap will be sent to the AIX SNMP agent on the local host.
-m *message*	Defines the message that the **snmptrap** command will send. *message* specifies the information the trap will hold. This information is in the text format. The **-m** flag must be the last flag specified.

Exit Status

0
Trap information was sent out correctly.

1
This indicates something was wrong during the process.

Examples

1. To send a trap with the message 'hello world' to the SNMP agent running on the local host, enter the following:

```
snmptrap -m hello world
```

Note: The community, public, must have read access to the SNMP agent running on the local host. For details, please refer to SNMP configuration documentation.

2. To send a trap with the community name, community1, and the message 'hello world' to the SNMP agent running on a remote host blah, enter the following:

```
snmptrap -c community1 -h blah -m hello world
```

Note: The community 'community1' must have read access to the SNMP agent running on the host 'blah'. For details, please refer to the SNMP configuration documentation.

3. To send a trap to the network manager running on a Linux platform and where the host name is `nehcyg`, type the following:

```
snmptrap -h nehcyg -m hello world
```

4. To send a trap to the network manager running on a Linux platform where the host name is *nehcyg*, and with the OID 1.3.6.1.4.1.2.6.191.1.6.1.0, enter the following:

```
snmptrap -h nehcyg –o 1.3.6.1.4.1.2.6.191.1.6.1.0 -m hello world
```

5. To send a trap with the `community1` community name, and the message `hello world` to the SNMP agent that is running on an IPv6 address, enter the following command:

```
snmptrap -c community1 -h 2000:1:1:1:209:6bff:feae:6d67 -m hello world
```

Note: The `community1` community must have read access to the SNMP agent that is running on the IPv6 address. For more information, see SNMP for network management.

6. To send a trap to the network manager that runs on an IPv6 address, and with the OID `1.3.6.1.4.1.2.6.191.1.6.1.0`, enter the following command:

```
snmptrap -h 2000:1:1:1:209:6bff:feae:6d67 -o 1.3.6.1.4.1.2.6.191.1.6.1.0 -m hello world
```

Files

Item	Description
/etc/snmpdv3.conf	Contains the configuration file for the SNMP version 3 agent.
/etc/snmpd.conf	Contains the configuration file for the SNMP version 1 agent.

snmpv3_ssw Command

Purpose

Switch the symbolic links among the non-encrypted **snmpdv3** agent, encrypted **snmpdv3** agent and **snmpdv1** agent.

Syntax

snmpv3_ssw [**-e** | **-n** | **-1**]

Description

Switch the symbolic links among the non-encrypted snmpdv3 agent, encrypted snmpdv3 agent and snmpdv1 agent, and then start the newly chosen SNMP agent. A user can choose which version of SNMP agent to run.

For example, if the current running SNMP agent is the encrypted **snmpdv3** agent, the actual SNMP agent executable which is running on the machine is "**/usr/sbin/snmpdv3e**". The symbolic links on the machine are:

- /usr/sbin/snmpd --> /usr/sbin/snmpdv3e
- /usr/sbin/clsnmp --> /usr/sbin/clsnmpe

If a user chooses to switch to the non-encrypted snmpdv3 agent, after user runs the **/usr/sbin/snmpv3_ssw** command with the **-n** option, the actual snmp agent which is running on the machine "**/usr/sbin/snmpdv3ne**". The symbolic links on the machine will be changed to:

- /usr/sbin/snmpd --> /usr/sbin/snmpdv3ne
- /usr/sbin/clsnmp --> /usr/sbin/clsnmpne

Flags

Item	Description
-e	Switch to the encrypted version of snmpdv3 agent.
-n	Switch to the non-encrypted version of snmpdv3 agent.
-1	Switch to the snmpdv1 agent.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To switch to the encrypted version of **snmpdv3** agent, enter:

```
/usr/sbin/snmp3_ssw -e
```

sno Command

Purpose

Provides a SNOBOL interpreter.

Syntax

sno [*File* ...]

Description

The **sno** command provides a SNOBOL compiler and interpreter, with some differences from standard SNOBOL. It reads the named files and the standard input and compiles all input through a statement containing the **end** label. The rest is available to the **syspit** pseudo-variable.

The **sno** command differs from standard SNOBOL in the following ways:

- There are no unanchored searches. To get the same effect, use lines similar to the following:

Item	Description
`a ** b`	Produces an unanchored search for *b*.
`a *x* b = x c`	Produces an unanchored assignment.

- There is no back referencing.

```
x = "abc"
```

Item	Description
`a *x* x`	Produces an unanchored search for **abc**.

- Function declaration is done at compile time by the use of the (non-unique) **define** label. Execution of a function call begins at the statement following the **define** label. Functions cannot be defined at run time, and the use of the name **define** is preempted. There is no provision for automatic variables other than parameters. For example:

```
define f()
define f(a, b, c)
```

- All labels except **define** (even **end**), must have a nonempty statement.
- Labels, functions, and variables must all have distinct names. In particular, the nonempty statement on **end** cannot merely name a label.
- If **start** is a label in the program, program execution begins there. If not, execution begins with the first executable statement. The **define** label is not an executable statement.
- There are no built-in functions.
- Parentheses for arithmetic are not needed. Normal precedence applies. Because of this, the arithmetic operators **** (backslash) and ***** (asterisk) must be set off by spaces.
- The right side of assignments must be nonempty.
- Either **'** (single quotation mark) or **"** (double quotation mark) can be used for literal quotation marks.
- The pseudo-variable **sysppt** is not available.

Examples

To run the file `test.s` through the **sno** command and direct the output into the file output, enter:

```
sno < test.s > output
```

Files

Item	Description
/usr/bin/sno	Contains the **sno** command.

sntp4 Command

Purpose

The **sntp4** command queries a Network Time Protocol (NTP) server and displays the offset time of the system clock with respect to the server clock.

Syntax

sntp [**-h | -help | -?**] [**-v** | **-V** | **-W**] [**-q** [**-f** *savefile*] | [{**-r** | **-a**} [**-P** prompt] [**-l** *lockfile*]] [**-c** *count*] [**-e** *minerr*] [**-E** *maxerr*] [**-d** delay | **-x** [*separation*] [**-f** *savefile*]] [**-4** | **-6**] [**-u**] [*address(es)*]]

Description

The **sntp4** command is a Simple Network Time Protocol (SNTP) client used to query a Network Time Protocol (NTP) server and displays the offset time of the system clock with respect to the server clock. If you execute the **sntp4** command logged in as a **root** to the system, the **sntp4** command corrects the system offset time. The **sntp4** command can be executed as an interactive command or from a script such as the**cron** job. The **sntp4** command implements the SNTP protocol defined in the RFC-2030, which is a subset of the NTP protocol defined in the RFC-1305. The **sntp4** command does not provide the full NTP implementation features such as sanity checks, access controls, security functions, and mitigation algorithms.

Note: Do not use the **sntp4** command for operating the system as a primitive server in a public time server network. The **sntp4** command man page located at the **./sntp** directory provides the complete disclosure. The disclosure mentions that the RFC-2030 forbids an SNTP client to operate as a server for NTP or SNTP clients. If such an operation is contemplated, do not allow access by clients on the public network.

By default the **sntp4** command displays the local date and time to the standard output in the following format:

```
1996 Oct 15 20:17:25.123 + 4.567 +/- 0.089 secs
```

where, `+ 4.567 +/- 0.089 secs` indicates the time offset and error bound of the system clock with respect to the server clock.

If the NTP server address is explicitly specified in the **sntp4** command, the **sntp4** command sends a single message to the server and waits up to *delay* seconds for a unicast server message. If the NTP server address is explicitly specified in the **sntp4** command, the **sntp4** command does not send a message to the server and waits up to *delay* seconds for a broadcast server message.

Flags

Item	Description
-4	Forces IP version 4 DNS resolution.
-6	Forces IP version 6 DNS resolution.
-a	Slews the system clock to the correct time by using the UNIX *adjtime* system call. This option requires the **root** privilege.
-c *count*	Sets the maximum number of NTP packets required to *count*. The acceptable values for this option ranges from 1 to 25 in unicast mode, and 5 to 25 in broadcast mode. The default value is 5 in unicast mode, and broadcast mode.
-d *delay*	Sets the maximum waiting time in broadcast mode to *delay* seconds. The acceptable values for this option ranges from 1 to 3600. The default value is 15 in unicast mode, and the default value is 300 in broadcast mode.
-e *minerr*	Sets the minimum offset to *minerr* seconds. The measured offset values lesser than the values set by this option are ignored. The acceptable values for this option ranges from 0.001 to 1 in unicast mode The default value is 0.1 in unicast mode, and the default value is 0.5 in broadcast mode.
-E *maxerr*	Sets the maximum offset to *maxerr* seconds. The measured offset values greater than the values set by this option are ignored. The acceptable values for this option ranges are from 1 to 60. The default value is five.

Item	Description
`-f` *savefile*	Stores records of previous packets when used with the **-x** option, which speeds up re-calculating the drift after SNTP has to be re-started (e.g. because of network or server outages). In order to restart the data, **sntp** must be restarted reasonably soon after it died (within a few times the value of separation), with the same value of the **-c** option, the same value of separation, and in the same mode (i.e. broadcast or client), though the NTP servers need not be the same for client mode, and with compatible values of other settings. Note that the file will be created with the default ownerships and permissions, using standard C facilities. The default is installation-dependent, but will usually be in the **/etc/sntp.state** file.
-h, **-help**	Displays usage information.
`-l` *lockfile*	Sets the name of the *lockfile* to ensure that there is only one instance of the SNTP running at a time. The default value is installation dependent and is specified in the **/etc/sntp.pid** file.
`-P` *prompt*	Sets the maximum automatic offset value to *maxerr* seconds. The acceptable values ranges from 1 to 3600, or no. The default value is 30. If the **sntp4** command is run interactively, the measured offset values greater than 30 will prompt the user for confirmation. Specifying no will disable this and the correction will be made regardless.
-q	Indicates that the **sntp4** command should query a daemon *savefile* maintained by the SNTP. This option does not require any privileges. The option does not modify the *savefile* nor the system clock.
-r	Steps the system clock to the correct time of the UNIX *settimeofday* system call. This option requires the root privilege.
`-u`	Uses an unprivileged port.
`-v`	Writes diagnostic messages and a limited amount of tracing to standard error. The **-v**, **-V** and **-W** give increasing levels of detail.
`-x` *separation*	Causes the program to run as a daemon (i.e. forever), and to estimate and correct for the clock drift. separation sets the minimum time between calls to the server in minutes if a NTP host is specified, and between broadcast packets if not. Acceptable values are from 1 to 1440 (a day), and the default (if **-x** is specified but separation is omitted) is 300.

Parameters

Item	Description
address	NTP server address.

Exit status

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: The user must be a member of the system group.

Files

Item	Description
/usr/sbin/ntp4/sntp4	Contains the **sntp** command
/usr/sbin/sntp --> /usr/sbin/ntp3/sntp	Default Symbolic link to NTP version 3 binaries from **/usr/sbin** directory.

Example

To get the time offset of the system clock relative to the server (9.41.254.24) clock, enter the following command:

```
sntp 9.41.254.24
```

The following output appears:

```
2009 Feb 25 12:28:38.00620 - 0.00679 +/- 0.31077 secs
```

sodebug Command

Purpose

Sets or unsets the socket debug flag (**SO_DEBUG** socket option) and trace level on sockets.

Syntax

sodebug [-h] [-l [*level*]] [-p *pid* | -s *sockaddr* [-t *type*]]

Description

The sodebug command sets, unsets, or lists the socket debug flag and trace level on active sockets

If the socket debug flag (also known as the **SO_DEBUG** socket option) is set for a socket, the events on this socket can be traced using the `trace` command.

You can use the `-l` option to set the socket debug flag on sockets that already exist on a system. The `-l` option also sets the trace level for a given socket.

If the sodebug command is run without any options, the socket debug flag status and trace level for each active socket displays.

The `trace` and `trpt` commands collect information based on the trace level.

The following table describes the information collected based on the trace level for trace hook ID 25A (TCPDBG):

	min	normal	detail
tcp_debug data (td_time, td_act, td_ostate, td_tcb, family and td_req)		X	X
tcpip header		X	X
Address of tcpcb		X	X
All tcpcb fields			X
Address of socket		X	X

	min	normal	detail
All socket fields			X

You can also set or unset the socket debug flag and the trace level as described below:

1. The following command enables the socket debug flag for all sockets that are subsequently created on the system:

   ```
   no -o sodebug=1
   ```

2. You can specify **|DEBUG**[**=***level*] in the wait/nowait field of a service in inetd.conf to turn on socket debugging for a specific service. You can set the trace level to **min**, **normal**, or **detail**. If no level is specified, the default level is **normal**.
3. You can set socket debugging on or off for all subsequent sockets created by a process using the **sodebug_env** parameter of the no command and specifying **export SODEBUG=***level* in a process environment. You can set the trace level to **min**, **normal**, or **detail**.

Flags

Item	Description
-h	Displays help for the **sodebug** command.
-l [*level*]	Specifies the trace level. Valid values for level are **none**, **min**, **normal**, and **detail**. If no level is specified, the default trace level is **normal**.
-p *pid*	Specifies the process ID of a process.
-s *sockaddr*	Specifies a socket by the socket address, the address of the socket's inpcb, or the address of the socket's tcpcb.
-t *type*	Specifies the type of address that is specified by the **-s** *sockaddr* option. Valid values are **socket**, **inpcb**, and **tcpcb**. The default value is **socket**.

Security

You must have root authority to run the **sodebug** command.

Examples

1. To list the debug flag and socket trace level for socket f100090002d0a800, type:

   ```
   sodebug -s f100090002d0a800
   ```

   The output is similar to the following example:

   ```
   socket address : f100090002d0a800 , sodebug flag : 0 , trace level : none(0)
   ```

2. To set the trace level to normal and set the debug flag to 1, type:

   ```
   sodebug -s f100090002d0a800 -l normal
   ```

   The output is similar to the following example:

   ```
   Setting new values for trace level and debug flag
   socket address : f100090002d0a800 , sodebug flag : 1 , trace level : normal(3)
   ```

soelim Command

Purpose

Processes **.so** requests in **nroff** command files.

Syntax

soelim [*File* ... | **-**]

Description

The **soelim** command reads specified files or standard input and performs inclusion specified by the **nroff** command and **troff** command requests of the form `.so filename` when the request appears at the beginning of input lines. Any combination of ASCII spaces and ASCII tab characters can follow the **.so** request and precede the file name. No characters should follow the file name.

The **soelim** command is useful because commands, such as the **tbl** command, do not normally perform file inclusions during processing.

When the **-** (minus sign) flag is specified, a file name corresponding to standard input is included.

Flag

Item	Description
-	Indicates a file name corresponding to standard input. **Note:** Inclusion can be suppressed by using a **'** (single quotation mark) instead of a **.** (period), as follows:

Parameter

Item	Description
File	Specifies files that the command performs inclusion on. The default is standard input.

```
'so /usr/share/lib/tmac/tmac.s
```

Example

Following is a sample usage of the **soelim** command:

```
soelim exum?.n | tbl | nroff -ms -Tlp | col -Tlp | pg
```

In this example, you use the **soelim** command to preprocess the file inclusion (**.so**) requests. The output is then passed to the **tbl** command. This makes it easier to place tables in separate files that can be included in forming a large document.

soestat Command

Purpose

Displays Serial over Ethernet (SoE) device driver information and statistics information of various SoE device drivers.

Syntax

soestat [**-a**] | [**-d** *sa_device_name* [**-p** *tty_device*]] [**-r**] [**-v**]

Description

The **soestat** command symbolically displays the driver information and statistics information of various SoE adapter (sa) serial devices and the teletype (tty) devices by using the SoE device driver.

Flags

-a
Displays statistics information for all sa serial devices and tty devices.

-d *sa_device_name*
Displays statistics information for a specific sa device and all tty devices that are configured for the specified sa device.

-p *tty_device*
Displays statistics information for a specific sa device and specific tty device.

-r
Resets all statistic information to their initial values.

-v
Verbose mode. Displays additional debug information.

Location

`/usr/sbin/soestat`

Files

/usr/sbin/soestat
Contains the **soestat** command.

/usr/lib/drivers/soedd
Contains the SoE kernel extension.

Security

You can run this command with all flags except the **-r** flag, which resets the statistic information. You must be either a root, or have aix.device.monitor.tty authority to use the **-r** flag.

Examples

1. To display information about all SoE devices, enter:

```
# soestat -a
```

2. To display information about the sa0 SoE Adapter only, enter:

```
# soestat -d sa0
```

3. To display information about tty1 tty (teletype) device, which is connected to the sa0 SoE adapter, enter:

```
# soestat -d sa0 -p tty1
```

4. To reset all statistic information, enter:

```
# soestat -ar
```

5. To reset information of the tty1 tty (teletype) device, which is connected to the sa0 SoE Adapter, enter:

```
# soestat -r -d sa0 -p tty1
```

sort Command

Purpose

Sorts files, merges files that are already sorted, and checks files to determine if they have been sorted.

Syntax

sort [**-A**] [**-b**] [**-c**] [**-d**] [**-f**] [**-i**] [**-m**] [**-n**] [**-r**] [**-u**] [**-o** *OutFile*] [**-t** *Character*] [**-T** *Directory*] [**-y** [*Kilobytes*]] [**-z** *RecordSize*] [[**+** [*FSkip*] [**.***CSkip*] [**b**] [**d**] [**f**] [**i**] [**n**] [**r**]] [**-** [*FSkip*] [**.***CSkip*] [**b**] [**d**] [**f**] [**i**] [**n**] [**r**]]] ... [**-k** *KeyDefinition*] ... [*File ...*]

Description

The **sort** command sorts lines in the files specified by the *File* parameter and writes the result to standard output. If the *File* parameter specifies more than one file, the **sort** command concatenates the files and sorts them as one file. A **-**(minus sign) in place of a file name specifies standard input. If you do not specify any file names, the command sorts standard input. An output file can be specified with the **-o** flag.

If no flags are specified, the **sort** command sorts entire lines of the input file based upon the collation order of the current locale.

Sort Keys

A sort key is a portion of an input line that is specified by a field number and a column number. Fields are parts of input lines that are separated by field separators. The default field separator is a sequence of one or more consecutive blank characters. However, these blank characters are considered to be a part of the following field for sorting purposes. You can specify the **-b** option to ignore these leading blank characters. A different field separator can be specified using the **-t** flag. The tab and the space characters are the blank characters in the C and English Language locales.

When using sort keys, the **sort** command first sorts all lines on the contents of the first sort key. Next, all the lines whose first sort keys are equal are sorted upon the contents of the second sort key, and so on. Sort keys are numbered according to the order they appear on the command line. If two lines sort equally on all sort keys, the entire lines are then compared based upon the collation order in the current locale.

When numbering columns within fields, the blank characters in a default field separator are counted as part of the following field. Field separator characters specified by the **-t** flag are not counted as parts of fields. Leading blank characters can be ignored using the **-b** flag.

Sort keys can be defined using the following two methods:

- **-k** *KeyDefinition*
- *FSkip***.***CSkip* (obsolescent version).

Sort Key Definition Using the -k Flag

The **-k** *KeyDefinition* flag uses the following form:

-k [*FStart* [**.***CStart*]] [*Modifier*] [**,** [*FEnd* [**.***CEnd*]][*Modifier*]]

The sort key includes all characters beginning with the field specified by the *FStart* variable and the column specified by the *CStart* variable and ending with the field specified by the *FEnd* variable and the column specified by the *CEnd* variable. If *Fend* is not specified, the last character of the line is assumed. If *CEnd* is not specified the last character in the *FEnd* field is assumed. Any field or column number in the *KeyDefinition* variable may be omitted. The default values are:

Item	Description
FStart	Beginning of the line

Item	Description
CStart	First column in the field
FEnd	End of the line
CEnd	Last column of the field

If there is any space between the fields, **sort** considers them as separate fields.

The value of the *Modifier* variable can be one or more of the letters **b**, **d**, **f**, **i**, **n**, or **r**. The modifiers apply only to the field definition they are attached to and have the same effect as the flag of the same letter. The modifier letter **b** applies only to the end of the field definition to which it is attached. For example:

```
-k 3.2b,3r
```

specifies a sort key beginning in the second nonblank column of the third field and extending to the end of the third field, with the sort on this key to be done in reverse collation order. If the *FStart* variable and the *CStart* variable fall beyond the end of the line or after the *FEnd* variable and the *CEnd* variable, then the sort key is ignored.

A sort key can also be specified in the following manner:

[**+**[*FSkip1*] [*.CSkip1*] [*Modifier*]] [**-**[*FSkip2*] [*.CSkip2*] [*Modifier*]]

The **+***FSkip1* variable specifies the number of fields skipped to reach the first field of the sort key and the **+***CSkip* variable specifies the number of columns skipped within that field to reach the first character in the sort key. The **-***FSkip* variable specifies the number of fields skipped to reach the first character *after* the sort key, and the **-***CSkip* variable specifies the number of columns to skip within that field. Any of the field and column skip counts may be omitted. The defaults are:

Item	Description
FSkip1	Beginning of the line
CSkip1	Zero
FSkip2	End of the line
CSkip2	Zero

The modifiers specified by the *Modifier* variable are the same as in the **-k** flag key sort definition.

The field and column numbers specified by **+***FSkip1***.***CSkip1* variables are generally one less than the field and column number of the sort key itself because these variables specify how many fields and columns to skip before reaching the sort key. For example:

```
+2.1b -3r
```

specifies a sort key beginning in the second nonblank column of the third field and extending to the end of the third field, with the sort on this key to be done in reverse collation order. The statement +2 . 1b specifies that two fields are skipped and then the leading blanks and one more column are skipped. If the **+***FSkip1***.***CSkip1* variables fall beyond the end of the line or after the **-***FSkip2***.***CSkip2* variables, then the sort key is ignored.

Note: The maximum number of fields on a line is 32.

Flags

Note: A **-b**, **-d**, **-f**, **-i**, **-n**, or **-r** flag that appears before any sort key definition applies to all sort keys. None of the **-b**, **-d**, **-f**, **-i**, **-n**, or **-r** flags may appear alone after a **-k** *KeyDefinition*; if they are attached to a *KeyDefinition* variable as a modifier, they apply only to the attached sort key. If one of these flags follows a **+***Fskip.Cskip* or **-***Fskip.Cskip* sort key definition, the flag only applies to that sort key.

Item	Description
-A	Sorts on a byte-by-byte basis using ASCII collation order instead of collation in the current locale.
-b	Ignores leading spaces and tabs to find the first or last column of a field.
-c	Checks that input is sorted according to the ordering rules specified in the flags. A nonzero value is returned if the input file is not correctly sorted.
-C	Checks that input is sorted according to the ordering rules specified in the flags except that a warning message shall not be sent to standard error if there is a disorder or, with **-u** option, a duplicate key is detected.
-d	Sorts using dictionary order. Only letters, digits, and spaces are considered in comparisons.
-f	Changes all lowercase letters to uppercase before comparison.
-i	Ignores all nonprinting characters during comparisons.
-k *KeyDefinition*	Specifies a sort key. The format of the *KeyDefinition* option is: [*FStart* [**.***CStart*]] [*Modifier*] [**,** [*FEnd* [**.***CEnd*]][*Modifier*]] The sort key includes all characters beginning with the field specified by the *FStart* variable and the column specified by the *CStart* variable and ending with the field specified by the *FEnd* variable and the column specified by the *CEnd* variable. The value of the *Modifier* variable can be **b**, **d**, **f**, **i**, **n**, or **r**. The modifiers are equivalent to the flags of the same letter. When a modifier is attached to a key definition, then no flag is applied to it.
-m	Merges multiple input files only; the input are assumed to be already sorted.
-n	Sorts numeric fields by arithmetic value. A numeric field may contain leading blanks, an optional minus sign, decimal digits, thousands-separator characters, and an optional radix character. Numeric sorting of a field containing any nonnumeric character gives unpredictable results.
-o *OutFile*	Directs output to the file specified by the *OutFile* parameter instead of standard output. The value of the *OutFile* parameter can be the same as the *File* parameter.
-r	Reverses the order of the specified sort.
-t *Character*	Specifies *Character* as the single field separator character.
-u	Suppresses all but one line in each set of lines that sort equally according to the sort keys and options.
-T *Directory*	Places all temporary files that are created into the directory specified by the *Directory* parameter.
-y[*Kilobytes*]	Starts the **sort** command using the number of kilobytes of main storage specified by the *Kilobytes* parameter and adds storage as needed. (If the value specified in the *Kilobytes* parameter is less than the minimum storage site or greater than the maximum, the minimum or maximum is used instead). If the **-y** flag is omitted, the **sort** command starts with the default storage size. The **-y0** flag starts with minimum storage, and the **-y** flag (with no *Kilobytes* value) starts with maximum storage. The amount of storage used by the **sort** command affects performance significantly. Sorting a small file in a large amount of storage is wasteful.

Item	Description
-z *RecordSize*	Prevents abnormal termination if any of the lines being sorted are longer than the default buffer size. The default buffer size is 20 KB. When the **-c** or **-m** flags are specified, the sorting phase is omitted and a system default buffer size is used. If sorted lines are longer than this size, the **sort** command terminates abnormally. The **-z** option specifies recording of the longest line in the sort phase so adequate buffers can be allocated in the merge phase. The *RecordSize* variable must designate a value in bytes equal to or greater than the longest line to be merged. The longest line size that is supported under C locale is approximately 2M characters and the longest line size that is supported under non-C locale is 1M characters. The **-z** option is ineffective under C locale.

Exit Status

This command returns the following exit values:

Item	Description
0	All input files were output successfully, or **-c** was specified and the input file was correctly sorted.
1	Under the **-c** option, the file was not ordered as specified, or if the **-c** and **-u** options were both specified, two input lines were found with equal keys.
>1	An error occurred.

Examples

1. To sort the `fruits` file with the **LC_ALL**, **LC_COLLATE**, or **LANG** environment variable set to En_US, enter:

   ```
   LANG=En_US sort fruits
   ```

   This command sequence displays the contents of the `fruits` file sorted in ascending lexicographic order. The characters in each column are compared one by one, including spaces, digits, and special characters. For instance, if the `fruits` file contains the text:

   ```
   banana
   orange
   Persimmon
   apple
   %%banana
   apple
   ORANGE
   ```

   the **sort** command displays:

   ```
   %%banana
   ORANGE
   Persimmon
   apple
   apple
   banana
   orange
   ```

   In the ASCII collating sequence, the % (percent sign) precedes uppercase letters, which precede lowercase letters. If your current locale specifies a character set other than ASCII, your results may be different.

2. To sort in dictionary order, enter:

   ```
   sort  -d fruits
   ```

This command sequence sorts and displays the contents of the `fruits` file, comparing only letters, digits, and spaces. If the `fruits` file is the same as in example 1, then the **sort** command displays:

```
ORANGE
Persimmon
apple
apple
%%banana
banana
orange
```

The **-d** flag ignores the % (percent sign) character because it is not a letter, digit, or space, placing %%banana with banana.

3. To group lines that contain uppercase and special characters with similar lowercase lines, enter:

   ```
   sort  -d  -f fruits
   ```

   The **-d** flag ignores special characters and the **-f** flag ignores differences in case. With the **LC_ALL**, **LC_COLLATE**, or **LANG** environment variable set to C, the output for the `fruits` file becomes:

   ```
   apple
   apple
   %%banana
   banana
   ORANGE
   orange
   Persimmon
   ```

4. To sort, removing duplicate lines, enter:

   ```
   sort  -d  -f  -u fruits
   ```

   The **-u** flag tells the **sort** command to remove duplicate lines, making each line of the file unique. This command sequence displays:

   ```
   apple
   %%banana
   ORANGE
   Persimmon
   ```

   Not only is the duplicate `apple` removed, but banana and ORANGE as well. These are removed because the **-d** flag ignores the %% special characters and the **-f** flag ignores differences in case.

5. To sort as in example 4, removing duplicate instances unless capitalized or punctuated differently, enter:

   ```
   sort  -u +0  -d  -f +0 fruits
   ```

   Entering the `+0 -d -f` does the same type of sort that is done with `-d -f` in example 3. Then the `+0` performs another comparison to distinguish lines that are not identical. This prevents the **-u** flag from removing them.

   Given the `fruits` file shown in example 1, the added `+0` distinguishes %%banana from banana and ORANGE from `orange`. However, the two instances of `apple` are identical, so one of them is deleted.

   ```
   apple
   %%banana
   banana
   ORANGE
   orange
   Persimmon
   ```

6. To specify the character that separates fields, enter:

   ```
   sort  -t: +1 vegetables
   ```

   This command sequence sorts the `vegetables` file, comparing the text that follows the first colon on each line. The +1 tells the **sort** command to ignore the first field and to compare from the start of the

second field to the end of the line. The `-t:` flag tells the **sort** command that colons separate fields. If `vegetables` contains:

```
yams:104
turnips:8
potatoes:15
carrots:104
green beans:32
radishes:5
lettuce:15
```

Then, with the **LC_ALL**, **LC_COLLATE**, or **LANG** environment variable set to C, the **sort** command displays:

```
carrots:104
yams:104
lettuce:15
potatoes:15
green beans:32
radishes:5
turnips:8
```

Note that the numbers are not in numeric order. This happened when a lexicographic sort compares each character from left to right. In other words, 3 comes before 5, so 32 comes before 5.

7. To sort numbers, enter:

   ```
   sort  -t: +1  -n vegetables
   ```

   This command sequence sorts the `vegetables` file numerically on the second field. If the `vegetables` file is the same as in example 6, then the **sort** command displays:

   ```
   radishes:5
   turnips:8
   lettuce:15
   potatoes:15
   green beans:32
   carrots:104
   yams:104
   ```

8. To sort more than one field, enter:

   ```
   sort  -t: +1 -2  -n +0 -1  -r vegetables
   ```

   OR

   ```
   sort  -t:  -k2,2 n  -k1,1 r vegetables
   ```

   This command sequence performs a numeric sort on the second field (`+1 -2 -n`). Within that ordering, it sorts the first field in reverse alphabetic order (`+0 -1 -r`). With the **LC_ALL**, **LC_COLLATE**, or **LANG** environment variable set to C, the output looks like this:

   ```
   radishes:5
   turnips:8
   potatoes:15
   lettuce:15
   green beans:32
   yams:104
   carrots:104
   ```

   The command sorts the lines in numeric order. When two lines have the same number, they appear in reverse alphabetic order.

9. To replace the original file with the sorted text, enter:

   ```
   sort  -o vegetables vegetables
   ```

   This command sequence stores the sorted output into the `vegetables` file (`-o vegetables`).

Files

Item	Description
/usr/bin/sort	Contains the **sort** command.

Item	Description
/var/tmp	Temporary space during the **sort** command processing.
/usr/tmp	Temporary space during the **sort** command processing, if file cannot be created in **/var/tmp**.
/tmp	Temporary space during the **sort** command processing, if file cannot be created in */var/tmp* or **/usr/tmp**.

sortbib Command

Purpose

Sorts a bibliographic database.

Syntax

sortbib [**-s***Keys*] [*Database* ...]

Description

The **sortbib** command sorts files of records containing **refer** command key letters by user-specified keys. The records can be separated by blank lines, or enclosed by the .[(period, left bracket) and the .] (period, right bracket) delimiters, but the two styles cannot be mixed together. The **sortbib** command reads through each database specified by the *Database* parameter and pulls out key fields, which are sorted separately. The sorted key fields contain the file pointer, byte offset, and length of corresponding records. These records are delivered using disk seeks and reads, so the **sortbib** command cannot be used in a pipeline to read standard input.

By default, the **sortbib** command alphabetizes by the first %A and %D fields, which contain the senior author and date.

The **sortbib** command sorts by the last word in the %A field, which is assumed to be the author's last name. A word in the final position, such as `jr.` or ed., is ignored if the name preceding ends with a comma. Authors with two-word last names, or names with uncommon constructions, can be sorted correctly by using the **nroff** command convention **\0** in place of a space character. Specifying the %Q field is similar to the %A field, except sorting begins with the first, not the last, word.

Note: Records with missing author fields should be sorted by title.

The **sortbib** command sorts by the last word of the %D line, which is usually the year. It ignores leading articles when sorting by titles in the %T or %J fields. The articles ignored are specific to the locale and specified in the locale-specific **refer message catalog**. Within this catalog, the articles are contained in a single message. Each article is separated by any number of ASCII space or tab characters. If a sort-significant field is absent from a record, the **sortbib** command places the record before other records containing that field.

No more than 16 databases can be sorted together at one time. Records longer than 4096 characters are truncated.

The *Database* parameter contains **refer** command key letters by user-specified keys that the **sortbib** command sorts through.

Flags

Item	Description
-s*Keys*	Specifies field keys to sort on.

Examples

1. To sorts by author, title, and date:

```
sortbib -sATD Database
```

2. To sort by author and date:

```
sortbib -sA+D Database
```

Files

Item	Description
/tmp/Sbib*XXXXX*	Contains the temporary file.
/usr/bin/sort	Contains the **sort** command.

sortm Command

Purpose

Sorts messages.

Syntax

sortm [**+***Folder*] [*Messages*] [**-datefield** *Field*] [**-noverbose** | **-verbose**]

Description

The **sortm** command sorts messages according to their `Date:` field and renumbers them consecutively beginning with number one. Messages that are in the folder, but not specified to be sorted, are placed after the sorted messages. The **sortm** command displays a message if it cannot parse a date field.

To specify a field other than the `Date:` field, specify the **-datefield** flag. If you specify a folder, it becomes the current folder. The current message remains the current message for the specified folder, even if it moves during the sort.

Flags

Item	Description
-datefield *Field*	Specifies the header field to be used in the sort. The `Date:` field is the default.
+*Folder*	Specifies the folder with messages to be sorted. The default is the current folder.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.

Item	Description
Messages	Specifies the messages to be sorted. Use the following references to specify messages: ***Number*** Number of the message. ***Sequence*** A group of messages specified by the user. Recognized values are: **all** All messages in a folder. This is the default. **cur or . (period)** Current message. **first** First message in a folder. **last** Last message in a folder. **next** Message following the current message. **prev** Message preceding the current message.
-noverbose	Prevents display of information during the sort. This flag is the default.
-verbose	Displays information during the sort. This information allows you to monitor the steps involved.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Profile Entries

The following entries are found in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Current-Folder:`	Sets the default current folder.
`Path:`	Specifies the *UserMhDirectory*.

Examples

1. To sort all the messages in the current folder according to the date, enter:

```
sortm
```

2. To sort messages 5 through 10 in the `easter` folder according to the date, enter:

```
sortm  +easter 5-10
```

Files

Item	Description
$HOME/.mh_profile	Contains the MH user profile.
/usr/bin/sortm	Contains the **sortm** command.

spell Command

Purpose

Finds English Language spelling errors.

Syntax

spell [**-b**] [**-i**] [**-l**] [**-v**] [**-x**] [**-d** *HashList*] [**-h** *HistoryList*] [**-s** *HashStop*] [**+** *WordList*] [*File* ...]

Description

The **spell** command reads words in the file indicated by the *File* variable and compares them to those in a spelling list. Words that cannot be matched in the spelling list or derived from words in the spelling list (by applying certain inflections, prefixes, and suffixes) are written to standard output. If no file name is specified, the **spell** command reads standard input.

The **spell** command ignores the same **troff**, **tbl**, and **eqn** codes as does the **deroff** command.

The coverage of the spelling list is uneven. You should create your own dictionary of special words used in your files. Your dictionary is a file containing a sorted list of words, one per line. To create your dictionary, use the **spellin** command.

Files containing an alternate spelling list, history list, and stop list can be specified by file name parameters following the **-d**, **-f**, and **-h** flags. Copies of all output can be accumulated in the history file.

Three programs help maintain and check the hash lists used by the **spell** command:

Item	Description
/usr/lbin/spell/hashmake	Reads a list of words from standard input and writes the corresponding 9-digit hash code to standard output.
/usr/bin/spellin *Number*	Reads the specified *Number* of hash codes from standard input and writes a compressed spelling list to standard output.
/usr/lbin/spell/hashcheck *SpellingList*	Reads a compressed *SpellingList*, recreates the 9-digit hash codes for all the words in it, and writes these codes to standard output.

The *File* parameter specifies the files that the **spell** command reads and compares them with the spelling list. If no file is specified, the command reads standard input.

Flags

Item	Description
-b	Checks British spelling. However, this flag does not provide a reasonable prototype for British spelling. The algorithms to derive a match against the spelling dictionary by applying certain inflections, prefixes, and suffixes are based on American English spelling.
-d *HashList*	Specifies the *HashList* file as the alternative spelling list. The default is **/usr/share/dict/hlist[ab]**.
-h *HistoryList*	Specifies the *HistoryList* file as the alternative history list, which is used to accumulate all output. The default is **/usr/lbin/spell/spellhist**. **Note:** The *HistoryList* file must be an existing file with read and write permissions.
-i	Suppresses processing of include files.
-l	Follows the chain of all include files (**.so** and **.nx** formatting commands). Without this flag, the **spell** command follows chains of all include files except for those beginning with **/usr/lib**.
-s *HashStop*	Specifies the *HashStop* file as the alternative stop list, which is used to filter out misspellings that would otherwise pass. The default is **/usr/share/dict/hstop**.
-v	Displays all words not in the spelling list and indicates possible derivations from the words.
-x	Displays every possible word stem with an = (equal sign).
+ *WordList*	Checks *WordList* for additional word spellings. *WordList* is the name of a file you provide that contains a sorted list of words, one per line. With this flag, you can specify a set of correctly spelled words (in addition to the **spell** command's own spelling list) for each job.

Exit Status

The following exit values are returned:

Item	Description
0	Indicates successful completion.
>0	Indicates an error occurred.

Examples

1. To check your spelling, enter:

   ```
   spell chap1 >mistakes
   ```

   This creates a file named `mistakes` containing all the words found in `chap1` that are not in the system spelling dictionary. Some of these may be correctly spelled words that the **spell** command does not recognize. Save the output of the **spell** command in a file because the word list may be long.

2. To check British spelling, enter:

   ```
   spell  -b chap1 >mistakes
   ```

   This checks `chap1` against the British dictionary and writes the questionable words in the `mistakes` file.

3. To see how the **spell** command derives words, enter:

```
spell  -v chap1 >deriv
```

This lists words not found literally in the dictionary but are derived from forms of dictionary words. The prefixes and suffixes used to form the derivations are indicated for each word. Words that are not in the dictionary at all are also listed.

4. To check your spelling against an additional word list, enter:

```
spell  +newwords chap1
```

This checks the spelling of words in `chap1` against the system dictionary and against `newwords`. The `newwords` file lists words in alphabetical order, one per line. You can create this file with a text editor, such as the ed editor, and alphabetize it with the **sort** command.

Files

Item	Description
/usr/share/dict/hlist[ab]	Contains hashed spelling lists, both American and British.
/usr/share/dict/hstop	Contains a hashed stop list.
/usr/lbin/spell/spellhist	Contains a history file.
/usr/lbin/spell/compress	Contains an executable shell program to compress the history file.
/usr/lbin/spell/hashmake	Creates hash codes from a spelling list.
/usr/bin/spellin *Number*	Creates spelling list from hash codes.
/usr/lbin/spell/hashcheck *SpellingList*	Creates hash codes from a compressed spelling list.
/usr/lbin/spell/spellinprg	Main program called by the **spellin** file.
/usr/lbin/spell/spellprog	Checks spelling.

spellin Command

Purpose

Creates a spelling list.

Syntax

spellin [*List* | *Number*]

Description

The **spellin** command creates a spelling list for use by the **spell** command. The parameter for the **spellin** command can be a file name or a number. The **spellin** command combines the words from the standard input and the already existing spelling list file and places a new spelling list on the standard output. If no list file is specified, a new list is created. If *Number* is specified, the **spellin** command reads the specified number of hash codes from standard input and writes a compressed spelling list.

Examples

To add the word `hookey` to the spelling list named `myhlist`, enter:

```
echo hookey | spellin /usr/share/dict/hlista > myhlist
```

spellout Command

Purpose

Verifies that a word is not in the spelling list.

Syntax

spellout [**-d**] *List*

Description

The **spellout** command looks up each word from standard input and prints on standard output those that are missing from the hashed list file specified by the *List* parameter. The hashed list file is similar to the dictionary file used by the **spell** command.

Flags

Item	Description
-d	Prints those words that are present in the hashed list file.

Examples

To verify that the word hookey is not on the default spelling list, enter:

```
echo hookey | spellout /usr/share/dict/hlista
```

In this example, the **spellout** command prints the word hookey on standard output if it is not in the hashed list file. With the **-d** flag, **spellout** prints the word hookey if it is found in the hash file.

splat Command

Purpose

Simple Performance Lock Analysis Tool (splat). Provides kernel and **pthread** lock usage reports.

Syntax

splat -i *file* [**-n** *file*] [**-o** *file*] [**-d** [bfta]] [**-l** *address*] [**-c** *class*] [**-s** [acelmsS]] [**-C** *cpus*] [**-S** *count*] [**-t** *start*] [**-T** *stop*] [**-p**]

splat -h [*topic*]

splat -j

Description

splat (Simple Performance Lock Analysis Tool) is a software tool which post-processes AIX trace files to produce kernel simple and complex lock usage reports. It also produces **pthread** mutex read-write locks, and condition variables usage reports.

Flags

Item	Description
-i *inputfile*	AIX trace file (REQUIRED).
-n *namefile*	File containing output of **gensyms** command.
-o *outputfile*	File to write reports to (DEFAULT: **stdout**).
-d *detail*	Detail can be one of:[**b**]asic: summary and lock detail (DEFAULT) [**f**]unction: basic + function detail [**t**]hread: basic + thread detail [**a**]ll: basic + function + thread detail
-c *class*	If the user supplies a decimal lock class index, **splat** will only report activity for locks in that class.
-l *address*	If the user supplies a hexadecimal lock address, **splat** will only report activity for the lock at that address. **splat** will filter a trace file for lock hooks containing that lock address and produce a report solely for that lock.
-s *criteria*	Sort the lock, function, and thread reports by the following criteria: **a** acquisitions **c** percent processor hold time **e** percent elapsed hold time **l** lock address, function address, or thread ID **m** miss rate **s** spin count **S** percent processor spin hold time (DEFAULT) **w** percent real wait time **W** average waitq depth
-C *cpus*	Specify the number of processors present for this trace.
-S *count*	The maximum number of entries in each report (DEFAULT: 10).
-t *starttime*	Time offset in seconds from the beginning of the trace.
-T *stoptime*	Time offset in seconds from the beginning of the trace to stop analyzing trace data. (DEFAULT: the end of the trace.)
-h [*topic*]	Help on usage or a specific topic. Valid topics are: • all • overview • input • names • reports • sorting
-j	Print a list of trace hooks used by splat.

Item	Description
-p	Specifies the use of the PURR register to calculate processor times.

Help

The following is a list of available help topics and a brief summary of each:

Item	Description
OVERVIEW	This text.
INPUT	AIX trace hooks required in order to acquire useful output from **splat**.
NAMES	What name utilities can be used to cause **splat** to map addresses to human-readable symbols.
REPORTS	A description of each report that splat can produce and the formulas used to calculate reported values.
SORTING	A list of all the available sorting options and how they are applied to **splat**'s output.

Splat Trace

Splat takes as primary input an AIX trace file which has been collected with the AIX trace command. Before analyzing a trace with **splat**, you will need to make sure that the trace is collected with an adequate set of hooks, including the following:

```
106 DISPATCH
10C DISPATCH IDLE PROCESS
10E RELOCK
112 LOCK
113 UNLOCK
134 HKWD_SYSC_EXECVE
139 HKWD_SYSC_FORK
419 CPU PREEMPT
465 HKWD_SYSC_CRTHREAD
46D WAIT LOCK
46E WAKEUP LOCK
606 HKWD_PTHREAD_COND
607 HKWD_PTHREAD_MUTEX
608 HKWD_PTHREAD_RWLOCK
609 HKWD_PTHREAD_GENERAL
```

Capturing these lock and unlock trace events can cause serious performance degradation due to the frequency that locks are used in a multiprocessor environment. Therefore, lock trace event reporting is normally disabled. In order to enable lock trace event reporting, the following steps must be taken before a trace can be collected which will include lock trace events that splat requires (KornShell syntax):

```
1. bosboot -ad /dev/hdisk0 -L

2. shutdown -Fr

3. (reboot the machine)

4. locktrace -S

5. mkdir temp.lib; cd temp.lib

6. ln -s /usr/ccs/lib/perf/libpthreads.a

7. export LIBPATH=$PWD:$LIBPATH
```

Steps 1 through 3 are optional. They enable the display of kernel lock class names instead of addresses. Please refer to **bosboot(1)** for more information on **bosboot** and its flags. Steps 5 through 7 are necessary for activating the user **pthread** lock instrumentation; the **temp.lib** subdirectory can be put anywhere. Steps 1 through 7 are necessary in order for the report to be complete.

Splat Names

Splat can take the output of **gensyms** as an optional input and use it to map lock and function addresses to human-readable symbols.

Lock classes and **offsets** can be used to identify a lock broadly, but not as specifically as the actual symbol.

Splat Reports

The report generated by **splat** consists of a report summary, a lock summary report section, and a list of lock detail reports, each of which may have an associated function detail and/or thread detail report.

```
Report Summary
^^^^^^^^^^^^^^
 The report summary consists of the following elements:

 - The trace command used to collect the trace.
 - The host that the trace was taken on.
 - The date that the trace was taken on.
 - The duration of the trace in seconds.
 - The estimated number of CPUs
 - The combined elapsed duration of the trace in seconds;
   ( the duration of the trace multiplied by the number of
     CPUs identified during the trace ).
 - Start time, which is the offset in seconds from the beginning of the
   trace that trace statistics begin to be gathered.
 - Stop time, which is the offset in seconds from the beginning of the
   trace that trace statistics stop being gathered.
 - Total number of acquisitions during the trace.
 - Acquisitions per second, which is computed by dividing
   the total number of lock acquisitions by the real-time
   duration of the trace.
 - % of Total Spin Time, this is the summation of all lock spin hold
   times, divided by the combined trace duration in seconds, divided by 100.
   The current goal is to have this value be less than 10% of the total
   trace duration.


 Lock Summary
 ^^^^^^^^^^^^

 The lock summary report has the following fields:

Lock                The name, lockclass or address of the lock.


Type              The type of the lock, identified by one of the following letters:
                      Q    A RunQ lock
                      S    A simple kernel lock
                      D    A disabled simple kernel lock
                      C    A complex kernel lock
                      M    A PThread mutex
                      V    A PThread condition-variable
                      L    A PThread read/write lock
Acquisitions     The number of successful lock attempts for this lock, minus
                  the number of times a thread was preempted while holding
                  this lock.

Spins            The number of unsuccessful lock attempts for this lock,
                  minus the number of times a thread was undispatched while
                  spinning.

Wait             The number of unsuccessful lock attempts that
 or               resulted in the attempting thread going to
Transform         sleep to wait for the lock to become available,
                  or allocating a krlock.

%Miss            Spins divided by Acquisitions plus Spins, multiplied by 100.

%Total           Acquisitions divided by the total number of all
                  lock acquisitions, multiplied by 100.

Locks/CSec       Acquisitions divided by the combined elapsed
                    duration in seconds.

Percent HoldTime
Real CPU          The percent of combined elapsed trace time that
```

```
                  threads held the lock in question while dispatched.
                  DISPATCHED_HOLDTIME_IN_SECONDS divided by combined
                  trace duration, multiplied by 100.

Real Elaps(ed)    The percent of combined elapsed trace time that
                  threads held the lock while dispatched or sleeping.
                  UNDISPATCHED_AND_DISPATCHED_HOLDTIME_IN_SECONDS divided
                  by combined trace duration, multiplied by 100.

Comb Spin         The percent of combined elapsed trace time that
                  threads spun while waiting to acquire this lock.
                  SPIN_HOLDTIME_IN_SECONDS divided by combined trace
                  duration, multiplied by 100.
```

The lock summary report defaults to a list of ten locks, sorted in descending order by percent spin hold time (the tenth field). The length of the summary report can be adjusted using the **-S** switch. The sorted order of the summary report (and all other reports) can be set with the **-s** switch whose options are described in the SORTING help section, **splat -h** sorting.

```
Lock Detail
^^^^^^^^^^^

 The lock detail report consists of the following fields:

 LOCK             The address (in hexadecimal) of the lock.

 NAME             The symbol mapping for that address (if available)

 CLASS            The lockclass name (if available) and hexadecimal offset,
                  used to allocate this lock ( lock_alloc() kernel service ).

Parent Thread    Thread id of the parent thread. This field only exists for Mutex,
                  Read/Write lock and Conditional Variable report.

 creation time    Elapsed time in seconds after the first event recorded in trace,
                  if available. This field only exists for Mutex, Read/Write lock
                  and Conditional Variable report.
 deletion time    Elapsed time in seconds after the first event recorded in trace,
                  if available. Tthis field only exists for Mutex, Read/Write lock
                  and Conditional Variable report.

 Pid              Pid number associated to the lock (this field only exists for Mutex,
                  Read/Write lock and Conditional Variable report).

 Process Name    Process name associated to the lock (this field only exists for Mutex,
                 Read/Write lock and Conditional Variable report).

 Call-Chain      Stack of called methods (if possible to have them, this field only
                 exists for Mutex, Read/Write lock and Conditional Variable report).


 Acquisitions     The number of successful lock attempts for this lock.
                  This field is named Passes for the conditional variable lock report.

 Miss Rate        The number of unsuccessful lock attempts divided by
                  Acquisitions plus unsuccessful lock attempts, multiplied
                  by 100.

 Spin Count       The number of unsuccessful lock attempts.

 Wait Count       The number of unsuccessful lock attempts that resulted in
                  the attempting thread going to sleep to wait for the lock
                  to become available.

 Transform Count  The number of krlock allocated and deallocated by the simple lock.

 Busy Count          The number of simple_lock_try() calls that returned busy.

 Seconds Held
 CPU              The total time in seconds that this lock was held by
                  dispatched threads.

 Elapsed             The total time in seconds that this lock was held by
                  both dispatched and undispatched threads.

NOTE: neither of these two values should exceed the
      total real elapsed duration of the trace.
```

```
 Percent Held
 Real CPU          The percent of combined elapsed trace time that
                   threads held the lock in question while dispatched.
                   DISPATCHED_HOLDTIME_IN_SECONDS divided by trace
                   duration, multiplied by 100.

 Real Elaps(ed)    The percent of combined elapsed trace time that
                   threads held the lock while dispatched or sleeping.
                   UNDISPATCHED_AND_DISPATCHED_HOLDTIME_IN_SECONDS divided
                   by trace duration, multiplied by 100.

 Comb Spin         The percent of combined elapsed trace time that
                   threads spun while waiting to acquire this lock.
                   SPIN_HOLDTIME_IN_SECONDS divided by trace duration,
                   multiplied by 100.

 Wait              The percentage of combined elapsed trace time that
                   threads unsuccessfully tried to acquire this lock.

 SpinQ             Splat keeps track of the minimum, maximum and average
                   depth of the spin queue (the threads spinning, waiting
                   for a lock to become available).

 WaitQ             As with the spin queue, splat also tracks the minimum,
                   maximum and average depth of the queue of threads waited
                   waiting for a lock to become available).

PROD               The associated krlocks prod calls count.

CONFER SELF        The confer to self calls count for the simple lock and the
                   associated krlocks.

CONFER TARGET      The confer to target calls count for the simple lock and the
                   associated krlocks. w/ preemption reports the successfull calls
                   count, resulting in a preemption.

CONFER ALL         The confer to all calls count for the simple lock and the
                   associated krlocks. w/ preemption reports the successfull calls
                   count, resulting in a preemption.

HANDOFF            The associated krlocks handoff calls count.

Lock Activity w/Interrupts Enabled (mSecs)
```

This section of the lock detail report are dumps of the raw data that splat collects for each lock, times expressed in milliseconds. The five states: LOCK, SPIN, WAIT, UNDISP(atched) and PREEMPT are the five basic states of **splat**'s enabled **simple_lock** finite state machine. The count for each state is the number of times a thread's actions resulted in a transition into that state. The durations in milliseconds show the minimum, maximum, average and total amounts of time that a lock request spent in that state.

```
   LOCK:    this state represents a thread successfully acquiring a lock.

   SPIN:    this state represents a thread unsuccessfully trying to acquire
            a lock.

   WAIT:    this state represents a spinning thread (in SPIN) going to sleep
            (voluntarily) after exceeding the thread's spin threshold.

   UNDISP:  this state represents a spinning thread (in SPIN) becoming
            undispatched (involuntarily) before exceeding the thread's
            spin threshold.

   PREEMPT: this state represents when a thread holding a lock is
            undispatched.


Lock Activity w/Interrupts Disabled (mSecs)
```

This section of the lock detail report are dumps of the raw data that splat collects for each lock, times expressed in milliseconds. The six states: LOCK, SPIN, LOCK with KRLOCK, KRLOCK LOCK, KRLOCK SPIN and TRANSFORM are the six basic states of **splat**'s disabled **simple_lock** finite state machine. The count for each state is the number of times a thread's actions resulted in a transition into that state. The

durations in milliseconds show the minimum, maximum, average and total amounts of time that a lock request spent in that state.

```
LOCK:        This state represents a thread successfully acquiring a lock.

SPIN:        This state represents a thread unsuccessfully trying to acquire
             a lock.

LOCK with    The thread has successfully acquired the lock, while holding
 KRLOCK:     the associated krlock, and is currently executing.

KRLOCK LOCK: The thread has successfully acquired the associated krlock,
             and is currently executing.

KRLOCK SPIN: The thread is executing and unsuccessfully attempting to acquire
             the associated krlock.

TRANSFORM:   The thread has successfully allocated a krlock it associates to,
             and is executing.


Function Detail
^^^^^^^^^^^^^^^

 The function detail report consists of the following fields:

 Function Name   The name or return address of the function which
                 used the lock.

 Acquisitions       The number of successful lock attempts for this lock.
                 For complex lock and read/write lock there is a
                 distinction between acquisition for writing
                 (Acquisition Write) and for reading
                 (Acquisition Read).


 Miss Rate      The number of unsuccessful lock attempts divided by
                Acquisitions, multiplied by 100.

 Spin Count     The number of unsuccessful lock attempts.
                For complex lock and read/write lock there is a
                distinction between spin count for writing
                (Spin Count Write) and for reading
                (Spin Count Read).

 Wait Count     The number of unsuccessful lock attempts that resulted in
                the attempting thread going to sleep to wait for the lock
                to become available.
                For complex lock and read/write lock there is a
                distinction between wait count for writing
                (Wait Count Write) and for reading
                (Wait Count Read).

Transform Count The number of times that a simple lock has allocated a krlock,
                while the thread was trying to acquire the simple lock.

 Busy Count         The number of simple_lock_try() calls that returned busy.

 Percent Held of Total Time
 CPU              The percent of combined elapsed trace time that
                  threads held the lock in question while dispatched.
                  DISPATCHED_HOLDTIME_IN_SECONDS divided by trace
                  duration, multiplied by 100.

 Elaps(ed)          The percent of combined elapsed trace time that
                  threads held the lock while dispatched or sleeping.
                  UNDISPATCHED_AND_DISPATCHED_HOLDTIME_IN_SECONDS divided
                  by trace duration, multiplied by 100.

 Spin             The percent of combined elapsed trace time that
                  threads spun while waiting to acquire this lock.
                  SPIN_HOLDTIME_IN_SECONDS divided by combined trace
                  duration, multiplied by 100.

 Wait             The percentage of combined elapsed trace time that
                  threads unsuccessfully tried to acquire this lock.

 Return Address   The calling function's return address in hexadecimal.

 Start Address       The start address of the calling function in hexadecimal.
```

```
 Offset           The offset from the function start address in hexadecimal.


Thread Detail
^^^^^^^^^^^^^

The thread detail report consists of the following fields:

 ThreadID        Thread identifier.

 Acquisitions    The number of successful lock attempts for this lock.

 Miss Rate       The number of unsuccessful lock attempts divided by
                 Acquisitions, multiplied by 100.

 Spin Count      The number of unsuccessful lock attempts.

 Wait Count      The number of unsuccessful lock attempts that resulted in
                 the attempting thread going to sleep to wait for the lock
                 to become available.

Transform Count The number of times that a simple lock has allocated a krlock,
                while the thread was trying to acquire the simple lock.

Busy Count      The number of simple_lock_try() calls that returned busy.

 Percent Held of Total Time
 CPU              The percent of combined elapsed trace time that
                  threads held the lock in question while dispatched.
                  DISPATCHED_HOLDTIME_IN_SECONDS divided by trace
                  duration, multiplied by 100.

 Elaps(ed)        The percent of combined elapsed trace time that
                  threads held the lock while dispatched or sleeping.
                  UNDISPATCHED_AND_DISPATCHED_HOLDTIME_IN_SECONDS divided
                  by trace duration, multiplied by 100.

 Spin             The percent of combined elapsed trace time that
                  threads spun while waiting to acquire this lock.
                  SPIN_HOLDTIME_IN_SECONDS divided by combined trace
                  duration, multiplied by 100.

 Wait             The percent of combined elapsed trace time that
                  threads unsuccessfully tried to acquire this lock.

 ProcessID        Process identifier (only for SIMPLE and COMPLEX Lock report).

 Process Name     Name of the process (only for SIMPLE and COMPLEX Lock report).
```

Splat Sorting

splat allows the user to specify which criteria is used to sort the summary and lock detail reports using the **-s** option. The default sorting criteria is to sort by percent spin hold time, which is the ratio of time that threads spent spinning for a lock compared to the combined duration of the trace. Using **-s**, the sort criteria can be changed to the following:

Item	Description
a	Acquisitions; the number times a thread successfully acquired a lock.
c	Percent processor hold time; the ratio of processor hold time with the combined trace duration.
e	Percent Elapsed hold time; the ratio of elapsed hold time with the combined trace duration.
l	location; the address of the lock or function, or the ID of a thread.
m	Miss rate; the ratio missed lock attempts with the number of acquisitions.
s	Spin count; the number of unsuccessful lock attempts that result in a thread spinning waiting for the lock.
S	Percent processor spin hold time (default).

Item	Description
w	Percent elapsed wait time; the percent of the total time that a nonzero number of threads waited on the lock.
W	Average **waitq** depth; the average number of threads waiting on the lock, equivalent to the average time each waiting thread spends in this state.

splat will use the specified criteria to sort the lock reports in descending order.

Restrictions

Other types of locks, such as VMM, XMAP, and certain Java-specific locks are not analyzed.

Files

Item	Description
/etc/bin/splat	Simple Performance Lock Analysis Tool (**splat**). Provides kernel and pthread lock usage reports.

split Command

Purpose

Splits a file into pieces.

Syntax

To Split a File Into Multiple Files Containing a Specified Number of Lines

split [**-l** *LineCount*] [**-a** *SuffixLength*] [*File* [*Prefix*]]

To Split a File Into Multiple Files Containing a Specified Number of Bytes

split -b *Number* [**k** | **m**] [**-a** *SuffixLength*] [*File* [*Prefix*]]

Description

The **split** command reads the specified file and writes it in 1000-line pieces to a set of output files. The name of the first output file is constructed by combining the specified prefix (*x* by default) with the *aa* suffix, the second by combining the prefix with the *ab* suffix, and so on lexicographically through *zz* (a maximum of 676 files). The number of letters in the suffix, and consequently the number of output name files, can be increased by using the **-a** flag.

You cannot specify a *Prefix* longer than **PATH_MAX** - 2 bytes (or **PATH_MAX** - *SuffixLength* bytes if the **-a** flag is specified). The **PATH_MAX** variable specifies the maximum path-name length for the system as defined in the **/usr/include/sys/limits.h** file.

If you do not specify an input file or if you specify a file name of - (minus sign), the **split** command reads standard input.

The **split** command can be used with any regular text or binary files. After a file has been split, it can be restored to its original form by using the **cat** command, and the file fragments will be listed in the appropriate order.

Flags

Note: The **-b** and **-l** flags are mutually exclusive.

Item	Description
-a *SuffixLength*	Specifies the number of letters to use in forming the suffix portion of the output name files. The number of letters determines the number of possible output filename combinations. The default is two letters.
-b *Number*	Splits the file into the number of bytes specified by the *Number* variable. Adding the *k* (kilobyte) or *m* (megabyte) multipliers to the end of the *Number* value causes the file to be split into *Number**1024 or *Number**1,048,576 byte pieces, respectively.
-l *LineCount*	Specifies the number of lines in each output file. The default is 1000 lines.

Exit Status

This command returns the following exit values:

Item	Description
0	The command ran successfully.
>0	An error occurred.

Examples

1. To split a file into 1000-line segments, enter:

   ```
   split book
   ```

   This example splits book into 1000-line segments named `xaa`, `xab`, `xac`, and so forth.
2. To split a file into 50-line segments and specify the file-name prefix, enter:

   ```
   split -l 50 book sect
   ```

   This example splits book into 50-line segments named `sectaa`, `sectab`, `sectac`, and so forth.
3. To split a file into 2KB segments, enter:

   ```
   split -b 2k book
   ```

   This example splits the book into 2*1024-byte segments named `xaa`, `xab`, `xac`, and so forth.
4. To split a file into more than 676 segments, enter:

   ```
   split -l 5 -a 3 book sect
   ```

   This example splits a book into 5-line segments named `sectaaa`, `sectaab`, `sectaac`, and so forth, up to `sectzzz` (a maximum of 17,576 files).

Files

Item	Description
/usr/bin/split	Contains the **split** command.

splitlvcopy Command

Purpose

Splits copies from one logical volume and creates a new logical volume from them.

Syntax

splitlvcopy [**-f**] [**-y** *NewLogicalVolumeName*] [**-Y** *Prefix*] *LogicalVolume Copies* [*PhysicalVolume* ...]

Description

Note:

1. To use this command, you must either have `root` user authority or be a member of the system group.
2. The **splitlvcopy** command is not allowed on a snapshot volume group or a volume group that has a snapshot volume group.
3. The **splitlvcopy** command is not supported on encrypted logical volumes.

Attention: Although the **splitlvcopy** command can split logical volumes that are open, including logical volumes containing mounted filesystems, this is not recommended. You may lose consistency between *LogicalVolume* and *NewLogicalVolume* if the logical volume is accessed by multiple processes simultaneously. When splitting an open logical volume, you implicitly accept the risk of potential data loss and data corruption associated with this action. To avoid the potential corruption window, close logical volumes before splitting and unmount filesystems before splitting.

The **splitlvcopy** command removes copies from each logical partition in *LogicalVolume* and uses them to create *NewLogicalVolume*. The *Copies* parameter determines the maximum number of physical partitions that remain in *LogicalVolume* after the split. Therefore, if *LogicalVolume* has 3 copies before the split, and the *Copies* parameter is 2, *LogicalVolume* will have 2 copies after the split and *NewLogicalVolume* will have 1 copy. You can not split a logical volume so that the total number of copies in *LogicalVolume* and *NewLogicalVolume* after the split is greater than the number of copies in *LogicalVolume* before the split.

The *NewLogicalVolume* will have all the same logical volume characteristics as *LogicalVolume*. If *LogicalVolume* does not have a logical volume control block the command will succeed with a warning message and creates *NewLogicalVolume* without a logical volume control block.

There are additional considerations to take when splitting a logical volume containing a filesystem. After the split there will be two logical volumes but there will only be one entry in the **/etc/filesystems** file which refers to *LogicalVolume*. To access *NewLogicalVolume* as a filesystem you must create an additional entry in **/etc/filesystems** with a different mount point which refers to *NewLogicalVolume*. If the mount point does not already exist, you have to create it before the new filesystem can be mounted. In addition, if *NewLogicalVolume* was created while *LogicalVolume* was open, you have to run the command

```
fsck /dev/NewLogicalVolume
```

before the new filesystem can be mounted.

You can not use the System Management Interface Tool (SMIT) to run this command. Message catalogs are not supported for this command and therefore the error messages are provided in English only with no message catalog numbers. Documentation for splitlvcopy consists of this man page.

Flags

Item	Description
-f	Specifies to split open logical volumes without requesting confirmation. By default, **splitlvcopy** requests confirmation before splitting an open logical volume. This includes open raw logical volumes and logical volumes containing mounted filesystems.
-y *NewLogicalVolumeName*	Specifies the name of the new logical volume to move copies to from *LogicalVolume*.

Item	Description
-Y *Prefix*	Specifies the *Prefix* to use instead of the prefix in a system-generated name for the new logical volume. The prefix must be less than or equal to 13 characters. A name cannot begin with a prefix already defined in the **PdDv** class in the Device Configuration Database for other devices, nor be a name already used by another device.

Parameters

Item	Description
Copies	Specifies the maximum number of physical partitions that remain in LogicalVolume after the split.
LogicalVolume	Specifies the logical volume name or logical volume ID to split.
PhysicalVolume	Specifies the physical volume name or the physical volume ID to remove copies from.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command or be a member of the system group.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Auditing Events: N/A

Examples

To split one copy of each logical partition belonging to logical volume named **oldlv** which currently has 3 copies of each logical partition, and create the logical volume **newlv**, enter:

```
splitlvcopy -y newlv oldlv 2
```

Each logical partition in the logical volume **oldlv** now has two physical partitions. Each logical partition in the logical volume **newlv** now has one physical partition.

Files

Item	Description
/usr/sbin/splitlvcopy	Contains the **splitlvcopy** command.
/tmp	Contains the temporary files created while the **splitlvcopy** command is running.

splitvg Command

Purpose

Splits a single mirror copy of a fully mirrored volume group.

Syntax

```
splitvg  [ -y SnapVGname ]  [ -c  Copy ] [ -f  ] [ -i ]  VGname
```

Description

The **splitvg** command splits a single mirror copy of a fully mirrored volume group into a snapshot volume group. The original volume group ***VGname*** will stop using the disks that are now part of the snapshot volume group ***SnapVGname***. Both volume groups will keep track of the writes within the volume group so that when the snapshot volume group is rejoined with the original volume group consistent data is maintained across the rejoined mirrors copies.

Note:

1. To split a volume group, all logical volumes in the volume group must have the target mirror copy and the mirror must exist on a disk or set of disks. Only the target mirror copy must exist on the target disk or disks.
2. The **splitvg** command will fail if any of the disks to be split are not active within the original volume group.
3. In the unlikely event of a system crash or loss of quorum while running this command, the **joinvg** command must be run to rejoin the disks back to the original volume group.
4. New logical volumes and file system mount points will be created in the snapshot volume group.
5. When the **splitvg** command targets a concurrent-capable volume group which is varied on in non-concurrent mode, the new volume group that is created will not be varied on when the **splitvg** command completes. The new volume group must be varied on manually.
6. The Platform keystore (PKS) and key file authentication methods of an encrypted logical volume are not supported in snapshot volume group to access the data.
7. You cannot remove the authentication methods of an encrypted logical volume in a primary volume group.
8. You cannot add or remove an authentication method of an encrypted logical volume in a snapshot volume group.

Flags

Item	Description
-y *SnapVGname*	Allows the volume group name to be specified rather than having the name generated automatically. Volume group names must be unique across the system and can range from 1 to 15 characters. The name cannot begin with a prefix already defined in the **PdDv** class in the Device Configuration database for other devices. The new volume group name is sent to standard output.
-c *Copy*	Which mirror to split. Valid values are 1, 2, or 3. The default is the second copy.
-f	Will force the split even if the mirror copy specified to create the snapshot volume group has stale partitions.
-i	Will split the mirror copy of a volume group into a new volume group that cannot be rejoined into the original.

Security

Access Control: You must have `root` authority to run this command.

Examples

1. To split a volume group, enter:

   ```
   splitvg testvg
   ```

   The second mirror copy of the volume group **testvg** is split into new volume group with an automatically generated name, which will be displayed.
2. To split first mirror copy of the volume group with the name **snapvg**, enter:

   ```
   splitvg -y snapvg -c 1 testvg
   ```

Files

Item	Description
/usr/sbin	Directory where the **splitvg** command resides.

splp Command

Purpose

Changes or displays printer driver settings.

Syntax

splp [**-b** *Option*] [**-B** *Number*] [**-c** *Option*] [**-C** *Option*] [**-e** *Option*] [**-f** *Option*] [**-F!**] [**-i** *Number*] [**-l** *Number*] [**-n** *Option*] [**-N** *Option*] [**-p** *Option*] [**-P** *Option*] [**-r** *Option*] [**-s** *Number*] [**-S** *Option*] [**-t** *Option*] [**-T** *Number*] [**-w** *Number*] [**-W** *Option*] [*DevicePath*]

Description

The **splp** command changes or displays settings for a printer device driver. The default device path is **/dev/lp0**; all flags are optional. If the device path does not begin with a **/** (backslash) character, the **/dev** directory is assumed. Also, if no flags are specified, the **splp** command reports the current settings for the specified device path. To change the current settings, specify the appropriate flags. No other processing is done, and there is no other output.

The changes that the **splp** command makes remain in effect until the next time you restart the system or rerun the **splp** command. The **splp** command can be run from the **/etc/inittab** command file to configure your printer each time you start up the system.

> **Note:** The **splp** command settings for the **-b**, **-c**, **-C**, **-f**, **-i**, **-l**, **-n**, **-p**, **-r**, **-t**, **-w**, and **-W** flags apply only when data is sent directly to the printer device (for example, redirecting the output of the **cat** command directly to the specifies device path). When files are queued for printing with the **enq**, **qprt**, **lp**, or **lpr** commands, the settings for these flags are ignored and are not changed.

Flags

Item	Description
-b *Option*	Specifies whether backspaces are sent to the printer: **+** Specifies backspaces be sent to the printer. **!** Specifies backspaces be discarded.
-B *Number*	Sets the speed to the specified number of bits per second. Values for the *Number* variable are 50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2400, 4800, 9600, 19,200, and 38,400.
-c *Option*	Specifies whether carriage returns are sent to the printer: **+** Sends carriage returns to the printer. **!** Translates carriage returns to line feeds.
-C *Option*	Specifies whether all lowercase characters are converted to uppercase characters: **+** Converts lowercase characters to uppercase characters. **!** Does not convert lowercase characters to uppercase characters.
-e *Option*	Specifies the processing to be performed when an error is detected: **+** Returns an error. **!** Waits until error clears.
-f *Option*	Specifies whether the printer is sent form feeds or simulates a form feed with line feeds or carriage returns: **+** Sends form feeds to the printer. **!** Simulates a form feed with line feeds or carriage returns.
-F!	Resets font status indicators for an 3812 Page Printer or an 3816 Page Printer. This flag causes fonts to be reloaded from the printer's font diskette into the printer's memory by the next spooled print job. This flag should be specified if the printer has been turned off and then turned back on, or if the fonts in the printer's memory have become corrupted.
-i *Number*	Indents the specified number of columns, where the value of the *Number* variable is an integer.
-l *Number*	Prints the specified number of lines per page, where the value of the *Number* variable is an integer.
-n *Option*	Specifies whether the printer is sent line feeds or translates line feeds to carriage returns: **+** Sends line feeds to the printer. **!** Translates line feeds to carriage returns.

Item	Description
-N *Option*	Specifies whether parity generation and detection is enabled: **+** Enables parity generation and detection. **!** Disables parity generation and detection.
-p *Option*	Specifies whether the system sends all characters to the printer unmodified or translates characters according to the settings for the **-b**, **-c**, **-C**, **-f**, **-i**, **-l**, **-n**, **-r**, **-t**, **-w**, and **-W** flags: **+** Sends all characters to the printer unmodified, overriding other settings. **!** Translates characters according to the settings.
-P *Option*	Specifies the parity: **+** Specifies odd parity. **!** Specifies even parity.
-r *Option*	Specifies whether carriage returns are added after line feeds: **+** Sends a carriage return after a line feed. **!** Does not send a carriage return after a line feed.
-s *Number*	Selects character size where the *Number* variable is the number of bits. Values for the *Number* variable can be 5, 6, 7, or 8. See the **termio.h** special file for additional information on character size.
-S *Option*	Specifies the number of stop bits per character: **+** 2 stop bits per character. **!** 1 stop bit per character.
-t *Option*	Specifies whether tabs are to be expanded: **+** Does not expand tabs. **!** Expands tabs on 8 position boundaries.
-T *Number*	Sets the time-out period to the number of seconds specified by the *Number* variable. The value of the *Number* variable must be an integer.
-w *Number*	Prints the number of columns specified by the *Number* variable. The value of the *Number* variable must be an integer.

Item	Description
-W *Option*	Specifies whether to wrap characters beyond the specified width to the next line and print `...` (3 dots) after the new-line character: **+** Wraps characters beyond the specified width to the next line and prints `...` (3 dots) after the new-line character. **!** Truncates characters beyond the specified width.

Examples

1. To display the current printer settings for the **/dev/lp0** printer, enter:

```
splp
```

2. To change the printer settings, enter:

```
splp   -w 80  -W +  -C +
```

This changes the settings of the **/dev/lp0** printer for 80-column paper (the **-w 80** flag). It also wraps each line that is more than 80 columns wide onto a second line (the **-W+** flag), and prints all alphabetic characters in uppercase (the **-C+** flag).

Files

Item	Description
/dev/lp*	Contains the printer attribute file.
/etc/inittab	Contains the printer configuration command file.

spost Command

Purpose

Routes a message.

Syntax

spost [**-noalias** | **-alias** *File ...*] [**-format** | **-noformat**] [**-filter** *File* | **-nofilter**] [**-width** *Number*] [**-watch** | **-nowatch**] [**-remove** | **-noremove**] [**-backup** | **-nobackup**] [**-verbose** | **-noverbose**]*File*

Description

The **spost** command routes messages to the correct destinations. The **spost** command is not started by the user. The **spost** command is called by other programs only.

The **spost** command searches all components of a message that specify a recipient's address and parses each address to check for proper format. The **spost** command then puts addresses in the standard format and starts the **sendmail** command. The **spost** command performs a function similar to the **post** command, but it does less address formatting than the **post** command.

The **spost** command is the default (over the **post** command). Change the default by setting the **postproc** variable in your **.mh_profile**. For example:

```
postproc: /usr/lib/mh/post
```

The *File* parameter is the name of the file to be posted.

Flags

Item	Description
-alias *File*	Searches the specified mail alias file for addresses. You can repeat this flag to specify multiple mail alias files. The **spost** command automatically searches the **/etc/mh/MailAliases** file.
-backup	Renames the message file by placing a , (comma) before the file name after the **spost** command successfully posts the message.
-filter *File*	Uses the header components in the specified file to copy messages sent to the `Bcc:` field recipients.
-format	Puts all recipient addresses in a standard format for the delivery transport system. This flag is the default.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.
-noalias	Does not use any alias files for delivering the message.
-nobackup	Does not rename the message after posting the file. This flag is the default.
-nofilter	Strips the `Bcc:` field header from the message and sends it to recipients specified in the `Bcc:` component. This flag is the default.
-noformat	Does not alter the format of the recipient addresses.
-noremove	Does not remove the temporary message file after posting the message.
-noverbose	Does not display information during the delivery of the message to the **sendmail** command. This flag is the default.
-nowatch	Does not display information during delivery by the **sendmail** command. This flag is the default.
-remove	Removes the temporary message file after the message has been successfully posted. This flag is the default.
-verbose	Displays information during the delivery of the message to the **sendmail** command. This information allows you to monitor the steps involved.
-watch	Displays information during the delivery of the message by the **sendmail** command. This information allows you to monitor the steps involved.
-width *Number*	Sets the width of components that contain addresses. The default is 72 columns.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Files

Item	Description
$HOME/.mh_profile	Contains the Message Handler (MH) user profile.
/tmp/pst*Number*	Contains the temporary message file.

Item	Description
/etc/mh/MailAliases	Contains the default mail aliases.
/usr/lib/mh/.mh_profile	Contains the Message Handler (MH) user profile.

spray Command

Purpose

Sends a specified number of packets to a host and reports performance statistics.

Syntax

/usr/sbin/spray *Host* [**-c** *Count*] [**-d** *Delay*] [**-i**] [**-l** *Length*]

Description

The **spray** command uses the Remote Procedure Call (RPC) protocol to send a one-way stream of packets to the host you specify. This command reports how many packets were received and at what transfer rate. The *Host* parameter can be either a name or an Internet address. The host only responds if the **sprayd** daemon is running.

Note: The **spray** command does not support IPv6.

See the **rpc.sprayd** daemon documentation for factors that affect **spray** command performance.

Flags

Item	Description
-c *Count*	Specifies the number of packets to send. The default value is the number of packets required to make the total stream size 100,000 bytes.
-d *Delay*	Specifies the time, in microseconds, the system pauses between sending each packet. The default is 0.
-i	Uses the Internet Control Message Protocol (ICMP) echo packets rather than the RPC protocol. Since ICMP echoes automatically, it creates a two-way stream. You must be root user to use this option.
-l *Length*	Specifies the number of bytes in the packet that holds the RPC call message. The default value of the *Length* parameter is 86 bytes, the size of the RPC and UDP headers. The data in the packet is encoded using eXternal Data Representation (XDR). Since XDR deals only with 32-bit quantities, the **spray** command rounds smaller values up to the nearest possible value. When the *Length* parameter is greater than 1500 for Ethernet or 1568 for token-ring, the RPC call can no longer fit into one Ethernet packet. Therefore, the *Length* field no longer has a simple correspondence to Ethernet packet size.

Examples

1. When sending a **spray** command to a workstation, specify the number of packets to send and the length of time the system will wait between sending each packet as follows:

   ```
   /usr/sbin/spray zorro  -c 1200 -d 2
   ```

   In this example, the `spray` command sends `1200` packets at intervals of 2 microseconds to the workstation named `zorro`.

2. To change the number of bytes in the packets you send, enter:

   ```
   /usr/sbin/spray zorro  -l 1350
   ```

   In this example, the `spray` command sends `1350`-byte packets to the workstation named `zorro`.
3. To send echo packets using the ICMP protocol instead of the RPC protocol, enter:

   ```
   /usr/sbin/spray zorro  -i
   ```

   In this example, the `spray` command sends echo packets to the workstation named `zorro`.

sprayd Daemon

Purpose

Receives packets sent by the **spray** command.

Syntax

/usr/lib/netsvc/spray/rpc.sprayd

Description

The **rpc.sprayd** daemon is a server that records the packets sent by the **spray** command. The **rpc.sprayd** daemon is normally started by the **inetd** daemon.

UDP Performance

User Datagram Protocol (UDP) performance with the **spray** command and the **rpc.sprayd** daemon can be affected by the following factors:

- How memory buffers (mbufs) are tuned for system configuration.
- The incoming burst rate (that is, interframe gap) of UDP packets for the **spray** command.
- Other system activity. Since the **rpc.sprayd** daemon runs as a normal user process, other activity (such as the **init** process, or the **syncd** daemon) can affect the operation of the **rpc.sprayd** daemon.
- Priority of the **rpc.sprayd** daemon process. The **rpc.sprayd** daemon has a floating process priority that is calculated dynamically.
- The size of the receive socket buffer used by the **rpc.sprayd** daemon. Because various implementations use different socket buffer sizes, measuring UDP performance with the **spray** command and the **rpc.sprayd** daemon is difficult and inconclusive.

Files

Item	Description
/etc/inetd.conf	TCP/IP configuration file that starts RPC daemons and other TCP/IP daemons.

srcmstr Daemon

Purpose

Starts the System Resource Controller.

Syntax

srcmstr /usr/sbin/srcmstr [**-r**] [**-B**]

Description

The **srcmstr** daemon is the System Resource Controller (SRC). The **srcmstr** daemon creates and controls subsystems, handles short subsystem status requests, passes requests on to a subsystem, and handles error notification.

The **srcmstr** daemon is normally started by using an **inittab** file entry.

Flags

Item	Description
-r	Accepts remote requests if the daemon is started with the **-r** flag. If you start **srcmstr** without the **-r** flag, remote requests are ignored.
-B	Specifies the **-B** flag that causes the **srcmstr** daemon to run as in previous releases (AIX 4.3.1 and earlier). **Note:** 1. The **srcmstr** daemon is typically started from **inittab**. To add the **-r** or **-B** flags, edit **/etc/inittab** and run **init q** or reboot. 2. The user must be running as root on the remote system. The local **/etc/hosts.equiv** file or the **/.rhosts** file must be configured to allow remote requests.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Auditing Events: If the auditing subsystem has been properly configured and is enabled, the **srcmstr** command will generate the following audit record (event) every time the command is executed:

Event	Information
SRC_Start	Lists in an audit log the name of the subsystems being started.
SRC_Stop	Lists in an audit log the name of the subsystems being stopped.

See **Setting Up Auditing** in *Security* for more details about how to properly select and group audit events, and how to configure audit event data collection.

Error Recovery

The default **/etc/inittab** specifies the **respawn** flag for the **srcmstr** daemon. If the **srcmstr** daemon terminates abnormally and the **/etc/inittab** specifies the **respawn** flag, the **srcmstr** daemon is restarted. It then determines which SRC subsystems were active during the previous invocation. The daemon re-establishes communication with these subsystems (if it existed previously), and initializes a private kernel extension and the **srcd** daemon to monitor the subsystem processes.

If a subsystem known to the previous invocation of **srcmstr** terminates, the SRC kernel extension notifies the **srcd** daemon. The **srcd** daemon sends a socket message to **srcmstr** and subsystem termination is handled as if the subsystem had been started by the current **srcmstr**. This function can be disabled by specifying the **-B** flag when the **srcmstr** daemon is started. The SRC kernel extension is in **/usr/lib/drivers/SRC_kex.ext**. The executable for **srcd** is **/usr/sbin/srcd**.

Files

Item	Description
/etc/inittab	Specifies stanzas read by the **init** command.
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration Object Class.
/etc/objrepos/SRCnotify	Specifies the SRC Notify Method Object Class.
/etc/hosts.equiv	Specifies that no remote requests will work if the specified host name is not in the **/etc/hosts.equiv** file.
/etc/services	Defines the sockets and protocols used for Internet services.
/dev/SRC	Specifies the **AF_UNIX** socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.
/dev/.SRC-unix/SRCD	Specifies the **AF_UNIX** socket file for the **srcd** daemon.
/var/adm/SRC/active_list	Contains a list of active subsystems. **Caution:** The structure of this file is internal to SRC and is subject to change.
/var/adm/SRC/watch_list	Contains a list of subsystem processes active during the previous invocation of the **srcmstr** daemon. **Caution:** The structure of this file is internal to SRC and is subject to change.
/.rhosts	Specifies remote machines and users (root only) that are allowed to request SRC function from this machine.

start-secldapclntd Command

Purpose

The **start-secldapclntd** script is used to start the **secldapclntd** LDAP client daemon.

Syntax

/usr/sbin/start-secldapclntd [**-C** *CacheSize*] [**-p** *NumOfThread*] [**-t** *CacheTimeOut*] [**-T** *HeartBeatIntv*] [**-o** *ldapTimeOut*]

Description

The **start-secldapclntd** script starts the **secldapclntd** daemon if it is not running. It does not do anything if the **secldapclntd** daemon is already running. The script also cleans the portmapper registration (if there is any) from previous **secldapclntd** daemon process before it starts the **secldapclntd** daemon. This prevents the startup failure of the new daemon process from portmap-per registration failure.

Flags

By default, the **secldapclntd** daemon reads the configuration information specified in the **/etc/security/ldap/ldap.cfg** file at startup. If the following options are given in command line when starting **secldapclntd** process, the options from the command line will overwrite the values in the **/etc/security/ldap/ldap.cfg** file.

Item	Description
-C *CacheSize*	Sets the maximum cache entries used by the **secldapclntd** daemon to CacheSize number of entries. The valid range is 100-65536 entries for user cache entry. The default value is 1000. The valid range is 10-65536 for group cache entry. The default value is 100. If you set the user cache entry in the **start-secldapclntd** command by using the **-C** option, the group cache entry is set to 10% of the user cache entry.
-o *ldapTimeOut*	Timeout period in seconds for LDAP client requests to the server. This value determines how long the client will wait for a response from the LDAP server. Valid range is 0 - 3600 (1 hour). Default is 60 seconds. Set this value to 0 to disable the timeout and force the client to wait indefinitely.
-p *NumOfThread*	Sets the number of thread used by the **secldapclntd** daemon to **NumOfThread** threads. Valid range is 1-256. The default is 10.
-t *CacheTimeout*	Sets the cache to expire in CacheTimeout seconds. Valid range is 60- 3600 seconds. The default is 300 seconds.
-T *HeartBeatIntv*	Sets the time interval of heartbeat between this client and the LDAP server. Valid values are 60-3,600 seconds. Default is 300.

Security

A user with the **aix.security.ldap** authorization is authorized to use this command.

Examples

1. To start the **secldapclntd** daemon, type:

   ```
   /usr/sbin/start-secldapclntd
   ```

2. To start the **secldapclntd** with using 20 threads and cache timeout value of 600 seconds, type:

   ```
   /usr/sbin/start-secldapclntd -p 20 -t 600
   ```

   It is recommended that you specify these values in the **/etc/security/ldap/ldap.cfg** file, so that these values will be used each time you start the **secldapclntd** process.

Files

Item	Description
/usr/sbin/start-secldapclntd	Used to start the **secldapclntd** LDAP client daemon.

startcondresp Command

Purpose

Starts monitoring a condition that has one or more linked responses.

Syntax

To start monitoring a condition:

`startcondresp` [`-h`] [`-TV`] *condition*[`:`*node_name*] [*response* [*response*...]]

To unlock or lock the condition/response association:

`startcondresp` {`-U` | `-L`} [`-h`] [`-TV`] *condition*[`:`*node_name*] *response*

Description

The `startcondresp` command starts the monitoring of a condition that has a linked response. A link between a condition and a response is called a *condition/response association.* In a cluster environment, the condition and the response must be defined on the same node. After monitoring is started, when the condition occurs, the response is run. If no responses are specified, monitoring is started for all responses linked to the condition. This causes all of the linked responses to run when the condition occurs. If more than one response is specified, monitoring is started only for those linked responses.

If one or more responses are specified and the responses are not linked with the condition, the `startcondresp` command links the specified responses to the condition, and monitoring is started. Use the `mkcondresp` command to link a response to a condition without starting monitoring.

If a particular condition/response association is needed for system software to work properly, it may be locked. A locked condition/response association cannot be started by the `startcondresp` command. If the condition/response association you specify on the `startcondresp` command is locked, it will not be started; instead an error will be generated informing you that this condition/response association is locked. To unlock a condition/response association, you can use the `-U` flag. However, because a condition/response association is typically locked because it is essential for system software to work properly, you should exercise caution before unlocking it. To lock a condition/response association so it cannot be started, stopped, or removed, reissue this command using its `-L` flag.

Flags

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

-U
Unlocks a condition/response association so it can be started, stopped, or removed. If a condition/response association is locked, this is typically because it is essential for system software to work properly. For this reason, you should exercise caution before unlocking it. When unlocking a condition/response association using the `-U` flag, no other operation can be performed by this command.

-L
Locks a condition/response association so it cannot be started, stopped, or removed. When locking a condition/response association using the `-L` flag, no other operation can be performed by this command.

Parameters

condition
Specifies the name of the condition linked to the response. The condition is always specified first.

node_name
Specifies the node in the domain where the condition is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

response
Specifies the name of one or more responses. Specifying more than one response links the responses to the condition if they are not already linked and starts monitoring for the specified responses.

Security

The user needs write permission for the `IBM.Association` resource class to run `startcondresp`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To start monitoring for the condition "FileSystem space used " by using the response "Broadcast event on-shift", whether or not the response is linked with the condition, run this command:

```
startcondresp  "FileSystem space used" "Broadcast event on-
shift"
```

2. To start monitoring for the condition "FileSystem space used " by using all of its linked responses, run this command:

```
startcondresp "FileSystem space used"
```

3. To start monitoring for the condition "FileSystem space used " by using the response "Broadcast event on-shift" and "E-mail root anytime", whether or not they are linked with the condition, run this command:

```
startcondresp "FileSystem space used" "Broadcast event on-shift" "E-mail root anytime"
```

These examples apply to management domains:

1. To start monitoring for the condition "FileSystem space used" on the management server using the response "Broadcast event on-shift", whether or not the response is linked with the condition, run this command on the management server:

```
startcondresp  "FileSystem space used" "Broadcast event on-
shift"
```

2. To start monitoring for the condition "FileSystem space used" on the managed node nodeB using the response "Broadcast event on-shift", whether or not the response is linked with the condition, run this command on the management server:

```
startcondresp  "FileSystem space used":nodeB "Broadcast event on-
shift"
```

This example applies to peer domains:

1. To start monitoring for the condition "FileSystem space used" on nodeA in the domain using the response "Broadcast event on-shift" (also on nodeA in the domain), whether or not the response is linked with the condition, run this command on any node in the domain:

```
startcondresp  "FileSystem space used":nodeA "Broadcast event on-shift"
```

Location

/opt/rsct/bin/startcondresp

startrpdomain Command

Purpose

Brings a peer domain that has already been defined online.

Syntax

`startrpdomain` [`-A` | `-L`] [`-t` *timeout*] [`-Q` *quorum_type* | *quorum_type_name*] [`-m` *fanout*] [`-h`] [`-w` [`-s` *Seconds*]] [`-TV`] *peer_domain*

Description

The `startrpdomain` command brings a defined peer domain online by starting the resources on each node belonging to the peer domain.

The `startrpdomain` command must be run on a node that is defined to the peer domain. The command invites all offline nodes defined to the peer domain to come online in the peer domain every time the command is run for the peer domain. The command can be run more than once in the peer domain. If all the nodes defined in the peer domain are already online, no action is performed.

The `startrpdomain` command determines the peer domain configuration to use to bring the peer domain online by examining the peer domain configuration on the nodes defined to the peer domain. The latest version of the peer domain configuration information that is found is used to bring the peer domain online. By default, the latest version of the peer domain configuration found on at least half of the nodes is used. Specifying the `-A` flag causes the latest version of the peer domain configuration found on all of the nodes defined in the peer domain to be used. Specifying the `-L` flag causes the configuration on the local node to be used.

In determining the latest version of the peer domain configuration information, a configuration timeout defines when to stop checking versions and begin to bring the peer domain online. The default timeout value is 120 seconds. The timeout value can be changed using the `-t` flag. The timeout value should be at least long enough so that the latest version of the peer domain configuration information from at least half of the nodes can be found.

A node can only be online to one peer domain at a time. The `startrpdomain` command cannot be run on a node for a peer domain when another peer domain is already online for that node.

Flags

-A
: Finds and uses the latest version of the peer domain configuration information from all of the nodes in the peer domain. This flag cannot be specified if the `-L` flag is specified. If neither flag (`-A` or `-L`) is specified, the latest version of the peer domain configuration information from at least half of the nodes in the peer domain is used.

-L
: Uses the latest version of the peer domain configuration information that is on the local node. This flag cannot be specified if the `-A` flag is specified. If neither flag (`-A` or `-L`) is specified, the latest version of the peer domain configuration information from at least half of the nodes in the peer domain is used.

-t *timeout*
: Specifies the timeout value in seconds. This flag limits the amount of time used to find the latest version of the peer domain configuration. When the timeout value is exceeded, the latest version of the peer domain configuration information found thus far is used. The timeout value should be long enough so that the latest version of the peer domain configuration information from at least half of the nodes can be found. The default timeout value is 120 seconds.

-Q *quorum_type* | *quorum_type_name*
Enables you to override the startup quorum mode. This can be specified as an integer quorum type or quorum type name. If you do not specify this flag, startup quorum mode will be specified using the `mkrpdomain` command's `-Q` flag (or the default quorum mode for your environment) when you created the peer domain. You can override the quorum startup mode only if the quorum mode has been defined as `normal` or `quick`. The valid values are:

0 | **normal**
Specifies normal start-up quorum rules. Half of the nodes will be contacted for configuration information.

1 | **quick**
Specifies quick start-up quorum rules. One node will be contacted for configuration information.

-m *fanout*
Specifies the maximum number of threads to use for this start operation. The `-m` flag overrides the default *fanout* value for the specified peer domain. This value is stored as a persistent attribute in the peer domain's `IBM.PeerNode` class. *fanout* can be an integer from 16 to 2048.

-h
Writes the command's usage statement to standard output.

-s
Specifies the wait time in seconds for the peer domain to be online before the command completes when the **-s** flag is used with the **-w** flag. If the waiting time exceeds the number of seconds, the command returns, but the online operation continues. The default value is 300 seconds (5 minutes). Use 0 to specify that the command must not return until the peer domain is online (no timeout on waiting).

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

-w
Waits for the peer domain to be online before the command completes. Use the **-s** flag to specify the waiting time in seconds.

Parameters

peer_domain
Specifies the name of a previously-defined peer domain that is to be brought online.

Security

The user of the `startrpdomain` command needs write permission for the `IBM.PeerDomain` resource class on each node that is defined to the peer domain. By default, `root` on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

6
The peer domain definition does not exist.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run from a node that is defined to the peer domain.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the `-F "-"` flag is specified, this command reads one or more node names from standard input.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

In these examples, `nodeA` is one of the nodes defined to `ApplDomain`.

1. To bring `ApplDomain` online, run this command on `nodeA`:

```
startrpdomain ApplDomain
```

2. To bring `ApplDomain` online using all of the nodes in the peer domain to obtain the latest version of the peer domain configuration information, run this command on `nodeA`:

```
startrpdomain -A ApplDomain
```

3. To bring ApplDomain online using a peer domain configuration timeout value of 240 seconds (to make sure that at least half of the nodes in the peer domain are used), run this command on nodeA:

```
startrpdomain -t 240 ApplDomain
```

Location

/opt/rsct/bin/startrpdomain

startrpnode Command

Purpose

Brings one or more nodes online to a peer domain.

Syntax

`startrpnode` [`-h`] [`-w` [-s *Seconds*]] [`-TV`] *node_name1* [*node_name2* ...]

`startrpnode -f` | `-F` { *file_name* | `"–"` } [`-h`] [`-w` [-s *Seconds*]] [`-TV`]

Description

The `startrpnode` command brings one or more offline nodes online to a peer domain. The peer domain is determined by the online peer domain where the command is run. The command must be run from a node that is online to the desired peer domain.

The node that is being brought online must have already been defined to be in this peer domain using the `addrpnode` command or the `mkrpdomain` command. The node must not be online to any other peer domain.

Flags

-f | -F { *file_name* | "–" }
Reads a list of node names from *file_name*. Each line of the file is scanned for one node name. The pound sign (#) indicates that the remainder of the line (or the entire line if the # is in column 1) is a comment.

Use `-f "-"` or `-F "-"` to specify STDIN as the input file.

-h
Writes the command's usage statement to standard output.

-s
Specifies the wait time in seconds for all of the specified nodes to be online before the command completes when the **-s** flag is used with the **-w** flag. If the waiting time exceeds the number of seconds, the command returns, but the online operation continues. The default value is 300 seconds (5 minutes). Use 0 to specify that the command must not return until all of the specified nodes are online (no timeout on waiting).

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

-w
Waits for all of the specified nodes to be online before the command completes. Use the **-s** flag to specify the waiting time in seconds.

Parameters

***node_name1* [*node_name2* ...]**

Specifies the peer domain node names of the nodes to be brought online to the peer domain. You can bring one or more nodes online using the `startrpnode` command. You must specify the node names in exactly the same format as they were specified with the `addrpnode` command or the `mkrpdomain` command. To list the peer domain node names, run the `lsrpnode` command.

Security

The user of the `startrpnode` command needs write permission for the `IBM.PeerNode` resource class on each node that is to be started in the peer domain. By default, `root` on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Exit Status

0

The command ran successfully.

1

An error occurred with RMC.

2

An error occurred with a command-line interface script.

3

An incorrect flag was entered on the command line.

4

An incorrect parameter was entered on the command line.

5

An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT

Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT

When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run from a node that is online to the peer domain. The node that is to be brought online must be offline to the peer domain, must not be online to any other peer domain, and must be reachable from where the command is run.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the `-f "-"` or `-F "-"` flag is specified, this command reads one or more node names from standard input.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

In this example, `nodeA` is defined and online to `ApplDomain`, `nodeB` is reachable from `nodeA`, and `nodeB` is not online to `ApplDomain` or any other peer domain. To bring `nodeB` online to `ApplDomain`, run this command from `nodeA`:

```
startrpnode nodeB
```

Location

/opt/rsct/bin/startrpnode

startrsrc Command

Purpose

Starts a defined resource (that is, brings it online).

Syntax

To start one or more resources, using data entered on the command line:

`startrsrc -s "`*selection_string*`" [ -N { `*node_file* `| "-" } ] [-n` *node_name*`] [-h] [-TV]` *resource_class* [*arg=value...*]

`startrsrc -r [-n` *node_name*`] [-h] [-TV]` *resource_handle* [*arg=value...*]

To start one or more resources using command arguments that are predefined in an input file:

`startrsrc -f` *resource_data_input_file* `-s "`*selection_string*`" [ -N {` *node_file* `| "-" } ] [-n` *node_name*`] [-h] [-TV]` *resource_class*

`startrsrc -f` *resource_data_input_file* `-r [-n` *node_name*`] [-h] [-TV]` *resource_handle*

To list the names and data types of the command arguments:

`startrsrc -l [-h]` *resource_class*

Description

The `startrsrc` command requests that the resource monitoring and control (RMC) subsystem bring one or more resources online. The request is performed by the appropriate resource manager.

To start one or more resources, use the `-s`flag to bring online all of the resources that match the specified selection string.

Instead of specifying multiple node names in *selection_string*, you can use the `-N` *node_file* flag to indicate that the node names are in a file. Use `-N "-"` to read the node names from standard input.

To start one specific resource, use the `-r` flag to specify the resource handle that represents that specific resource.

Use the `-l` flag to determine whether the specified resource class accepts any additional command arguments.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM `nodegrp` command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

The successful completion of this command does not guarantee that the resource is online, only that the resource manager successfully received the request to bring this resource online. Monitor the dynamic attribute `OpState` of the resource to determine when the resource is brought online. Register an event for the resource, specifying the `OpState` attribute, to know when the resource is actually online. Or, intermittently run the `lsrsrc` command until you see that the resource is online (the value of `OpState` is 1). For example:

```
lsrsrc -s 'Name == "/filesys1"' -t IBM.FileSystem Name OpState
```

Parameters

resource_class
Specifies the name of the resource class that contains the resources that you want to bring online.

resource_handle
Specifies the resource handle that corresponds to the resource you want to bring online. Use the `lsrsrc` command to obtain a list of valid resource handles. The resource handle must be enclosed within double quotation marks, for example:

```
"0x4017 0x0001 0x00000000 0x0069684c 0x0d4715b0 0xe9635f69"
```

arg=value...
Specifies one or more pairs of command argument names and values.

arg
Specifies the argument name.

value
Specifies the value for this argument. The value data type must match the definition of the argument data type.

Command arguments are optional. If any *arg=value* pairs are entered, there must be one *arg=value* pair for each command argument defined for the online function for the specified resource class.

Use `startrsrc -l` to get a list of the command argument names and data types for the specific resource class.

Flags

-f *resource_data_input_file*
Specifies the name of the file that contains resource argument information. The contents of the file would look like this:

```
PersistentResourceArguments::
argument1 = value1
argument2 = value2
```

-l
Lists the command arguments and data types. Some resource managers accept additional arguments that are passed to the online request. Use this flag to list any defined command arguments and the data types of the command argument values.

-n *node_name*
Specifies the name of the node where the resource is to be brought online. *node_name* is a `NodeNameList` attribute value. Use this flag to bring a floating resource online on a different node if the node where it was online might be down.

Do *not* specify this flag if you want the resource to be brought online on the node where it is known.

-N { *node_file* | "-" }
Specifies that node names are read from a file or from standard input. Use `-N` *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (*#*) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use `-N "-"` to read the node names from standard input.

The CT_MANAGEMENT_SCOPE environment variable determines the scope of the cluster. If CT_MANAGEMENT_SCOPE is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and CT_MANAGEMENT_SCOPE is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-s "*selection_string*"
Specifies the selection string. All selection strings must be enclosed within either double or single quotation marks. If the selection string contains double quotation marks, enclose the entire selection string in single quotation marks. For example:

```
-s 'Name == "testing"'
-s 'Name ?= "test"'
```

Only persistent attributes can be listed in a selection string.

-h
Writes the command usage statement to standard output.

-T
Writes the command trace messages to standard error. For your software service organization use only.

-V
Writes the command verbose messages (if there are any available) to standard output.

Environment variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Standard output

When the `-h` flag is specified, this command usage statement is written to standard output. When the `-V` flag is specified, this command verbose messages (if there are any available) are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
No resources were found that match the specified selection string.

Security

You need write permission for the *resource_class* specified in `startrsrc` to run `startrsrc`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *Administering RSCT* guide for information about the ACL file and how to modify it.

Implementation specifics

This command is part of the `rsct.core.rmc` fileset for AIX operating system and `rsct.core-3.1.0.0-0.`*platform*`.rpm` package for Linux, Solaris, and Windows operating systems, where *platform* is `i386`, `ppc`, `ppc64`, `s390`, or `x86_64`.

Location

/opt/rsct/bin/startrsrc

Examples

Suppose that you have a peer domain called `foo` with three defined nodes: nodeA, nodeB, and nodeC. nodeA has two Ethernet cards: `ent0` and `ent1`.

1. Suppose nodeA is online and `ent0` (on nodeA) is offline. To bring `ent0` online on nodeA, run this command on nodeA:

   ```
   startrsrc -s 'Name == "ent0"' IBM.EthernetDevice
   ```

2. Suppose nodeA and nodeB are online, `ent0` (on nodeA) is offline, and you are currently logged on to nodeB. To bring `ent0` online on nodeA, run this command on nodeB:

   ```
   startrsrc -s 'Name == "ent0"' -n nodeA IBM.EthernetDevice
   ```

3. Suppose file system `/filesys1` is defined, but not mounted on nodeB. To bring `/filesys1` online on nodeB, run this command on nodeA:

   ```
   startrsrc -s 'Name == "/filesys1"' -n nodeB IBM.FileSystem
   ```

4. Suppose the resource handle for `ent0` on nodeA is:

   ```
   0x406b 0x0001 0x00000000 0x0069564c 0x0dc1f272 0xb9de145e
   ```

   To bring `ent0` online on nodeA, run this command on nodeA:

   ```
   startrsrc -r "0x406b 0x0001 0x00000000 0x0069564c 0x0dc1f272 0xb9de145e"
   ```

startsrc Command

Purpose

Starts a subsystem, a group of subsystems, or a subserver.

Syntax

To Start a Subsystem

startsrc [**-a** *Argument*] [**-e** *Environment*] [**-h** *Host*] { **-s** *Subsystem* | **-g** *Group*}

To Start a Subserver

startsrc [**-h** *Host*] **-t** *Type* [**-o** *Object*] [**-p** *SubsystemPID*]

Description

The **startsrc** command sends the System Resource Controller (SRC) a request to start a subsystem or a group of subsystems, or to pass on a packet to the subsystem that starts a subserver.

If a start subserver request is passed to the SRC and the subsystem to which the subserver belongs is not currently active, the SRC starts the subsystem and transmits the start subserver request to the subsystem.

Flags

Item	Description
-a *Argument*	Specifies an argument string that is passed to the subsystem when the subsystem is executed. This string is passed from the command line and appended to the command line arguments from the subsystem object class. The *Argument* string specified is a maximum of 1200 characters or the command is unsuccessful. The command argument is passed by the SRC to the subsystem, according to the same rules used by the shell. Quoted strings are passed as a single argument, and blanks outside a quoted string delimit an argument. Single and double quotes can be used.
-e *Environment*	Specifies an environment string that is placed in the subsystem environment when the subsystem is executed. The *Environment* string specified is a maximum of 1200 characters, or the command is unsuccessful. Using the same rules that are used by the shell, the SRC sets up the environment for the subsystem. Quoted strings are assigned to a single environment variable and blanks outside quoted strings delimit each environment variable to be set. For example: `-e "HOME=/tmp TERM=dumb MESSAGE=\"Multiple word message\""`would set `HOME=/tmp` as the first, `TERM=dumb` as the second, and `MESSAGE="Multiple word message"` as the third environment variable for the subsystem.
-g *Group*	Specifies a group of subsystems to be started. The command is unsuccessful if the *Group* name is not contained in the subsystem object class.
-h *Host*	Specifies the foreign host on which this start action is requested. The local user must be running as "root". The remote system must be configured to accept remote System Resource Controller requests. That is, the **srcmstr** daemon (see **/etc/inittab**) must be started with the **-r** flag and the **/etc/hosts.equiv** or **.rhosts** file must be configured to allow remote requests.
-o *Object*	Specifies that a subserver object is to be passed to the subsystem as a character string. It is the subsystems responsibility to determine the validity of the *Object* string.
-p *SubsystemPID*	Specifies a particular instance of the subsystem to which the start subserver request is to be passed.
-s S*ubsystem*	Specifies a subsystem to be started. The *Subsystem* can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the *Subsystem* is not contained in the subsystem object class.
-t *Type*	Specifies that a subserver is to be started. The command is unsuccessful if *Type* is not contained in the subserver object class.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To start a subsystem with arguments and environment variables, enter:

```
startsrc  -s srctest  -a "-D DEBUG"  -e "TERM=dumb HOME=/tmp"
```

This starts the `srctest` subsystem with "TERM=dumb", "HOME=/tmp" in its environment and "-D DEBUG" as two arguments to the subsystem.

2. To start a subsystem group on a foreign host, enter:

   ```
   startsrc  -g tcpip  -h zork
   ```

   This starts all the subsystems in the subsystem `tcpip` group on the `zork` machine.

3. To start a subserver, enter:

   ```
   startsrc  -t tester
   ```

   This sends a start subserver request to the subsystem that owns the `tester` subsystem.

4. To start a subsystem with command arguments, enter:

   ```
   startsrc  -s srctest  -a "-a 123 -b \"4 5 6\""
   ```

   This places "-a" as the first argument, "123" as the second, "-b" as the third, and"456" as the fourth argument to the `srctest` subsystem.

Files

Item	Description
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration Object Class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration Object Class.
/etc/services	Defines the sockets and protocols used for Internet services.
/dev/SRC	Specifies the **AF_UNIX** socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.

startup Command

Purpose

Turns on accounting functions at system startup.

Syntax

/usr/sbin/acct/startup

Description

The **startup** command turns on the accounting functions when the system is started, if called by the **/etc/rc** command file. See the **startup** example for the command to add to the **/etc/rc** file.

Security

Access Control: This command should grant execute (x) access only to members of the adm group.

Examples

To turn on the accounting functions when the system is started, add the following to the **/etc/rc** file:

```
/usr/bin/su - adm -c /usr/sbin/acct/startup
```

The **startup** shell procedure will then record the time and clean up the previous day's records.

Files

Item	Description
/usr/sbin/acct	The path to the accounting commands.

startvsd Command

Purpose

startvsd – Makes a virtual shared disk available and activates it.

Syntax

startvsd [-p | **-b**] {**-a** | *vsd_name* ...}

Description

The **startvsd** command makes the specified virtual shared disks available and activates them. It is equivalent to running the **preparevsd** command followed by the **resumevsd** command on the specified virtual shared disk.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Start a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Flags

-p
Specifies that the primary server node defined for the global volume group is to be the active server.

See the *RSCT: Managing Shared Disks* for more information.

-b
Specifies that the secondary server node defined for the global volume group is to be the active server.

-a
Specifies that all virtual shared disks that have been defined are to be started.

Parameters

vsd_name
Specifies a virtual shared disk.

Security

You must have root authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Standard Output

Current RVSD subsystem run level.

Examples

To make available and activate the virtual shared disk **vsd1vg1n1**, enter:

```
startvsd vsd1vg1n1
```

Location

/opt/rsct/vsd/bin/startvsd

Related Information

Commands: **cfgvsd**, **lsvsd**, **preparevsd**, **resumevsd**, **stopvsd**, **suspendvsd**, **ucfgvsd**

startwpar Command

Purpose

Activates a workload partition.

Syntax

/usr/sbin/startwpar [**-a**] [`-m`] [`-v`] [**-1** [**-R**] | **-2** [**-e***VAR=values ...*] | | **-I**] *WparName*

Description

The `startwpar` command activates a workload partition that is defined by the `mkwpar` command. It includes:

- Exporting devices from the global environment into the workload partition
- Mounting the workload partition file systems
- Assigning and activating the workload partition IP addresses
- Activating the workload partition WLM class, if any
- Creating the `init` command
-

The `startwpar` command fails if no workload partition exists with the given name.

Flags

Item	Description
-1	Phase 1: the loaded state. Specifies the **startwpar** command to stop before creating or running any process. Only programmatic consumers (consumers with administrative lock) can use this **-1** flag.
-2	Phase 2: starts initial processes. If the workload partition is already configured with the **startwpar -1** option, use the **-2** flag to complete the startup of the workload partition by spawning the registered application (application workload partitions), init (system workload partitions), or the registered **alternate init** if the workload partition was created with the **-c** (checkpointable) option of the **mkwpar** or **wparexec** commands. The operation context is identical to that of the normal **startwpar** operation for the type of workload partition that is queried. This option is in contrast with the **-I** option, whereby the **startwpar** process is replaced by the workload partition process. Only programmatic consumers can use this **-2** flag.
-a	Automatically resolves conflicting static settings if they occurred. Resolvable settings include hostname and network configuration.
-e *VAR=values ...*	Allows customization of the environment available to the initial process created by the **startwpar -2** flag. The parameter should be a single argument (appropriately quoted and escaped) in the form of *VAR=value ...* Only programmatic consumers can use this **-e** flag.
-I	Specifies that the **startwpar** command to **exec** the initial process for the workload partition: **/usr/lib/wpars/wparinit** for system workload partitions, and **/usr/lib/wpars/vinit** for application workload partitions. The **alternate init** command, if registered, is never run through this flag. The process is created and run through **exec**, and replaces the **startwpar** process. This is in contrast to the **-2** flag, whereby the initial process is run in its usual context. Only programmatic consumers can use this **-I** flag.
`-m`	Specifies that the workload partition should be started in maintenance mode. Networks that are associated with the workload partitionworkload partition are not configured, so the only access to the workload partition is from the global system. Do not use the **-m** flag to configure workload partitions with NFS file systems.
-R	When used with the **-1** flag, the **-R** flag specifies that the workload partition is to be configured for restart rather than fresh start. Only programmatic consumers can use this **-R** flag.
`-v`	Specifies to show verbose output.

Parameters

Item	Description
VAR=values ...	Values that can be interpreted into the **-e** flag as a single argument by the shell. It can contain punctuations, spaces and so on.
WparName	The name of the workload partition to be started.

Security

Access Control: Only the root user can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To start the workload partition called *roy*, enter:

```
startwpar roy
```

startx Command

Purpose

Initializes an X session.

Syntax

startx [**-d** *Display***:***0*] [**-t** | **-w**] [**-x** Startup | [**-r** *Resources*] [**-m** *Window_Manager*]] [**-wait**]

Description

The **startx** command streamlines the process of starting an X session.

The command does the following:

- Sets the user's **DISPLAY** environment variable to identify the X server to the X clients
- When run from a workstation, starts the X server
- Starts the X clients.

The **startx** command redirects X server and X client error messages to the file specified by the user's **XERRORS** environment variable. This process is useful for debugging and gives the X server a clean startup and shutdown appearance on a workstation.

If a startup script file name is not given at the command line with the **-x** option, then the **startx** command searches for a file specified by the user's **XINITRC** environment variable. If the **XINITRC** environment variable is not set, then the **startx** command searches the user's home directory for a file called **.Xinit**, **.xinit**, **.Xinitrc**, **.xinitrc**, or **.xsession**, respectively, to begin the X client programs.

If a startup file is not found, the **startx** command runs the Window Manager indicated at the command line with the **-m** option, or invokes the window manager **mwm**, **twm**, **awm**, or **uwm** after finding the associated configuration file (**.mwmrc**, **.twmrc** , **.awmrc**, or **.uwmrc**, respectively). If a window manager configuration file is not found in the user's home directory, **startx** initiates an **Xterm** client and the **mwm** window manager.

When a startup file is not found, the **startx** command also instructs the loading of the resources file given at the command line with the **-r** option, or a file from the user's home directory called **.Xdefaults**, **.xdefaults**, **.Xresources**, or **.xresources**, respectively. If an X resources file is not found, then the X session will not be personalized.

If a startup file exists for a workstation and no resources are loaded by the user, then the **xinit** command within the **startx** command attempts to load an **.Xdefaults** file.

The use of a workstation is assumed when the X session is initiated from **/dev/lft***. If this is not the case, then the **-t** or **-w** option must be used.

Flags

Item	Description
-d *Display***:***0*	Specifies the display name of the X server to pass to the X clients during the process for startup.
-m *Window_Manager*	Starts the Window Manager when no startup script is found.
-r *Resources*	Loads the resources file when no startup script is found.
-t	Starts X clients for an X terminal.
-w	Starts the X server and X clients for an X window session on a workstation.
-wait	Prevents the X session from being restarted when the **xdm** command invokes **startx**.
-x Startup	Starts an X window session using the startup script.

Note: You can use one or both of the **-m** and **-r** options, or the **-x** option, but you cannot use the **-x** option with the **-m** and **-r** options. In the startup script, it is the responsibility of the user to start a window manager session, load X resources, and spawn X clients.

Examples

1. To start an X session on a workstation, or an X terminal, enter:

```
startx
```

2. To force start an X session on a workstation, enter:

```
startx -w
```

3. To start an X session for an X terminal, and log off the user's telnet session, enter:

```
startx; kill -9 $$
```

4. To start an X session using the **.xinitrc** script, enter:

```
startx -x .xinitrc
```

5. To start an X session using the **mwm** window manager, enter:

```
startx -m mwm
```

   However, if a startup script file is found, the **-w** option is ignored.

6. In the startup script, it is the responsibility of the user to start a window manager, load X resources, and spawn X clients. The following is an example of an **.xsession** script.

```
#!/bin/csh
 (mwm &)
 xrdb -load .Xdefaults
 (xclock -g 75x75+0+0 &)
 (xbiff -g 75x75+101-0 &)
 if ("/dev/lft*" == "`tty`") then
    aixterm -g 80x24+0+0 +ut -C -T `hostname`
 else
    aixterm -g 80x24+0+0 +ut -T `hostname`
 endif
```

For a workstation, the last line in the startup script should be a foreground **aixterm** command with the **-C** option for console messages.

For an X terminal, the last line in the startup script should be a foreground **aixterm** command without the **-C** option. In addition, because some X terminals do not terminate the **telnet** session upon closing, the user must exit the current telnet session before using hot keys to switch to the X session.

Also, the **startx** command can be used by the **xdm** command in the **/usr/lib/X11/xdm/Xsession** file. This provides the **xdm** command with the features of the **startx** command.

Files

The following file names have been historically used for the startup of an X session.

Item	Description
$HOME/.xerrors	Where **startx** is to redirect error messages. By default, **startx** redirects errors to the **.xerrors** file in user's home directory.
$HOME/.Xinit,	
$HOME/.xinit,	
$HOME/.Xinitrc,	
$HOME/.xinitrc,	
$HOME/.xsession	Used as a Startup file containing shell commands to start a window manager, load X resources, and spawn X clients.
$HOME/.Xdefaults,	
$HOME/.xresources	Used as an X resources file loaded to set user preferences for X clients.
$HOME/.mwmrc	An **mwm** configuration file.
$HOME/.twmrc	A **twm** configuration file.
$HOME/.awmrc	An **awm** configuration file.
$HOME/.uwmrc	A **uwm** configuration file.
/dev/lft*	The terminal, or tty, interface of a workstation's initial **login** shell.

statd Daemon

Purpose

Provides crash and recovery functions for the locking services on NFS.

Syntax

/usr/sbin/rpc.statd [**-d** *DebugLevel*] [**-D**] [**-t** *threads*]

Description

The **statd** daemon interacts with the **lockd** daemon to provide crash and recovery functions for the locking services on Network File System (NFS). The **statd** daemon must always be started before the **lockd** daemon.

The **statd** daemon is started and stopped by the following SRC commands:

```
startsrc -s rpc.statd
```

```
stopsrc -s rpc.statd
```

The status monitor maintains information on the location of connections as well as the status in the **/var/statmon/sm** directory, the **/var/statmon/sm.bak** directory, and the **/var/statmon/state** file. When

restarted, the **statd** daemon queries these files and tries to reestablish the connection it had prior to termination. To restart the **statd** daemon, and subsequently the **lockd** daemon, without prior knowledge of existing locks or status, delete these files before restarting the **statd** daemon.

Flags

Item	Description
-t *threads*	Specifies the maximum number of rpc.statd threads allowed. The Default value is 50.
-d *DebugLevel*	Specifies the debug level of rpc.statd. The debug level is disabled by default.
-D	Specifies which **statmon** directory to use. Without the **-D** flag, rpc.statd will use the **/var/statmon** directory. With the **-D** flag, rpc.statd will use the **statmon** directory under the current directory. The **-D** flag is disabled by default. **Note:** When statd is started manually with the **startsrc** command and using the **-D** flag, the current work directory (CWD) is used for srcmstr. Being srcmstr executed at boot for root and if anv $HOME for root is different from /, eg /root then statmon data will go into /root/statmon directory.

statvsd Command

Purpose

Displays virtual shared disk device driver statistics of a node.

Syntax

statvsd

Description

The `statvsd` command displays virtual shared disk statistics of a node. For example, on a busy server an increasing number of "requests queued waiting for a buddy buffer" is normal and does not necessarily imply a problem. Of more value is the "average buddy buffer wait_queue size" which is the number of requests queued for a buddy buffer when the `statvsd` command was issued. See the "Examples" section for the meaning of output lines.

Flags

None.

Parameters

None.

Security

You must be in the AIX **bin** group to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

The following examples display virtual shared disk device driver statistics.

1. The header line indicates the version and release of the code. For example:

```
VSD driver (vsdd): IP/SMP Version:4 Release:1
```

2. The level of virtual shared disk parallelism defaults to 9 and is the buf_cnt parameter on the uphysio call that the device driver makes in the kernel. For example:

```
9 vsd parallelism
```

3. The maximum IP message size in bytes. For example:

```
61440 vsd max IP message size
```

4. The number of requests that had to wait for a request block. For example:

```
61440 vsd max IP message size
```

5. The number of requests that had to wait for a pbuf (a buffer used for the actual physical I/O request submitted to the disk). For example:

```
0 requests queued waiting for a pbuf
```

6. The number of requests that had to wait for a buddy buffer. A buffer that is used on a server to temporarily store date for I/O operations originating at a client node. For example:

```
2689 requests queued waiting for a buddy buffer
```

7. The number of requests queued for a buddy buffer when the `statvsd` command was issued. For example:

```
0 average buddy buffer wait_queue size
```

8. The number of requests that a server has rejected, typically because of an out-of-range sequence number or an internal problem. For example:

```
4 rejected requests
```

9. The number of responses that a client has rejected. Typically because a response arrived after a retry was already sent to the server. For example:

```
0 rejected responses
```

10. The number of requests that were placed on the rework queue. For example:

```
0 requests rework
```

11. The number of read requests that were not on a 64 byte boundary. For example:

```
0 64 byte unaligned reads
```

12. The number of requests that got a DMA shortage. This condition would require the I/O operation to be executed in nonzero copy mode. For example:

```
0 DMA space shortage
```

13. The number of requests that have timed out. The current timeout period is approximately 15 minutes. For example:

```
0 timeouts
```

14. There are a fixed number of retries. The retries counters display the number of requests that have been retried for that particular "retry bucket." Numbers appearing further to the right represent requests that have required more retries. When a request exhausts its number of retries, it gets recorded as a timeout. For example:

```
retries: 0 0 0 0 0 0 0 0 0

          0 total retries
```

15. Sequence numbers are internally used by the device driver. These numbers are managed by the device driver and the Recoverable virtual shared disk subsystem. For example:

```
Non-zero Sequence Numbers

node#     expected     outgoing     outcase?     Incarnation:0
 11        125092         0            |

 11 Nodes Up with zero sequence numbers: 1 3 5 7 9 11 12 13 14 15 16
```

Location

/opt/rsct/vsd/bin/statvsd

stop-secldapclntd Command

Purpose

The **stop-secldapclntd** script is used to terminate the **secldapclntd** LDAP client daemon.

Syntax

/usr/sbin/stop-secldapclntd

Description

The **stop-secldapclntd** script terminates the running **secldapclntd** daemon process. It returns an error if the **secldapclntd** daemon is not running.

Security

A user with the **aix.security.ldap** authorization is authorized to use this command.

Example

To stop the running **secldapclntd** daemon process, type:

```
/usr/sbin/stop-secldapclntd
```

Files

Item	Description
/usr/sbin/stop-secldapclntd	Used to terminate the **secldapclntd** LDAP client daemon.

stopcondresp Command

Purpose

Stops the monitoring of a condition that has one or more linked responses.

Syntax

To stop monitoring a condition:

`stopcondresp` [`-q`] [`-h`] [`-TV`] *condition*[`:`*node_name*] [*response* [*response*...]]

To unlock or lock the condition/response association:

`stopcondresp` {`-U` | `-L`} [`-h`] [`-TV`] *condition*[`:`*node_name*] *response*

Description

The `stopcondresp` command stops the monitoring of a condition that has one or more linked responses. If no response is specified, all of the linked responses for the condition are stopped. If one or more responses is specified, only those responses that are linked to the condition are stopped. When the condition occurs, the response is not run. If no responses are active for a condition, the condition is no longer monitored.

If a particular condition/response association is needed for system software to work properly, it may be locked. A locked condition/response association cannot be stopped by the `stopcondresp` command. If the condition/response link you specify on the `stopcondresp` command is locked, it will not be stopped; instead an error will be generated informing you that the condition/response association is locked. To unlock a condition/response association, you can use the `-U` flag. A condition/response association is typically locked because it is essential for system software to work properly, so you should exercise caution before unlocking it.

Flags

-q
Does not return an error when either *condition* or *response* does not exist or when the *condition* linked with *response* is not being monitored.

-h
Writes the command's usage statement to standard output.

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

-U
Unlocks a condition/response association so it can be started, stopped, or removed. If a condition/response association is locked, this is typically because it is essential for system software to work properly. For this reason, you should exercise caution before unlocking it. When unlocking a condition/response association using the `-U` flag, no other operation can be performed by this command.

-L
Locks a condition/response association so it cannot be started, stopped, or removed. When locking a condition/response association using the -L flag, no other operation can be performed by this command.

Parameters

condition
Specifies the name of the condition linked to the response. The condition is always specified first.

node_name
Specifies the node in the domain where the condition is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

response
Specifies the names of one or more responses. Monitoring is stopped for the specified responses. (If a specified response is not linked to the condition, it is ignored.)

Security

The user needs write permission for the `IBM.Association` resource class to run `stopcondresp`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the -h flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To stop monitoring for the condition "FileSystem space used " which has the response "Broadcast event on-shift" linked with it, run this command:

```
stopcondresp "FileSystem space used" "Broadcast event on-shift"
```

2. To stop monitoring for the condition "FileSystem space used " using all of its linked responses, run this command:

```
stopcondresp "FileSystem space used"
```

This example applies to management domains:

1. To stop monitoring for the condition "FileSystem space used " on the managed node nodeB which has the response "Broadcast event on-shift" linked with it, run this command on the management server:

```
stopcondresp "FileSystem space used:nodeB" "Broadcast event on-shift"
```

This example applies to peer domains:

1. To stop monitoring for the condition "FileSystem space used " on the node nodeA which has the response "Broadcast event on-shift" linked with it, run this command on any node in the domain:

```
stopcondresp "FileSystem space used:nodeA" "Broadcast event on-shift"
```

Location

/opt/rsct/bin/stopcondresp

stoprpdomain Command

Purpose

Takes an online peer domain offline.

Syntax

`stoprpdomain` [`-f`] [`-h`] [`-w` [-s *Seconds*]] [`-TV`] *peer_domain*

Description

The `stoprpdomain` command takes all of the nodes that are currently online in the peer domain offline. The peer domain definition is not removed from the nodes.

The command must be run on a node that is online in the peer domain. If the command is run on a node that is offline to the peer domain, no action is performed.

If a Cluster-Aware AIX (CAA) cluster is configured, no action is performed because a peer domain operation in a CAA environment exists and is online for the life of the CAA cluster.

The `-f` flag must be used to override a subsystems rejection of the request to take the peer domain offline. A subsystem may reject the request if a peer domain resource is busy, such as in the case of a shared disk. Specifying the `-f` flag in this situation indicates to the subsystems that the peer domain must be brought offline regardless of the resource state.

Flags

-f
Forces the subsystems to accept the stop request when it otherwise would not.

-h
Writes the command's usage statement to standard output.

-s
Specifies the wait time in seconds for the peer domain to be offline before the command completes when the **-s** flag is used with the **-w** flag . If the waiting time exceeds the number of seconds, the command returns, but the offline operation continues. The default value is 300 seconds (5 minutes). Use 0 to specify that the command must not return until the peer domain is offline (no timeout on waiting).

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

-w
Waits for the peer domain to be offline before the command completes. Use the **-s** flag to specify the waiting time in seconds.

Parameters

peer_domain
Specifies the name of the online peer domain that is to be brought offline.

Security

The user of the `stoprpdomain` command needs write permission for the `IBM.PeerDomain` resource class on each node that is defined to the peer domain. By default, `root` on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

6
The peer domain definition does not exist.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is online in the peer domain.

Implementation Specifics

This command is part of the **rsct.basic.rte** fileset for the AIX® operating system.

Standard Input

When the `-f "-"` or `-F "-"` flag is specified, this command reads one or more node names from standard input.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

In these examples, `nodeA` is one of the nodes defined and is online to `ApplDomain`.

1. To take ApplDomain offline, run this command on nodeA:

```
stoprpdomain ApplDomain
```

2. To take ApplDomain offline while making sure the stop request will not be rejected by any subsystem, run this command on nodeA:

```
stoprpdomain -f ApplDomain
```

Location

/opt/rsct/bin/stoprpdomain

stoprpnode Command

Purpose

Takes one or more nodes offline from a peer domain.

Syntax

`stoprpnode` [`-f`] [`-h`] [`-w` [-s *Seconds*]] [`-TV`] *node_name1* [*node_name2*...]

`stoprpnode -F` { *file_name* | `"-"` } [`-f`] [`-h`] [`-w` [-s *Seconds*]] [`-TV`]

Description

The `stoprpnode` command takes an online node offline from a peer domain. The peer domain is determined by the online peer domain where the command is run. The command must be run from a node that is online to the desired peer domain.

If a Cluster-Aware AIX (CAA) cluster is configured, no action is performed because a peer domain operation in a CAA environment exists and is online for the life of the CAA cluster.

The `-f` flag must be used to override a subsystem's rejection of the request to take a node offline. A subsystem may reject the request if a node resource is busy, such as in the case of a shared disk. Specifying the `-f` flag in this situation indicates to the subsystems that the node must be brought offline regardless of the resource state.

If this command is used to take more than one node offline by specifying more than one *node_name* parameter, and the node that this command is running on is in the list, it will be brought offline last.

Flags

-f
Forces the subsystems to accept the stop request when it otherwise would not.

-F { *file_name* | "–" }
Reads a list of node names from *file_name*. Each line of the file is scanned for one node name. The pound sign (#) indicates that the remainder of the line (or the entire line if the # is in column 1) is a comment.

Use `-F "-"` to specify STDIN as the input file.

-h
Writes the command's usage statement to standard output.

-s
Specifies the wait time in seconds for all of the specified nodes to be offline before the command completes when the **-s** flag is used with the **-w** flag. If the waiting time exceeds the number of seconds, the command returns, but the offline operation continues. The default value is 300 seconds

(5 minutes). Use 0 to specify that the command must not return until all of the specified nodes are offline (no timeout on waiting).

-T
Writes the command's trace messages to standard error. For your software service organization's use only.

-V
Writes the command's verbose messages to standard output.

-w
Waits for all of the specified nodes to be offline before the command completes. Use the **-s** flag to specify the waiting time in seconds.

Parameters

***node_name1* [*node_name2*...]**
Specifies the peer domain node names of the nodes that are to be brought offline from the peer domain. You must specify the node names in exactly the same format as they were specified with the `addrpnode` command or the `mkrpdomain` command. To list the peer domain node names, run the `lsrpnode` command.

Security

The user of the `stoprpnode` command needs write permission for the `IBM.PeerNode` resource class on each node that is to be stopped in the peer domain. By default, `root` on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Exit Status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with a command-line interface script.

3
An incorrect flag was entered on the command line.

4
An incorrect parameter was entered on the command line.

5
An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is online to the peer domain. The node to be brought offline must be reachable from the node on which the command is run.

Implementation Specifics

This command is part of the **rsct.basic.rte** fileset for the AIX® operating system.

Standard Input

When the `-F "-"` flag is specified, this command reads one or more node names from standard input.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

In these examples, nodeA and nodeB are online to `ApplDomain`.

1. To take nodeB offline, run this command on nodeA:

```
stoprpnode nodeB
```

2. To take nodeB offline and force the offline request, run this command on nodeA:

```
stoprpnode -f nodeB
```

Location

/opt/rsct/bin/stoprpnode

stoprsrc Command

Purpose

Stops a resource (that is, takes it offline).

Syntax

To stop one or more resources, using data entered on the command line:

`stoprsrc -s "`*selection_string*`" [ -N { `*node_file* `| "-" } ] [-h] [-TV]` *resource_class* [*arg=value...*]

`stoprsrc -r [-h] [-TV]` *resource_handle* [*arg=value...*]

To stop one or more resources using command arguments that are predefined in an input file:

`stoprsrc -f` *resource_data_input_file* `-s "`*selection_string*`" [ -N { `*node_file* `| "-" } ] [-h] [-TV]` *resource_class*

`stoprsrc -f` *resource_data_input_file* `-r [-h] [-TV]` *resource_handle*

To list the names and data types of the command arguments:

`stoprsrc -l [-h]` *resource_class*

Description

The `stoprsrc` command requests that the resource monitoring and control (RMC) subsystem take one or more resources offline. The request is performed by the appropriate resource manager.

To stop one or more resources, use the `-s` flag to take offline all of the resources that match the specified selection string.

Instead of specifying multiple node names in *selection_string*, you can use the `-N` *node_file* flag to indicate that the node names are in a file. Use `-N "-"` to read the node names from standard input.

To stop one specific resource, use the `-r` flag to specify the resource handle that represents that specific resource.

Use the `-l` flag to determine whether the specified resource class accepts any additional command arguments.

If Cluster Systems Management (CSM) is installed on your system, yYou can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM `nodegrp` command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

The successful completion of this command does not guarantee that the resource is offline, only that the resource manager successfully received the request to take this resource offline. Monitor the resource dynamic attribute `OpState` to determine when the resource is taken offline. Register an event for the resource, specifying the `OpState` attribute, to know when the resource is offline. Or, intermittently run the `lsrsrc` command until you see that the resource is offline (the value of `OpState` is 2). For example:

```
lsrsrc -s 'Name == "/filesys1"' -t IBM.FileSystem Name OpState
```

Parameters

resource_class
Specifies the name of the resource class that contains the resources that you want to take offline.

resource_handle
Specifies the resource handle that corresponds to the resource you want to take offline. Use the `lsrsrc` command to obtain a list of valid resource handles. The resource handle must be enclosed within double quotation marks, for example:

```
"0x4017 0x0001 0x00000000 0x0069684c 0x0d4715b0 0xe9635f69"
```

arg=value...
Specifies one or more pairs of command argument names and values.

arg
Specifies the argument name.

value
Specifies the value for this argument. The value datatype must match the definition of the argument datatype.

Command arguments are optional. If any *arg=value* pairs are entered, there should be one *arg=value* pair for each command argument defined for the offline function for the specified resource class.

Use `stoprsrc -l` to get a list of the command argument names and datatypes for the specific resource class.

Flags

-f *resource_data_input_file*
Specifies the name of the file that contains resource argument information. The contents of the file would look like this:

```
PersistentResourceArguments::
argument1 = value1
argument2 = value2
```

-l
Lists the command arguments and data types. Some resource managers accept additional arguments that are passed to the offline request. Use this flag to list any defined command arguments and the data types of the command argument values.

-N { *node_file* | "-" }
Specifies that node names are read from a file or from standard input. Use `-N` *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (#) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use `-N "-"` to read the node names from standard input.

The CT_MANAGEMENT_SCOPE environment variable determines the scope of the cluster. If CT_MANAGEMENT_SCOPE is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and CT_MANAGEMENT_SCOPE is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-s "*selection_string*"
Specifies the selection string. All selection strings must be enclosed within either double or single quotation marks. If the selection string contains double quotation marks, enclose the entire selection string in single quotation marks. For example:

```
-s 'Name == "testing"'

-s 'Name ?= "test"'
```

Only persistent attributes can be listed in a selection string.

-h
Writes the command usage statement to standard output.

-T
Writes the command trace messages to standard error. For your software service organization use only.

-V
Writes the command verbose messages (if there are any available) to standard output.

Environment variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0
Specifies *local* scope.

1
Specifies *local* scope.

2
Specifies *peer domain* scope.

3
Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Standard output

When the `-h` flag is specified, this command usage statement is written to standard output. When the `-V` flag is specified, this command verbose messages (if there are any available) are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0
The command ran successfully.

1
An error occurred with RMC.

2
An error occurred with the command-line interface (CLI) script.

3
An incorrect flag was specified on the command line.

4
An incorrect parameter was specified on the command line.

5
An error occurred with RMC that was based on incorrect command-line input.

6
No resources were found that match the specified selection string.

Security

You need write permission for the *resource_class* specified in `stoprsrc` to run `stoprsrc`. Permissions are specified in the access control list (ACL) file on the contacted system. See the *Administering RSCT* guide for information about the ACL file and how to modify it.

Implementation specifics

This command is part of the `rsct.core.rmc` fileset for the AIX operating system.

Location

/opt/rsct/bin/stoprsrc

Examples

Suppose that you have a peer domain called `foo` with three defined nodes: nodeA, nodeB, and nodeC. nodeA has two Ethernet cards: `ent0` and `ent1`.

1. Suppose nodeA is online and `ent0` (on nodeA) is also online. To take `ent0` offline on nodeA, run this command on nodeA:

   ```
   stoprsrc -s 'Name == "ent0"' IBM.EthernetDevice
   ```

2. Suppose nodeA and nodeB are online, `ent0` (on nodeA) is also online, and you are currently logged on to nodeB. To take `ent0` offline on nodeA, run this command on nodeB:

   ```
   stoprsrc -s 'NodeName == "A" AND Name == "ent0"' IBM.EthernetDevice
   ```

3. Suppose nodeA and nodeB are online and file system `/filesys1` is defined and mounted on nodeB. To take `/filesys1` offline on nodeB, run this command on nodeA:

   ```
   stoprsrc -s 'NodeName == "B" AND Name == "/filesys1"' IBM.FileSystem
   ```

4. Suppose the resource handle for `ent0` on nodeA is:

   ```
   0x406b 0x0001 0x00000000 0x0069564c 0x0dc1f272 0xb9de145e
   ```

   To take `ent0` offline on nodeA, run this command on nodeA:

   ```
   stoprsrc -r "0x406b 0x0001 0x00000000 0x0069564c 0x0dc1f272 0xb9de145e"
   ```

stopsrc Command

Purpose

Stops a subsystem, a group of subsystems, or a subserver.

Syntax

To Stop a Subsystem

stopsrc [**-h** *Host*] [**-f** | **-c**] { **-a** | **-g** *Group* | **-p** *SubsystemPID* | **-s** *Subsystem* }

To Stop a Subserver

stopsrc [**-h** *Host*] [**-f**] **-t** *Type* [**-p** *SubsystemPID*] [**-P** *SubserverPID* | **-o** *Object*]

Description

The **stopsrc** command sends a request to the System Resource Controller (SRC) to stop a subsystem, a group of subsystems, or all subsystems. The **stopsrc** command sends the System Resource Controller a subsystem request packet that is forwarded to the subsystem for a stop subserver request.

In the absence of the **-f** (stop force) flag, a normal stop action is assumed. A normal stop requests that a subsystem or subserver complete all current processing, release resources when all application activity has been completed, and then end. No new requests for work should be accepted by the subsystem.

A forced stop requests that a subsystem or subserver end quickly, releasing all resources, but not wait for application activity to complete.

A cancel action stops the subsystem after the subsystem's resources are released and after a grace period. This grace period is specified in the subsystem object class. The cancel stop is used only for subsystem stops and is always sent to the subsystem as the **SIGTERM** signal. The subsystem should catch this signal, perform subsystem clean up operations, and end. If the subsystem does not end within the wait time period, specified in the subsystem object class, the subsystem is sent a **SIGKILL** signal to ensure that the subsystem stops.

If the subsystem uses sockets or message queues for communication, a packet is constructed and sent to the subsystem. If the subsystem uses signals for communication, the subsystem is sent the appropriate signal from the subsystem object class.

Flags

Item	Description
-a	Specifies that all subsystems are to be stopped.
-c	Specifies that the stop request is a canceled stop request. For a cancel stop request, a **SIGTERM** signal is sent to the subsystem. After the wait time contained in the subsystem object class has passed, if the subsystem has not yet ended, the subsystem is sent a **SIGKILL** signal.
-f	Specifies a forced stop request.
-g *Group*	Specifies that a group of subservers is to be stopped. The command is unsuccessful if the *Group* name is not contained in the subsystem object class.
-h *Host*	Specifies the foreign *Host* machine on which this stop action is requested. The local user must be running as "root". The remote system must be configured to accept remote System Resource Controller requests. That is, the **srcmstr** daemon (see **/etc/inittab**) must be started with the **-r** flag and the **/etc/hosts.equiv** or **.rhosts** file must be configured to allow remote requests.
-o *Object*	Specifies that a subserver *Object* value is to be passed to the subsystem as a character string.
-p *SubsystemPID*	Specifies a particular instance of the subsystem to stop, or a particular instance of the subsystem to which the stop subserver request is to be passed.
-P *SubserverPID*	Specifies that a subserver PID is to be passed to the subsystem as a character string.
-s *Subsystem*	Specifies a subsystem to be stopped. The *Subsystem* parameter can be the actual subsystem name or the synonym name for the subsystem. The **stopsrc** command stops all currently active instances of the subsystem. The command is unsuccessful if the *Subsystem* name is not contained in the subsystem object class.
-t *Type*	Specifies that a subserver is to be stopped. The **stopsrc** command is unsuccessful if the *Type* specified is not contained in the subserver object class.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To stop force a subsystem on a foreign host, enter:

   ```
   stopsrc  -h zork  -s srctest  -f
   ```

   This forces a stop on all the instances of the `srctest` subsystem on the `zork` machine.
2. To stop cancel a subsystem group, enter:

   ```
   stopsrc  -g tcpip  -c
   ```

   This activates a stop cancel on all the subsystems in the `tcpip` group.
3. To stop a subserver, enter:

   ```
   stopsrc  -t tester  -p 1234
   ```

   This stops the `tester` subserver that belongs to the `srctest` subsystem with a subsystem PID of 1234.
4. To stop all subsystems, enter:

   ```
   stopsrc  -a
   ```

   This stops all the active subsystems on the local machine.

Files

Item	Description
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration Object Class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration Object Class.
/etc/services	Defines the sockets and protocols used for Internet services.
/dev/SRC	Specifies the **AF_UNIX** socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.

stopvsd Command

Purpose

stopvsd – Makes a virtual shared disk unavailable.

Syntax

`stopvsd` {`-a` | *vsd_name* ...}

Description

The **stopvsd** command brings the specified virtual shared disks from the suspended state to the stopped state. This makes the virtual shared disks unavailable. All applications that have outstanding requests for the virtual shared disk see these requests terminate with error. Read and write requests return errors with **errno** set to **ENODEV**. If the virtual shared disk is in the stopped state, this command leaves it in the stopped state.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Stop a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Flags

-a
Specifies that all virtual shared disks in the suspended state are to be stopped.

Parameters

vsd_name
Specifies a virtual shared disk. If the virtual shared disk is not is the suspended state, you get an error message.

Security

You must have root authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Standard Output

Current RVSD subsystem run level.

Examples

To bring the virtual shared disk **vsd1vg1n1** from the suspended state to the stopped state, enter:

```
stopvsd vsd1vg1n1
```

Location

/opt/rsct/vsd/bin/stopvsd

stopwpar Command

Purpose

Deactivates an active workload partition.

Syntax

/usr/sbin/stopwpar[-h | -F] [-r] [-t *seconds* | -N] [-v] *WparName*

Description

The `stopwpar` command deactivates a running workload partition. This includes stopping the following tasks:

- Stopping processes running within the workload partitions.
- Unloading the workload partition's WLM class, if any.
- Deactivating the workload partition's IP addresses, if any.
- Unmounting the workload partition's file systems, if any.
- Restarting the system workload partition.
- Removing the application workload partition.

The `stopwpar` command fails under the following circumstances:

- The specified workload partition does not exist.
- One or more processes cannot be stopped by the **kill** command (use the `-F` flag to force.)
- One or more file systems cannot be unmounted (use the `-F` flag to force.)

Flags

Item	Description
`-F`	Forces the workload partition to stop and signals the running processes more aggressively and unmounts the remote file systems. If the processes cannot be stopped, the workload partition remains in the `Broken` state and cannot be restarted.
`-h`	Uses a hard stop to signal the workload partition subsystems to end. The default timeout value is 60 seconds when using a hard stop.
`-N`	Specifies that the shutdown/halt to complete with no timeout.
`-r`	Restarts the workload partition after all stopping operations complete. This is equivalent to calling the `startwpar` command after the `stopwpar` command. This flag is not valid for application workload partitions.
`-t` *seconds*	Specifies the timeout length in number of seconds to wait for shutdown/halt to complete before the command fails and the program exits. The default is to fail after 600 seconds if the shutdown/halt has not completed.
`-v`	Specifies to show verbose output.

Parameters

Item	Description
WparName	Name of workload partition to stop. This parameter must be the last parameter on the command line.

Security

Access Control: Only the root user can run this command for system workload partitions. For application workload partitions, only the creator of the workload partition (or root) can run this command.

Examples

1. To stop the workload partition called *roy*, enter:

```
stopwpar roy
```

2. To discontinue the shutdown processing for the workload partition called *pinto* after 85 seconds, enter:

```
stopwpar -t 85 pinto
```

stpinet Method

Purpose

Disables the inet instance.

Syntax

stpinet [**-l "***Interface ...***"**] [**-t** *Time*]

Description

If **stpinet** is started with a list of network interfaces specified with the **-l** option, then this method only stops those IFs. Otherwise, **stpinet** informs users of the impending demise of TCP/IP, using the **wall** command, and invokes the **ifconfig** command to mark each configured IF as **down**. If no network interfaces are specified, the status flag of the inet instance is set to DEFINED.

Flags

Item	Description
-l "*Interface ...***"**	Specifies the name of the interface to be disabled.
-t *Time*	Specifies the time in minutes until the inet instance is stopped.

Examples

The following example disables the inet instance `tr0` five minutes from the time the method is executed:

```
stpinet -l "tr0" -t 5
```

strace Command

Purpose

Prints STREAMS trace messages.

Syntax

strace [*mid sid level*] ...

Description

The **strace** command without parameters writes all STREAMS event trace messages from all drivers and modules to its standard output. These messages are obtained from the STREAMS **log** driver. If parameters are provided, they must be in triplets. Each triplet indicates that tracing messages are to be received from

the given module or driver, subID (usually indicating minor device), and priority level equal to or less than the given level. The `all` token may be used for any member to indicate no restriction for that attribute.

Parameters

Item	Description
mid	Specifies a STREAMS module ID number.
sid	Specifies a subID number.
level	Specifies a tracing priority level.

Output Format

The format of each trace message output is:

```
<seq> <time> <ticks> <level> <flags> <mid> <sid> <text>
```

Item	Description
`<seq>`	Trace sequence number
`<time>`	Time of message in *hh:mm:ss*
`<ticks>`	Time of message, in machine ticks, since system was started
`<level>`	Tracing priority level
`<flags>`	Has one of the following values: **E** Message is also in the error log **F** Indicates a fatal error **N** Mail was sent to the system administrator
`<mid>`	Module ID number of source
`<sid>`	SubID number of source
`<text>`	Formatted text of the trace message On multiprocessor systems, `<text>` is composed of two parts: • the number of the processor where the owner of the message has sent it, • the formatted text itself.

Once initiated, the **strace** command continues to execute until terminated by the user.

> **Note:** Due to performance considerations, only one **strace** command is permitted to open the STREAMS log driver at a time. The log driver has a list of the triplets specified in the command invocation, and compares each potential trace message against this list to decide if it should be formatted and sent up to the **strace** process. Hence, long lists of triplets have a greater impact on overall STREAMS performance. Running the **strace** command has the most impact on the timing of the modules and drivers generating the trace messages that are sent to the **strace** process. If trace messages are generated faster than the **strace** process can handle them, some of the messages will be lost. This last case can be determined by examining the sequence numbers on the trace messages output.

Examples

1. To output all trace messages from the module or driver whose module ID is 41, enter:

```
strace 41 all all
```

2. To output those trace messages from driver or module ID 41 with sub-IDs 0, 1, or 2:

```
strace 41 0 1 41 1 1 41 2 0
```

   Messages from sub-IDs 0 and 1 must have a tracing level less than or equal to 1. Those from sub-ID 2 must have a tracing level of 0.

strchg Command

Purpose

Changes stream configuration.

Syntax

To push modules onto a stream:

strchg **-h** *Module1* [**,** *Module2* ...]

To pop modules off a stream:

strchg **-p** [**-a** | **-u** *Module*]

To push and pop modules to conform to the configuration file:

strchg **-f** *File*

Description

The **strchg** command is used to alter the configuration of the stream associated with the user's standard input. The **strchg** command **pushes modules** on the stream, pops modules off of the stream, or both. Only the root user or owner of a STREAMS device can alter the configuration of that stream. If another user attempts to alter the configuration, the **strchg** command will not succeed.

> **Note:** If modules are pushed in the wrong order, the stream might not function as expected.

Flags

Item	Description
-a	Pops all modules above the topmost driver off of a stream. The **-p** flag must be used in front of the **-a** flag.
-f *File*	Pushes and pops the necessary modules to conform the stream to the configuration given in the specified file. The **-h**, **-p**, and **-f** flags are mutually exclusive.
-h *Module1*	Pushes modules onto a stream. The modules are listed on the command line in the order they are to be pushed.
-p	Pops a module off of a stream. Used alone, the **-p** flag pops the topmost module from the stream.
-u *Module*	Pops all modules above the specified module off of a stream. The **-p** flag must be used in front of the **-u** flag. The **-a** and **-u** flags are mutually exclusive.

Parameters

Item	Description
Module1	Specifies the module to be pushed onto a stream. (Used by the **-h** flag.)
Module	Specifies the topmost module to remain on a stream. All modules above this module are popped off of the stream. (Used by the **-u** flag.)
File	Contains a list of modules representing the desired configuration of the stream. Each module name must appear on a separate line, where the first name represents the topmost module and the last name represents the module that is closest to the driver.

Return Values

On successful completion, the **strchg** command returns a value of 0. Otherwise, it returns a nonzero value and prints an error message indicating usage error, a bad module name, too many modules to push, failure of an **ioctl** operation on the stream, or failure to open the file specified by the *File* parameter.

Examples

1. To push the `ldterm` module on the stream, enter:

   ```
   strchg -h ldterm
   ```

2. To pop the topmost module from the stream associated with the `/dev/term/24` device, enter:

   ```
   strchg -p < /dev/term/24
   ```

   The user must be the owner of this device or the root user.

3. If the `fileconf` file contains the following:

   ```
   compat
   ldterm
   ptem
   ```

   the following command configures the stream so that the `ptem` module is pushed over the driver, followed by the `ldterm` module, and the `compat` module is pushed closest to the stream head.

   ```
   strchg -f fileconf
   ```

strclean Command

Purpose

Cleans up the STREAMS error logger.

Syntax

strclean [**-d**] [**-a** *Age*]

Description

The **strclean** command is used to clean up the STREAMS error-logger directory on a regular basis: for example, by using the **cron** daemon. By default, all files with names matching **error.*** in the **/var/adm/streams** directory that have not been modified in the last three days are removed.

Note: The **strclean** command is typically run using the **cron** deamon on a daily or weekly basis.

Flags

Item	Description
-a *Age*	Specifies the maximum age, in days, for a log file.
-d	Specifies a directory other than the default directory.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

The following example has the same result as running the **strclean** command with no parameters.

```
strclean -d /var/adm/streams -a 3
```

Files

Item	Description
/var/adm/streams/error.*	Contains the STREAMS error log.

strconf Command

Purpose

Queries stream configuration.

Syntax

strconf [**-t** | **-m** *module*]

Description

The **strconf** command is used to query the configuration of a stream. When used without any flags, it prints a list of all the modules in the stream as well as the topmost driver. The list is printed with one name per line, where the first name printed is the topmost module on the stream and the last item printed is the name of the driver.

Note: The **strconf** command only reads from standard input.

Flags

Item	Description
-m *Module*	Determines if the specified module is present on the stream. If the module is present, the **strconf** command prints the message yes and returns a value of 0. If it is not present, the **strconf** command prints the message no and returns a nonzero value. The **-t** and **-m** flags are mutually exclusive.
-t	Prints only the topmost module of the stream (if one exists).

Parameter

Item	Description
Module	Specifies the module for which to look.

Examples

1. For a stream that has only the `ldterm` module pushed above the `ports` driver, the **strconf** command (with no flags) would produce the following output:

   ```
   ldterm
   ports
   ```

2. Entering the following command asks if the `ldterm` module is on the stream:

   ```
   strconf -m ldterm
   ```

   The command produces the following output while returning an exit status of 0:

   ```
   yes
   ```

strerr Daemon

Purpose

Receives error log messages from the STREAMS **log driver** .

Syntax

strerr

Description

The **strerr** daemon receives error log messages from the STREAMS log driver and appends them to a log file. The error log files produced reside in the directory **/var/adm/streams**, and are named **error.***mm-dd*, where *mm* is the month and *dd* is the day of the messages contained in each log file.

The format of an error log message is:

```
<seq> <time> <ticks> <flags> <mid> <sud> <text>
```

These fields are defined as follows:

Item	Description
`<seq>`	Error sequence number
`<time>`	Time of message in *hh:mm:ss*
`<ticks>`	Time of message in machine ticks since boot priority level
`<flags>`	Has one of the following values: **T** The message was also sent to a tracing process **F** Indicates a fatal error **N** Send mail to the person who administers your system

Item	Description
<mid>	Module ID number of source
<sid>	Sub-ID number of source
<text>	Formatted text of the error message On multiprocessor systems, <text> is composed of two parts: • the number of the processor where the owner of the message has sent it, • the formatted text itself.

Messages that appear in the error log are intended to report exceptional conditions that require the attention of the person who administers your system. Those messages indicating the total failure of a **STREAMS driver or module** should have the **F** flag set. Those messages requiring the immediate attention of the administrator should have the **N** flag set, which causes the error logger to send the message to that person by way of the **mail** command. The priority level usually has no meaning in the error log, but does have meaning if the message is also sent to a tracer process.

Once initiated, the **strerr** daemon continues to execute until terminated by the user. Usually, the **strerr** daemon is executed asynchronously.

> **Note:** Only one **strerr** daemon at a time is permitted to open the STREAMS log driver. If a module or driver is generating a large number of error messages, running the error logger causes a degradation in STREAMS performance. If a large number of messages are generated in a short time, the log driver may not be able to deliver some of the messages. This situation is indicated by gaps in the sequence numbering of the messages in the log files.

Files

Item	Description
/var/adm/streams/error.*mm-dd*	Error log file.

strinfo Command

Purpose

Displays administrative information about STREAMS activity.

Syntax

strinfo **-m** | **-q**

Description

The **strinfo** command displays information for debugging purposes about STREAMS, drivers and modules, or stream heads and the STREAMS run queue.

Flags

Item	Description
-m	Displays information on drivers and modules present in STREAMS.
-q	Displays informations on active stream heads, and on the run queue which holds the STREAMS module and driver service procedures.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display information about STREAMS drivers and modules in use, enter:

   ```
   strinfo  -m
   ```

   This produces a listing similar to the following:

   ```
   Device: 'sad', dcookie 0xf, flags:0x4, str 0x19a69e8
   Device: 'slog', dcookie 0x10, flags:0x4, str 0x19a6c18
   Device: 'rs', dcookie 0x11, flags:0x2, str 0x19bcb00
   Module: 'bufcall', flags:0x1, str 0x19a5c00
   Module: 'ldterm', flags:0x0, str 0x19cc858
   ```

   In this example `dcookie` indicates the major number, `flags` indicates the flags configuration, and `str` is the STREAMS table address.
2. To display information about active stream heads and the STREAMS run queue, enter:

   ```
   strinfo  -q
   ```

   This produces a listing similar to the following:

   ```
   Active Stream Heads
   sth      sth_dev  sth_rq   sth_wq   sth_flag rq->q_first
   05a7ee00 00110001 05ad7000 05ad7074 00000818 00000000

   STREAMS Service Queue
   Queue 0x5ad7000 Flags 0x10
   ```

File

Item	Description
/usr/sbin/strinfo	Contains the **strinfo** command.

strings Command

Purpose

Finds the printable strings in a file.

Syntax

strings [**-a**] [**-**] [**-o**] [**-t** *Format*] [**-n** *Number*] [**-***Number*] [*File* ...]

Description

The **strings** command looks for printable strings in a file. A string is any sequence of 4 or more printable characters that end with a new-line or a null character. The **strings** command is useful for identifying random object files.

Flags

Item	Description
-a or **-**	Searches the entire file, not just the data section, for printable strings. If this flag is omitted, the **strings** command only looks in the initialized data space of object files.
-n *Number*	Specifies a minimum string length other than the default of 4 characters. The maximum value of a string length is 4096. This flag is identical to the **-***Number* flag.
-o	Lists each string preceded by its octal offset in the file. This flag is identical to the **-t o** flag.
-t *Format*	Lists each string preceded by its offset from the start of the file. The format is dependent on the character used as the *Format* variable. **d** Writes the offset in decimal. **o** Writes the offset in octal. **x** Writes the offset in hexadecimal. **Note:** When the **-o** and the **-t** *Format* flags are defined more than once on a command line, the last flag specified controls the behavior of the **strings** command.
-*Number*	Specifies a minimum string length other than the default of 4 characters. The maximum value of a string length is 4096. This flag is identical to the **-n** *Number* flag.
File	Binary or object file to be searched.

Exit Status

This command returns the following exit values:

Item	Description
0	Specifies that the command ran successfully.
>0	Specifies that an error occurred.

Examples

1. To search a file, enter:

```
strings strings
```

The **string** command displays:

```
@(#)56
1.17 com/cmd/scan/strings.c, cdmscan, bos320 5/7/92 10:21:20
Standard input
strings.cat
/usr/mbin/strings
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
%7o
%7d
%7x
%7o
%7d
```

2. To search for strings at least 12 characters long, enter:

```
strings -12 strings
```

The **string** command displays:

```
1.17 com/cmd/scan/strings.c, cdmscan, bos320 5/7/92 10:21:20
Standard input
/usr/mbin/strings
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
```

3. To search for strings at least 20 characters long and show the offset in hexadecimal, enter:

```
strings -t x -n 20 strings
```

The **string** command displays:

```
1017 1.17 com/cmd/scan/strings.c, cmdscan, bos320 5/7/92 10:21:20
108c Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
10d8 Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
1124 Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
1170 Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
11bc Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
```

strip Command

Purpose

Reduces the size of an Extended Common Object File Format (XCOFF) object file by removing information used by the binder and symbolic debug program.

Syntax

strip [**-V**] [**-r** [**-l**] | **-x** [**-l**] | **-t** | **-H** | **-e** | **-E**] [**-X** {**32**|**64**|**32_64**}] [**—**] *File* ...

Description

The **strip** command reduces the size of XCOFF object files. The **strip** command optionally removes the line number information, relocation information, the debug section, the typchk section, the comment section, file headers, and all or part of the symbol table from the XCOFF object files. Once you use this command, symbolic debugging of the file is difficult; therefore, you should normally use the **strip** command only on production modules that you have debugged and tested. Using the **strip** command reduces the storage overhead required by an object file.

For each object module, the **strip** command removes information as specified by the supplied options. For each archive file, the **strip** command removes the global symbol table from the archive.

You can restore a stripped symbol table to an archive or library file by using the **ar -s** command.

The **strip** command with no options removes the line number information, relocation information, symbol table, the debug section, and the typchk section, and the comment section.

Flags

Item	Description
-e	Sets the **F_LOADONLY** flag in the optional header of the object file. If the object file is placed in an archive, this flag indicates to the binder (**ld** command) that symbols in the object file should be ignored when linking with the archive.
-E	Resets (turns off) the **F_LOADONLY** bit in the optional header of the object file. (See **-e** flag).
-H	Removes the object file header, any optional header, and all section headers. **Note:** Symbol Table information is not removed.
-l	(Lowercase L) Strips the line number information from the object file.

Item	Description
-r	Removes all symbol table information except those entries for external and static symbols. Does not remove the relocation information. Also removes the debug and typchk sections. This option produces an object file that can still be used as input to the linkage editor (**ld** command).
-t	Removes most symbol table information but does not remove function symbols or line number information.
-V	Prints the version number of the **strip** command.
-x	Removes the symbol table information but does not remove static or external symbol information. The **-x** flag also removes relocation information, therefore linking to the file would not be possible.
-X *mode*	Specifies the type of object file **strip** should examine. The *mode* must be one of the following: **32** Processes only 32-bit object files **64** Processes only 64-bit object files **32_64** Processes both 32-bit and 64-bit object files The default is to process 32-bit object files (ignore 64-bit objects). The *mode* can also be set with the **OBJECT_MODE** environment variable. For example, **OBJECT_MODE=64** causes **strip** to process any 64-bit objects and ignore 32-bit objects. The **-X** flag overrides the **OBJECT_MODE** variable.
—	(Double hyphen) Interprets all arguments following this flag as file names. This allows you to strip files whose names start with a hyphen.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To remove the symbol table and line number information from the **a.out** file, enter:

```
strip a.out
```

2. To remove the object file header of the **a.out** file, enter:

```
strip -H a.out
```

3. To remove both the 32-bit and 64-bit symbol tables from **lib.a**, enter:

```
strip -X 32_64 lib.a
```

Files

Item	Description
/usr/ccs/bin/strip	Contains the **strip** command.

stripnm Command

Purpose

Displays the symbol information of a specified object file.

Syntax

stripnm [**-x** | **-d**] [**-s**] [**-z**] *File*

Description

The **stripnm** command (when run without the **-s** flag) prints the symbol table of a specified object file to standard output. The file specified by the *File* parameter can be a single object file or an archive library of object files. If the file specified by the *File* parameter is an archive, a listing for each object file in the archive is produced. If the symbol table has been stripped from the object file, the **stripnm** command extracts symbol names from the traceback tables (even if the **-s** flag is not specified) and the loader section of the object file(s). If the traceback tables do not exist, an error message is displayed.

Each symbol name is preceeded by its address and one character representing the symbol type (similar to **nm** output). When used with **-z**, the output format is the same as it was before AIX 5.2, that is each symbol name is followed by its address (a series of blanks if the address is undefined) and the type of class and section type. The address field can be displayed as a decimal (the default value with **-z**, or when **-d** is used) or hexadecimal (the default value without **-z**, or if the **-x** flag is used).

Source file names are also collected and reported by the **stripnm** command. All the symbols following a source file name line belongs to the same source file, until the next source file name line is encountered. For stripped files, the source file name is reported as being the object file name.

When run using the **-s** flag, the **stripnm** command ignores the symbol table if present and always extracts routine names from the traceback tables and the loader section of the object file(s).

When no symbol table is present or the **-s** flag is used, the **stripnm** command also searches for glue code and pointer glue information. Both are sequences of instructions found in the text section of the object file.

The glue code for 32 bit applications is composed of the following sequences of instructions:

```
8182xxxx #   lwz r12,xxxx(r12) (xxxx is the TOC entry index)
90410014 #   stw r2,14(r1)
800c0000 #   lwz r0,0(r12)
```

```
804c0004 #   lwz r2,4(r12)
7c0903a6 # mtctr r0
4e800420 # bctr
```

The loader section entry whose address matches the TOC entry pointed to by xxxx gives the function name for this sequence of glue code.

For 64 bit executables, the glue code sequences are as follows:

```
982xxxx #    ld r12,xxxx(r2) (xxxx is the TOC entry index)
8410028 #   std r2,28(r1)
80c0000 #    ld r0,0(r12)
84c0008 #    ld r2,8(r12
c0903a6 # mtctr r0
e800420 #  bctr
```

The pointer glue code for 32 bit applications is composed of the following sequence:

```
800b0000 #   lwz r0,0(r11)
90410014 #   stw r2,20(r1)
7c0903a6 # mtctr r0
804b0004 #   lwz r2,4(r11)
816b0008 #   lwz r11,8(r11)
4e80xx20 #  bctr
```

For 64bit executables, the pointer glue code sequence is as follows:

```
e80b0000 #    ld r0,0(r11)
f8410028 #   std r2,20(r1)
7c0903a6 # mtctr r0
e84b0008 #    ld r2,8(r11)
e96b0010 #    ld r11,16(r11)
4e80xx20 #  bctr
```

Pointer glue exists only in one copy and is always reported as symbol `._prtgl`.

The stripnm command searches the Text section from beginning to end for these sequences. If the command finds a sequence of instructions that matches, it is reported as glue code or pointer glue.

Source file symbols are generated artificially by **stripnm** for both glue code and pointer glue. For 32 bit executables, the source file is `glink.s` for all glue code entries, and `ptrgl.s`, for the pointer glue. For 64 bit executables, the source files are repectively `glink64.s` and `ptrgl_64.s`.

The **stripnm** command can also be used to search for symbol information in the **/unix** file. If the **/unix** file does not correspond to the currently running kernel, a warning message displays.

Flags

Item	Description
-d	Prints symbol address values in decimal format. This is the default with **-z**.
-s	Forces to ignore symbol table.
-x	Prints symbol address values in hexadecimal format. This is the default without **-z**.
-z	Uses the old format.

Examples

1. To list the symbols of the **a.out** object file, type:

```
stripnm a.out
```

2. To list the symbols address values, in decimal, from the **a.out** object file, type:

```
stripnm -d a.out
```

3. To list symbols from the object file from libc.a in the old format, but using hexadecimal addresses, type:

```
stripnm -xz libc.a
```

strload Command

Purpose

Loads and configures **Portable Streams Environment** (PSE).

Syntax

strload [**-u** | **-q**] [**-f** *File*] [**-d** *List*] [**-m** *List*]

Description

The **strload** command enables the system administrator to load and unload drivers and modules and to query the load status of PSE and its dependents.

By default, the **strload** command loads PSE according to the **/etc/pse.conf** file. The **-f** flag allows the administrator to use an alternate configuration file. The **-d** and **-m** flags are used to specify drivers and modules that are not present in the configuration files (such as when new drivers are being developed). The **-q** flag reports on the system load status (kernel existence) of the referenced drivers and modules.

Configuration File

The configuration file is a flat ASCII, line-oriented database. Comments are introduced by a # (pound sign), and continue until the end of the line. Blank lines are ignored. The form for each record is:

```
attributes filename [argument [node [minor ...] ] ]
```

Fields are separated by spaces, tabs, or both. A - (dash) can be specified as the field value, indicating that the default value is to be used. The fields are defined as follows:

Item	Description
`attributes`	Describes the extension to load. The acceptable values are: **d** Specifies a driver. **m** Specifies a module. **s** Creates the node as a standard (not cloned) device. **+** Specifies that the extension can be configured more than once. This value must be specified for all lines containing the extension file name.

Item	Description
filename	Specifies the object file containing the extension. If the command is issued with a "/" (slash) in the filename of the driver or module to be loaded, unloaded or queried, the **strload** command uses the value in the filename field explicitly. If there is no "/" in the filename entry, the **strload** command first looks for a copy of the driver or module in the current directory. If the driver or module is not in the current directory, **strload** looks for the driver or module in the **/usr/lib/drivers/pse** directory.

Note: It is recommended that the **strload** command be issued from the root directory (/). The **strload** command for load, unload, and query must always be issued from the same directory.

The kernel extension loader REQUIRES that the path names used be identical in load, unload and queries. This, coupled with the way the filename is determined by **strload**, could cause problems. Every byte in the path name used by the **strload** command must EXACTLY match every positionally corresponding byte in the path name used by the kernel extension loader because the kernel does a **strcmp()** on the filename when looking for matches. If the **strload** command is issued from a different directory to unload the module or driver, one of the following events occurs:

- If the **strload** command does not find a copy of the driver or module in the new current directory, **strload** attempts to unload the driver or module in the **/usr/lib/drivers/pse** directory. However, this path name may not be the same as the path name that the loader has logged for that driver or module. If the path name is not the same, the **strload** command fails.
- If the **strload** command finds another copy of the module or driver in the new current directory, then the path names are the same, and the loader correctly unloads the driver or module that was loaded. Thus, the **strload** command succeeds, but the results may not be as the user intended.

For example:

The following scenario (NOT recommended) causes "spx", also known as "A", to be unloaded. This is probably not the desired effect.

```
mkdir /tmp/foo /tmp/bar
cp /usr/lib/drivers/pse/spx /tmp/foo/A
cp /bin/ls /tmp/bar/A
cd /tmp/foo
strload -d A      # The loader knows the path and filename as
                  # "A" because "A" is found in the current
                  # directory
cd /tmp/bar
strload -q -d A   # Reports "yes" because there is "A" in the
                  # current directory. Note that the file "A"
                  # in /tmp/bar is NOT the same file "A" in
                  # /tmp/foo, but the loader does not care
                  # because it identifies the file by
                  # pathname.
strload -u -d A   # Unloads spx (also known as "A")!
```

The following is an error scenario:

```
mkdir /tmp/foo2 /tmp/bar2
cp /usr/lib/drivers/pse/spx /tmp/foo2/A
cd /tmp/foo2
strload -d A     # The loader knows the path and filename as
                 # "A"because "A" is found in the current
                 # directory.
cd /tmp/bar2
strload -q -d A  # Answers "no". There is no filename
                 # in /tmp/bar2 that matches "A", so strload
                 # prepends pathname "/usr/lib/drivers/pse" to
                 # "A". "/usr/lib/drivers/pse/A" is not found,
                 # so strload answers "no".
strload -u -d A  # Fails - "A" does not exist.
```

The following is an error scenario:

```
cd /usr/lib/drivers/pse
strload -d spx   # The loader knows the path and filename as
```

```
                 # "spx" because "spx" is found in the
                 # current directory.
cd /
strload -q -d spx # Answers "no". There is no filename in /
                  # that matches "spx", so strload prepends
                  # the pathname "/usr/lib/drivers/pse" to
                  # "spx". "/usr/lib/drivers/pse/spx" is found
                  # since it exists, so strload gives
                  # "/usr/lib/drivers/pse/spx" to the loader.
                  # The strcmp() fails since
                  # "/usr/lib/drivers/pse/spx" and "spx" do
                  # not match exactly.
strload -u -d spx # Fails - "spx" does not exist.
```

Item	Description
argument	Has no meaning for the **strload** command. This field is optional. It is passed to the extension when its configuration routine is called. Its interpretation is specific to that extension. The default argument is the value of the filename field.
node	Specifies the name of the node to create. This field is optional. It applies only to drivers and is used as the created node name when the driver is loaded. By default, the created node is **/dev/**filename.
minor	Specifies additional, non-clone nodes to create for this driver. This field is optional. The node names are created by appending the minor number to the cloned driver node name. No more than five minor numbers can be given (from 0 to 4), and a node is created for each one.

The **-d** and **-m** flags cause the configuration file to be ignored, unless it is explicitly named on the command line, as follows:

```
strload -f /tmp/my.conf -d newdriver
```

> **Note:** The **-d** and **-m** flags do not override the configuration file. That is, if driver **dgb** is loaded by using the configuration file, the **-d** flag will attempt to reload it but will fail. The configuration file is processed before the **-d** and **-m** flags.

The *List* variable for the **-d** and **-m** flags is a comma-separated list of file names, each of which contains a single PSE driver or module. The configuration process proceeds as if a line of one of the following forms was found in the configuration file:

```
d filename
```

```
m filename
```

Flags

Item	Description
-d *List*	Lists PSE device drivers to load or unload. The *List* variable specifies a comma-separated list of driver object names.
-f *File*	Configures PSE according to the configuration information contained in the file indicated by the *File* variable. The default configuration file is **/etc/pse.conf**.
-m *List*	Lists PSE modules to load or unload. The *List* variable specifies a comma-separated list of module object names.
-q	Reports load status of extensions.
-u	Unloads extensions.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. Entering the following command loads PSE (if not already loaded), the dgb and ssb drivers from the **/usr/lib/drivers/pse/** directory, and the aoot module from the current directory, but does not use the configuration file:

```
root# strload -d dgb,ssb -m ./aoot
```

2. To unload the aoot module only, enter:

```
root# strload -u -m ./aoot
```

3. Entering the following command asks if the spx driver exists:

```
root# strload -q -d
spx
```

   and produces the following output if not:

```
spx
: no
```

4. The following is an example configuration file:

```
#example configuration file
d       dgb                                     #line 1
d       mux    -       -       0                #line 2
ds      foo                                     #line 3
d+      xtiso  tcp     /dev/xti/tcp             #line 4
d+      xtiso  udp     /dev/xti/udp             #line 5
m       aoot                                    #line 6
```

   Line 1 loads the dgb driver extension as a cloned device named /dev/dgb. The argument passed to the dgb configuration routine is dgb.

   Line 2 loads the mux driver extension as a cloned device named /dev/mux and also creates a standard device name /dev/mux0 with a minor number of 0 (zero). (No more than five device names can be created with minor numbers from 0 to 4.)

   Line 3 loads the foo driver extension as a standard device (not cloned) named /dev/foo. The minor number is 0.

   Lines 4 and 5 load the xtiso driver extension, and configure it twice: once as tcp and once as udp. The clone nodes created are /dev/xti/tcp and /dev/xti/udp. The configuration routine of xtiso is called twice: once with the argument tcp, and once with udp.

   Line 6 loads the aoot module extension. No node is created, and the configuration routine is passed the value aoot.

5. To load the streams **dlpi** driver, enter:

```
strload -f /etc/dlpi.conf
```

Files

Item	Description
/usr/lib/drivers/pse/*	Contains PSE kernel extensions.

Item	Description
/etc/pse.conf	Default PSE configuration file.
/**usr**/**sbin**/**strload**	Contains the **strload** command.

strreset Command

Purpose

Resets a stream.

Syntax

strreset [**-M** *Major*] [**-m** *Minor*]

Description

The **strreset** command resets an open stream by generating an M_FLUSH message to the stream head. You use it mainly to reset blocked streams. When it is impossible to reopen the stream, issue an I_FLUSH ioctl(), or equivalent command. This situation may happen with a process sleeping in a module's close routine, when signals can not be sent to the process (a zombie process exiting, for example).

Flags

Item	Description
-M *Major*	Specifies the major number for the special file associated with the stream to be reset.
-m *Minor*	Specifies the minor number for the special file associated with the stream to be reset.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Auditing Events: N/A

Files

Item	Description
/usr/sbin/strreset	Contains the **strreset** command.

strtune Command

Purpose

This command has several related functions:

- Get or set the streams tunable parameters.
- Define the objects to trace using the component trace.
- List the tunable values of the stream modules.
- List the tunable values of the active queues.

Syntax

strtune {**-n** *name* **| -q** *addr*} **-o** *tunable_name*[=*value*] **-o** *tunable_name*[=*value*] ...

strtune [**-n** *name* **| -q** *addr* [**-a**]] **-o** *trclevel*[=*value*]

strtune [**-M**]

strtune [**-Q**]

strtune [**-f** *tunefile*]

Description

There are no restrictions on the use of this command when it is used to display or list values, but when using this command to modify tunable values or to define objects to trace, you must have root authority.

Flags

-n *name*
Defines a stream module name or a device name.

-q *addr*

Defines an active queue address.

If the command sets tunables, it modifies the queue pair, or the only queue, depending on the synchronization level of the queue. If the synchronization level is not SQLVL_QUEUE, the synchronization level is also propagated to all queue pairs.

-o *tunable_name*
Defines the name of the tunable parameter. Possible values are:

- **hiwat**, which defines the high water mark for the flow control on a queue
- **lowat**, which defines the low water mark for the flow control on a queue
- **minpsz**, which defines the minimum packet size
- **maxpsz**, which defines the maximum packet size. A value of -1 indicates an infinite packet size.

The **strtune** command can initialize several tunables by listing the **-o** option several times.

value
If no new value is given, the command displays the value of the tunable. Only a user with root authority can modify a tunable parameter value.

-n *name*
Defines a stream module name. If the **-n** or **-q** flag is not present in the command, the command will display or modify the global variable containing the pse global trace level (**pse_trclevel**).

-q *addr*

Defines an active queue address. If the **-n** or **-q** flag is not present in the command, the command will display or modify the global variable containing the pse global trace level (**pse_trclevel**).

If the command sets the trace level, it modifies the queue pair, or the only queue, depending on the synchronization level of the queue. If the synchronization level is not SQLVL_QUEUE, the synchronization level is also propagated to all queue pairs.

-o ***trclevel***
Displays or modifies the trace level. The **-o** flag cannot be listed more than once.

value
If no new value is given, the command displays the value of the tunable. Only a user with root authority can modify a tunable parameter value.

-a
Use this flag to force the strtune command to propagate the new value to all queues in the stream (from stream head to driver). If the synchronization level is not SQLVL_QUEUE, the synchronization level is also propagated to all queue pairs.

-M
Displays the name, idname, and associated tunable parameter (**minpsz**, **maxpsz**, **lowat**, **hiwat**, **trclevel**) values for each module.

-Q
Displays the name, idname, and associated tunable parameter (**minpsz**, **maxpsz**, **lowat**, **hiwat**, **trclevel**) values for each active queue.

-f ***tunefile***
The *tunefile* variable holds the filepath to the file that contains the tunable parameter settings. Each line of the *tunefile* file is managed as one command; if there are any modification commands in the tunefile, the user must have root authority for those modifications to be implemented.

Exit Status

0
Successful completion.

>0
An error occurred.

Examples

1. To display the **hiwat** tunable value of **ldterm** module:

```
strtune -n ldterm -o hiwat
```

2. With root authority, to set the value of **hiwat** for the **ldterm** module to 8192:

```
strtune -n ldterm -o hiwat=8192
```

3. To run the following lines:

```
-n udp6 -o lowat=256
-n dlpi -o hiwat=4096 -o lowat=128 -o minpsz=128
```

that are listed in the **/tmp/ff** file:

```
strtune -f /tmp/ff
```

This will result in the following commands being run:

```
strtune -n udp6 -o lowat=256
strtune -n dlpi -o hiwat=4096 -o lowat=128 -o minpsz=128
```

File

src/bos/usr/sbin/strtune/strtune.c
Contains the **strtune** command.

struct Command

Purpose

Translates a FORTRAN program into a RATFOR program.

Syntax

struct [**-s**] [**-i**] [**-a**] [**-b**] [**-n**] [**-t***Number*] [**-c***Number*] [**-e***Number*] [*File*]

Description

The **struct** command translates the FORTRAN program specified by *File* (standard input default) into a RATFOR program. Wherever possible, RATFOR control constructs replace the original FORTRAN. Statement numbers appear only where still necessary. Cosmetic changes are made, including changing Hollerith strings into quoted strings and relational operators into symbols (for example, **.GT.** into **>**). The output is appropriately indented.

The **struct** command knows FORTRAN 66 syntax, but not full FORTRAN 77. If an input FORTRAN program contains identifiers that are reserved words in RATFOR, the structured version of the program will not be a valid RATFOR program. The labels generated cannot go above 32767. If you get a **goto** statement without a target, try using the **-e** flag.

Flags

Item	Description
-a	Turn sequences of **else-if** statements into a non-RATFOR switch of the form: `switch` `{  case pred1: code` `  case pred2: code` `  case pred3: code` `  default: code` `  }` The **case** predicates are tested in order. The code appropriate to only one case is executed. This generalized form of **switch** statement does not occur in RATFOR.
-b	Generates **goto** statements instead of multilevel **break** statements.
-c *Number*	Increments successive labels in the output program by the nonzero integer *Number*. The default is 1. Do not insert a space between **-c** and *Number.*
-e *Number*	If *Number* is 0 (default), places code within a loop only if it can lead to an iteration of the loop. Do not insert a space between **-e** and *Number.*
-i	Do not turn computed **goto** statements into switches. (RATFOR does not turn switches back into computed **goto** statements.)
-n	Generates **goto** statements instead of multilevel **next** statements.
-s	Input is accepted in standard format. Comments are specified by a **c**, **C**, or ***** in column 1, and continuation lines are specified by a nonzero, nonblank character in column 6. Input is in the form accepted by the **f77** command.
-t *Number*	Makes the nonzero integer *Number* the lowest valued label in the output program. The default is 10. Do not insert a space between **-t** and *Number.*

If *Number* is nonzero, admits small code segments to a loop if otherwise the loop would have exits to several places including the segment, and the segment can be reached only from the loop. In this case, small is close to, but not equal to, the number of statements in the code segment. Values of *Number* under 10 are suggested.

Examples

To translate the `test.f` FORTRAN program into the `newtest.ratfor` RATFOR program, enter:

```
struct -s -i -n -t2 test.f > newtest.ratfor
```

Files

Item	Description
/tmp/struct*	Temporary files used during processing of the **struct** command.
/usr/lib/struct/structure	File that handles processing for the **struct** command.
/usr/lib/struct/beautify	File that handles processing for the **struct** command.
/usr/ucb/struct	Contains the **struct** command.

sttinet Method

Purpose

Enables the inet instance.

Syntax

sttinet [**-l** *Interface* ...]

Description

The **sttinet** method enables the inet instance by calling the **ifconfig** command and sets the status flag of the inet instance to AVAILABLE.

> **Note:** The **sttinet** method is a programming tool and should not be executed from the command line.

Flags

Item	Description
-l *Interface ...*	Specifies which specific interface to enable. If no interfaces are specified, then all configured interfaces are started.

Examples

The following method enables the inet instance:

```
sttinet -l tr0 -l tr1
```

stty-cxma Command

Purpose

Sets and reports the terminal options for a TTY configuration of the 128-port asynchronous subsystem.

Syntax

stty-cxma [**-a**] [**-g**] [*Option(s)*] [*ttyName*]

Description

If no flags or options are specified, the **stty-cxma** command reports all 128-port special driver settings and modem signals, as well as all standard parameters reported by the **stty** command for the tty device that is the current standard input.

The *ttyName* parameter can be specified to set or report options for a tty device for other than the standard input. The *ttyName* parameter can be a simple tty name, such as **tty0**, or can be prefixed by **/dev/**, such as **/dev/tty0**. This option may be used on a modem control line when no carrier is present.

Further options can be specified to change flow control settings, set transparent print options, force modem control lines, and display all tty settings. Unrecognized options are passed to the **stty** command for interpretation.

Flags

Item	Description
-a	Writes all the unique 128-port settings as well as all the standard tty settings reported by **stty -a** to standard output.

Item	Description
-g	Writes option settings to standard output in a form usable by another stty command.

Options

The following options specify transient actions to be performed immediately:

Item	Description
break	Sends a 250 MS break signal out on the tty line.
flush	Discards tty input and output immediately.
flushin	Discards tty input only.
flushout	Discards tty output only.

The actions specified by the following options are in effect until the device is closed. The next time the device is opened, default values are used.

Item	Description
dtr	Raises the DTR modem control line, unless DTR hardware flow control is selected.
-dtr	Drops the DTR modem control line, unless DTR hardware flow control is selected.
rts	Raises the RTS modem control line, unless RTS hardware flow control is selected.
-rts	Drops the RTS modem control line, unless RTS hardware flow control is selected.
startin	Releases flows control to resume stopped input.
startout	Restarts stopped output exactly as if an XON character was received.
stopin	Activates flow control to stop input.
stopout	Stops output exactly as if an XOFF character was received.

Item	Description
2200flow	Enables 2200 style flow control on the port. The 2200 terminals support an attached printer and use the following four flow control characters: **0xF8** terminal XON **0xF9** printer XON **0xFA** terminal XOFF **0xFB** printer XOFF
-2200flow	Disables 2200 style flow control on the port.
2200print	Runs flow control for the terminal and flow control for the transparent print device (as set by the **2200flow** option) independently.
-2200print	Runs terminal and printer flow control (as set by the **2200flow** option) together. So if either the terminal or the printer XOFF character is received, all output is paused until the matching XON character is received.
altpin	Switches the location of the DSR and DCD inputs on the modular connector, so that DCD is available when using an 8-pin RJ45 connector instead of the 10-pin RJ45 connector.
-altpin	Restores the availability of DSR when using the 10-pin RJ45 connector.
aixon	Enables auxiliary flow control, so that two unique characters are used for XON and XOFF. If both XOFF characters are received, transmission will not resume until both XON characters are received.
-aixon	Disables auxiliary flow control.
astartc *c*	Sets auxiliary XON flow control character. The character may be given as a decimal, octal, or hexadecimal number.
astopc *c*	Sets auxiliary XOFF flow control character. The character may be given as a decimal, octal, or hexadecimal number.
bufsize *n*	Sets the driver's estimate of the size of the transparent printer's input buffer. After a period of inactivity, the driver bursts this many characters to the transparent printer before reducing to the maximum CPS rate specified by the **maxcps** option rate selected above. The default value is 100 characters.
ctspace	Enables CTS hardware output flow control, so local transmission pauses when CTS drops.
-ctspace	Disables CTS hardware output flow control.
dcdpace	Enables DCD hardware output flow control, so local transmission pauses when DCD drops.
-dcdpace	Disables DCD hardware output flow control.
dsrpace	Enables DSR hardware output flow control, so local transmission pauses when DSR drops.
-dsrpace	Disables DSR hardware output flow control.
dtrpace	Enables DTR hardware input flow control, so DTR drops to pause remote transmission.
-dtrpace	Disables DTR hardware input flow control.

Item	Description
edelay *n*	Sets the rate at which the128-port asynchronous adapter wakes up the driver on input. The adapter wakes the driver every *n* milliseconds. The default value is 100 milliseconds.
fastbaud	Alters the baud rate table, so 50 baud becomes 57600 baud.
-fastbaud	Restores the baud rate table, so 57500 baud becomes 50 baud.

Item	Description
fastcook	Performs cooked output processing on the128-port asynchronous adapter to reduce host CPU usage and increase raw mode input performance.
-fastcook	Disables cooked output processing.
forcedcd	Disables carrier sense, so the tty may be opened and used even when the carrier is not present.
-forcedcd	Reenables carrier sense.
maxchar *n*	Sets the maximum number of transparent print characters the driver places in the output queue. Reducing this number increases system overhead; increasing this number delays operator keystroke echo times when the transparent printer is in use. The default value is 50 characters.
maxcps *n*	Sets the maximum CPS (characters per second) rate at which characters are output to the transparent print device. The rate chosen should be just below the average print speed. If the number is too low, printer speed is reduced. If the number is too high, the printer resorts to flow control, and user entry on the CRT is impaired accordingly. The default value is 100 CPS.
offstr *s*	Sets the CRT escape sequence to turn transparent print off. An arbitrary octal character *xxx* may be given as `\xxx`.
onstr *s*	Sets the CRT escape sequence to turn transparent print on. An arbitrary octal character *xxx* may be given as `\xxx`.
rtspace	Enables RTS hardware input flow control, so RTS drops to pause remote transmission.
-rtspace	Disables RTS hardware input flow control.
startc *c*	Sets the XON flow control character. The character may be given as a decimal, octal, or hexadecimal number.
stopc *c*	Sets the XOFF flow control character. The character may be given as a decimal, octal, or hexadecimal number.
term *t*	Sets the transparent printer on and off strings to values specified in the internal default table. Internal defaults are used for the following terminals:**adm31**, **ansi**, **dg200**, **dg210**, **hz1500**, **mc5**, **microterm**, **multiterm**, **pcterm**, **tvi**, **vp-a2**, **vp-60**, **vt52**, **vt100**, **vt220**, **wyse30**, **wyse50**, **wyse60**, or **wyse75**. If the terminal type is not found in the internal default table, the transparent print on and off strings are set to the values specified by the **po** and **pf** attributes in the **termcap** file.

Examples

1. To display all the unique 128-port settings as well as all the standard tty settings for a tty port configured on a 128-port asynchronous controller as `/dev/tty0`, enter:

```
stty-cxma -a tty0
```

2. To make DCD available when using an 8-pin RJ45 connector for a tty port configured on a 128-port asynchronous controller as `/dev/tty3`, enter:

```
stty-cxma altpin tty3
```

This command interchanges the location of the DSR and DCD inputs on the modular connector.

Files

Item	Description
/usr/ebin/tty/stty-cxma	Contains the **stty-cxma** command.

stty Command

Purpose

Sets, resets, and reports workstation operating parameters.

Syntax

stty [**-a**] [**-g**] [*Options*]

Description

The **stty** command sets certain I/O options for the device that is the current standard input. This command writes output to the device that is the current standard output.

This version of the operating system uses the standard X/Open Portability Guide Issue 4 interface to control the terminals, maintaining a compatibility with POSIX and BSD interfaces. The **stty** command supports both POSIX and BSD compliant options, but the usage of POSIX options is strongly recommended. A list of **obsolete BSD options**, with the corresponding POSIX options, is also provided.

When you redirect standard input from a tty device by typing:

```
stty -a </dev/ttyx
```

the **stty** command (POSIX) will hang while waiting for the **open()** of that tty until the RS-232 carrier detect signal has been asserted. Exceptions to this rule occur if the **clocal** or **forcedcd** (128-port only) option is set.

Flags

Item	Description
-a	Writes the current state of all option settings to standard output.
-g	Writes option settings to standard output in a form usable by another **stty** command.

Options

The **stty** command supports following categories of options:

- Control Modes
- Input Modes
- Output Modes
- Local Modes
- Hardware Flow Control Modes
- Control Character Assignments

- Combination Modes
- Window Size

Control Modes

Control Modes	Description
clocal	Assumes a line without modem control.
-clocal	Assumes a line with modem control.
cread	Enables the receiver.
-cread	Disables the receiver.
cstopb	Selects 2 stop bits per character.
-cstopb	Selects 1 stop bit per character.
cs5, **cs6**, **cs7**, **cs8**	Selects character size.
hup, **hupcl**	Hangs up dial-up connection on the last close.
-hup, **-hupcl**	Does not hang up dial-up connection on the last close.
parenb	Enables parity generation and detection.
-parenb	Disables parity generation and detection.
parodd	Selects odd parity.
-parodd	Selects even parity.
0	Hangs up phone line immediately.
speed	Sets the workstation input and output speeds to the specified *speed* number of bits per second. All speeds are not supported by all hardware interfaces. Possible values for *speed* are: **50**, **75**, **110**, **134**, **200**, **300**, **600**, **1200**, **1800**, **2400**, **4800**, **9600**, **19200**, **19.2**, **38400**, **38.4**, **exta**, and **extb**. **Note: exta**, **19200**, and **19.2** are synonyms; **extb**, **38400**, and **38.4** are synonyms.
ispeed *speed*	Sets the workstation input speed to the specified *speed* number of bits per second. All speeds are not supported by all hardware interfaces, and all hardware interfaces do not support this option. Possible values for *speed* are the same as for the *speed* option.
ospeed *speed*	Sets the workstation output speed to the specified *speed* number of bits per second. All speeds are not supported by all hardware interfaces, and all hardware interfaces do not support this option. Possible values for *speed* are the same as for the *speed* option.

Input Modes

Input Modes	Description
brkint	Signals INTR on break.
-brkint	Does not signal INTR on break.
icrnl	Maps CR to NL on input.
-icrnl	Does not map CR to NL on input.
ignbrk	Ignores BREAK on input.
-ignbrk	Does not ignore BREAK on input.

Input Modes	Description
igncr	Ignores CR on input.
-igncr	Does not ignore CR on input.
ignpar	Ignores parity errors.
-ignpar	Does not ignore parity errors.
inlcr	Maps NL to CR on input.
-inlcr	Does not map NL to CR on input.
inpck	Enables parity checking.
-inpck	Disables parity checking.
istrip	Strips input characters to 7 bits.
-istrip	Does not strip input characters to 7 bits.
iuclc	Maps uppercase alphabetic characters to lowercase.
-iuclc	Does not map uppercase alphabetic characters to lowercase.
ixany	Allows any character to restart output.
-ixany	Allows only the START (the Ctrl-Q key sequence) to restart output.
ixoff	Sends START/STOP characters when the input queue is nearly empty/ full.
-ixoff	Does not send START/STOP characters.
ixon	Enables START/STOP output control. Once START/STOP output control has been enabled, you can pause output to the workstation by pressing the Ctrl-S key sequence and resume output by pressing the Ctrl-Q key sequence.
-ixon	Disables START/STOP output control.
imaxbel	Echoes the BEL character and discards the last input character if input overflows.
-imaxbel	Discards all input if input overflows.
parmrk	Marks parity errors.
-parmrk	Does not mark parity errors.

Output Modes

Output Modes	Description
bs0, **bs1**	Selects style of delay for backspaces (**bs0** siginifes no delay).
cr0, **cr1**, **cr2**, **cr3**	Selects style of delay for CR characters (**cr0** siginifes no delay).
ff0, **ff1**	Selects style of delay for form feeds (**ff0** siginifes no delay).
nl0, **nl1**	Selects style of delay for NL characters (**nl0** siginifes no delay).
ofill	Uses fill characters for delays.
-ofill	Uses timing for delays.
ocrnl	Maps CR characters to NL characters.
-ocrnl	Does not map CR characters to NL characters.
olcuc	Maps lowercase alphabetic characters to uppercase on output.

Output Modes	Description
-olcuc	Does not map lowercase alphabetic characters to uppercase on output.
onlcr	Maps NL characters to CR-NL characters.
-onlcr	Does not map NL characters to CR-NL characters.
onlret	On the terminal, NL performs the CR function.
-onlret	On the terminal, NL does not perform the CR function.
onocr	Does not output CR characters at column zero.
-onocr	Outputs CR characters at column zero.
opost	Processes output.
-opost	Does not process output; that is, ignores all other output options.
ofdel	Uses DEL characters for fill characters.
-ofdel	Uses NUL characters for fill characters.
tab0, **tab1**, **tab2**	Selects style of delay for horizontal tabs (**tab0** siginifes no delay).
tab3	Expands tab character to variable number of spaces.
vt0, **vt1**	Selects style of delay for vertical tabs (**vt0** siginifes no delay).

Local Modes

Local Modes	Description
echo	Echoes every character typed.
-echo	Does not echo characters.
echoctl	Echoes control characters as ^X (Ctrl-X), where X is the character given by adding 100 octal to the code of the control character.
-echoctl	Does not echo control characters as ^X (Ctrl-X).
echoe	Echoes the ERASE character as the "backspace space backspace" string. **Note:** This mode does not keep track of column position, so you can get unexpected results when erasing such things as tabs and escape sequences.
-echoe	Does not echo the ERASE character, just backspace.
echok	Echoes a NL character after a KILL character.
-echok	Does not echo a NL character after a KILL character.
echoke	Echoes the KILL character by erasing each character on the output line.
-echoke	Just echoes the KILL character.
echonl	Echoes the NL character.
-echonl	Does not echo the NL character.
echoprt	Echoes erased characters backwards with / (slash) and \ (backslash).
-echoprt	Does not echo erased characters backwards with / (slash) and \ (backslash).
icanon	Enables canonical input (canonical input allows input-line editing with the ERASE and KILL characters). See the discussion about **canonical mode input** in Line Discipline Module (ldterm) in *Communications Programming Concepts*.

Local Modes	Description
-icanon	Disables canonical input.
iexten	Specifies that implementation-defined functions shall be recognized from the input data. Recognition of the following control characters requires **iexten** to be set: **eol2, dsusp, reprint, discard, werase, lnext.** The functions associated with these modes also require **iexten** to be set: **imaxbel**, **echoke**, **echoprt**, and **echoctl**.
-iexten	Specifies that implementation-defined functions shall not be recognized from the input data.
isig	Enables the checking of characters against the special control characters INTR, SUSP and QUIT.
-isig	Disables the checking of characters against the special control characters INTR, SUSP and QUIT.
noflsh	Does not clear buffers after INTR, SUSP, or QUIT control characters.
-noflsh	Clears buffers after INTR, SUSP, or QUIT control characters.
pending	Causes any input that is pending after a switch from raw to canonical mode to be re-input the next time a read operation becomes pending or the next time input arrives. Pending is an internal state bit.
-pending	No text is pending.
tostop	Signals **SIGTOU** for background output.
-tostop	Does not signal **SIGTOU** for background output.
xcase	Echoes uppercase characters on input, and displays uppercase characters on output with a preceding \ (backslash).
-xcase	Does not echo uppercase characters on input.

Hardware Flow Control Modes

These options are extensions to the X/Open Portability Guide Issue 4 standard.

Item	Description
cdxon	Enables CD hardware flow control mode on output.
-cdxon	Disables CD hardware flow control mode on output.
ctsxon	Enables CTS hardware flow control mode on output.
-ctsxon	Disables CTS hardware flow control mode on output.
dtrxoff	Enables DTR hardware flow control mode on input.
-dtrxoff	Disables DTR hardware flow control mode on input.
rtsxoff	Enables RTS hardware flow control mode on input.
-rtsxoff	Disables RTS hardware flow control mode on input.

Control Assignments

To assign a control character to a character string, type:

```
stty Control String
```

where the *Control* parameter may be the intr, quit, erase, kill, eof, eol, eol2, start, stop, susp, dsusp, reprint, discard, werase, lnext, min, or time character. (Use the min and time characters with the **-icanon** option.)

Note: The values for min and time are interpreted as integer values, not as character values.

The *String* parameter may be any single character such as c. An example of this control assignment is:

```
stty stop c
```

Another way of assigning control characters is to enter a character sequence composed of a \^ (backslash, caret) followed by a single character. If the single character after the ^ (caret) is one of the characters listed in the ^c (caret c) column of the following table, the corresponding control character value will be set. For example, to assign the DEL control character by using the ? (question mark) character, type the string \^? (backslash, caret, question mark), as in:

```
stty erase \^?
```

caret Control Characters in stty	
^c	**Value**
a, A	<SOH>
b, B	<STX>
c, C	<ETX>
d, D	<EOT>
e, E	<ENQ>
f, F	<ACK>
g, G	<BEL>
h, H	<BS>
i, I	<HT>
j, J	<LF>
k, K	<VT>
l, L	<FF>
m, M	<CR>
n, N	<SO>
o, O	<SI>
p, P	<DLE>
q, Q	<DC1>
r, R	<DC2>
s, S	<DC3>
t, T	<DC4>
u, U	<NAK>
v, V	<SYN>
w, W	<ETB>
x, X	<CAN>
y, Y	<EM>
z, Z	<SUB>

caret Control Characters in stty *(continued)*	
^c	**Value**
[	<ESC>
****	<FS>
]	<GS>
^	<RS>
_	<US>
?	<DEL>
@	<NUL>

Combination Modes	Description
cooked	See the **-raw** option.
ek	Sets ERASE and KILL characters to the Ctrl-H and Ctrl-U key sequences, respectively.
evenp	Enables **parenb** and **cs7**.
-evenp	Disables **parenb** and sets **cs8**.
lcase, **LCASE**	Sets **xcase**, **iuclc**, and **olcuc**. Used for workstations with uppercase characters only.
-lcase, **-LCASE**	Sets **-xcase**, **-iuclc**, and **-olcuc**.
nl	Sets **-icrnl** and **-onlcr**.
-nl	Sets **icrnl**, **onlcr**, **-inlcr**, **-igncr**, **-ocrnl**, and **-onlret**.
oddp	Enables **parenb**, **cs7**, and **parodd**.
-oddp	Disables **parenb** and sets **cs8**.
parity	See the **evenp** option.
-parity	See the **-evenp** option.
sane	Resets parameters to reasonable values.
raw	Allows raw mode input (no input processing, such as erase, kill, or interrupt); parity bit passed back.
-raw	Allows canonical input mode.
tabs	Preserves tabs.
-tabs, **tab3**	Replaces tabs with spaces when printing.

Window size	Description
cols *n*, **columns** *n*	The terminal (window) size is recorded as having *n* columns.
rows *n*	The terminal (window) size is recorded as having *n* rows.
size	Prints the terminal (window) sizes to standard output (first rows and then columns).

Obsolete Options

The following BSD options are supported by the **stty** command. For each of them, the recommended POSIX option is given.

Item	Description
all	Use the **stty -a** command to display all current settings.
crt	Use the **sane** option to reset parameters to reasonable values.
crtbs	Use the **-echoe** option.
crterase	Use the **echoe** option.
-crterase	Use the **-echoe** option.
crtkill	Use the **echoke** option.
-crtkill	Use the **echok** and **-echoke** options.
ctlecho	Use the **echoctl** option.
-ctlecho	Use the **-echoctl** option.
decctlq	Use the **-ixany** option.
-decctlq	Use the **ixany** option.
even	Use the **evenp** option.
-even	Use the **-evenp** option.
everything	Use the **stty -a** command to display all current settings.
litout	Use the **-opost** option.
-litout	Use the **opost** option.
odd	Use the **oddp** option.
-odd	Use the **-oddp** option.
pass8	Use the **-istrip** option.
-pass8	Use the **istrip** option.
prterase	Use the **echoprt** option.
speed	Use the **stty** command to display current settings.
tandem	Use the **ixoff** option.
-tandem	Use the **-ixoff** option.

Examples

1. To display a short listing of your workstation configuration, type:

   ```
   stty
   ```

   This lists settings that differ from the defaults.
2. To display a full listing of your workstation configuration, type:

   ```
   stty -a
   ```
3. To enable a key sequence that stops listings from scrolling off the screen, type:

   ```
   stty ixon ixany
   ```

   This sets **ixon** mode, which lets you stop runaway listing by pressing the Ctrl-S key sequence. The **ixany** flag allows you to resume the listing by pressing any key. The normal workstation configuration includes the **ixon** and **ixany** flags, which allows you to stop a listing with the Ctrl-S key sequence that only the Ctrl-Q key sequence will restart.
4. To reset the configuration after it has been messed up, type:

```
Ctrl-J stty  sane Ctrl-J
```

Press the Ctrl-J key sequence before and after the command instead of the Enter key. The system usually recognizes the Ctrl-J key sequence when the parameters that control Enter key processing are messed up.

Sometimes the information displayed on the screen may look strange, or the system will not respond when you press the Enter key. This can happen when you use the **stty** command with parameters that are incompatible or that do things you don't understand. It can also happen when a screen-oriented application ends abnormally and does not have a chance to reset the workstation configuration.

Entering the **stty sane** command sets a reasonable configuration, but it may differ slightly from your normal configuration.

5. To save and restore the terminal's configuration:

```
OLDCONFIG=`stty -g`          # save configuration
stty -echo                   # do not display password
echo "Enter password: \c"
read PASSWD                  # get the password
stty $OLDCONFIG              # restore configuration
```

This command saves the workstation's configuration, turns off echoing, reads a password, and restores the original configuration.

Entering the **stty -echo** command turns off echoing, which means that the password does not appear on the screen when you type it at the keyboard. This action has nothing to do with the **echo** command, which displays a message on the screen.

File

Item	Description
/usr/bin/stty	Contains the **stty** command.

style Command

Purpose

Analyzes surface characteristics of a document.

Syntax

style [**-a**] [**-e**] [**-l***Number*] [**-ml**] [**-mm**] [**-p**] [**-P**] [**-r***Number*] *File* ...

Description

The **style** command analyzes the surface characteristics of the writing style of an English-language document. It reports on readability, sentence length and structure, word length and usage, verb type, and sentence openers. Because the **style** command runs the **deroff** command before looking at the text, header files that contain appropriate formatting information should be included as part of the input.

Note: The use of nonstandard formatting macros may cause incorrect sentence breaks.

Flags

Item	Description
-a	Prints all sentences with their length and readability index.
-e	Prints all sentences that begin with an expletive such as "`There are`".
-l*Number*	Prints all sentences longer than the number of words specified by the parameter *Number*.

Item	Description
-ml	Causes the **deroff** command to skip lists; use **-ml** if a document contains many lists of sentence fragments.
-mm	Overrides the default **ms** macro package.
-p	Prints all sentences that contain a passive verb.
-P	Prints parts of speech of the words in the document.
-r*Number*	Prints all sentences whose readability index is greater than *Number*.

su Command

Purpose

Changes the user ID associated with a session.

Syntax

su [**-**] [*Name* [*Argument ...*]]

Description

The **su** command changes user credentials to those of the root user or to the user specified by the *Name* parameter, and initiates a new session. The user name might include a Distributed Computing Environment (DCE) cell specification.

Note: The root user is not required to satisfy the DCE authentication when switching to a DCE user. In this case, the user's DCE credentials are not required.

Any arguments, such as flags or parameters, that are specified by the *Arguments* parameter must relate to the login shell defined for the user specified by the *Name* parameter. These arguments are passed to the specified user's login shell. For example, if the login shell for user `Fred` is **/usr/bin/csh**, you can include any of the flags for the **csh** command, such as the **-f** flag. When the **su** command runs, it passes the **-f** flag to the **csh** command. When the **csh** command runs, the **-f** flag omits the **.cshrc** startup script.

Note: If the *domainlessgroups* attribute is set in the **/etc/secvars.cfg** file and if the user belongs to the Lightweight Directory Access Protocol (LDAP) domain or files domain, all the group IDs are fetched from the LDAP domain and the files domain.

The following functions are performed by the **su** command:

Item	Description
account checking	Validates the user account to be certain it exists, that it is enabled for the **su** command, that the current user is in a group permitted to switch to this account with the **su** command, and that it can be used from the current controlling terminal.
user authentication	Validates the user's identity, using the system-defined primary authentication methods for the user. If a password has expired, the user must supply a new password.
credentials establishment	Establishes initial user credentials, using the values in the user database. These credentials define the user's access rights and accountability on the system.

Item	Description
session initiation	If the **-** flag is specified, the **su** command initializes the user environment from the values in the user database and the **/etc/environment** file. When the **-** flag is not used, the **su** command does not change the directory.

These functions are performed in the sequence shown. If one function is unsuccessful, the succeeding functions are not done. Refer to the **ckuseracct**, **ckuserID**, **authenticate**, **setpcred**, and **setpenv** subroutines for the semantics of these functions.

To restore the previous session, type `exit` or press the Ctrl-D key sequence. This action ends the shell called by the **su** command and returns you to the previous shell, user ID, and environment.

If the **su** command is run from the **/usr/bin/tsh** shell, the trusted shell, you exit from that shell. The **su** command does not change the security characteristics of the controlling terminal.

Each time the **su** command is executed, an entry is made in the **/var/adm/sulog** file. The **/var/adm/sulog** file records the following information: date, time, system name, and login name. The **/var/adm/sulog** file also records whether or not the login attempt was successful: a + (plus sign) indicates a successful login, and a - (minus sign) indicates an unsuccessful login.

Note: Successful use of the **su** command resets the **unsuccessful_login_count** attribute in the **/etc/security/lastlog** file only if the user's **rlogin** and **login** attributes are both set to **false** in **/etc/security/user**. Otherwise, the **su** command doesn't reset the **unsuccessful_login_count**, because the administrator often uses the **su** command to fix user account problems. The user is able to reset the attribute through a local or remote login.

Flags

Item	Description
-	Specifies that the process environment is to be set as if the user had logged in to the system using the **login** command. Nothing in the current environment is propagated to the new shell. **Note:** This behavior is intended for compatibility with alternate UNIX shell environments where flag options are allowed ahead of the Name parameter.

Security

The **su** command is a PAM-enabled application with a service name of `su`. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the `usw` stanza of **/etc/security/login.cfg**, to `PAM_AUTH` as the root user.

The authentication mechanisms used when PAM is enabled depend on the configuration for the `su` service in **/etc/pam.conf**. The **su** command requires **/etc/pam.conf** entries for the `auth`, `account`, `password`, and `session` module types. In order for the **su** command to exhibit a similar behavior through PAM authentication as seen in standard AIXauthentication, the `pam_allowroot` module must be used as sufficient and called before **pam_aix** in both the `auth` and `account` su service stacks. Listed below is a recommended configuration in **/etc/pam.conf** for the `su` service:

```
#
# AIX su configuration
#
su auth sufficient /usr/lib/security/pam_allowroot
su auth required /usr/lib/security/pam_aix

su account sufficient /usr/lib/security/pam_allowroot
su account required /usr/lib/security/pam_aix

su session required /usr/lib/security/pam_aix

su password required /usr/lib/security/pam_aix
```

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

To get the full functionality of the command, besides the **accessauths**, the role should also have the **aix.security.su** authorization.

On a Trusted AIX system, when the **su** command is invoked with the **-** flag, the following conditions must be met for both sensitivity and integrity labels:

- The current user's maximum clearance must dominate the new user's maximum clearance.
- The new user's minimum clearance must dominate the current user's minimum clearance.
- The current user's effective clearance must be dominated by the new user's maximum clearance and must dominate the new user's minimum clearance.

Examples

1. To obtain root user authority, enter one of the following commands:

   ```
   su
   ```

   This command runs a subshell with the effective user ID and privileges of the root user. You will be asked for the root password. Press End-of-File, Ctrl+D key sequence, to end the subshell and return to your original shell session and privileges.

   ```
   su --
   ```

   This command runs a subshell with the effective user ID and privileges of the `root` user. Enter the `root` password, when prompted. Press End-of-File, Ctrl+D key sequence, to end the subshell and return to your original shell session and privileges.
2. To obtain the privileges of the `jim` user, enter the following command:

   ```
   su jim
   ```

   This command runs a subshell with the effective user ID and privileges of `jim`.
3. To set up the environment as if you had logged in as the `jim` user, enter:

   ```
   su  - jim
   ```

   This starts a subshell using `jim`'s login environment.
4. To run the backup command with root user authority and then return to your original shell, enter:

   ```
   su root "-c /usr/sbin/backup -9 -u"
   ```

   This command runs the **backup** command with root user authority within root's default shell. You must give the correct root password when queried for the command to execute.
5. Enter one of the following commands to change the user credentials of the current session to root user:

   ```
   su -
   ```

   ```
   su - root
   ```

   ```
   su - --
   ```

   The preceding commands start a subshell by using the `root` user's login environment.

Files

Item	Description
/usr/bin/su	Contains the **su** command.
/etc/environment	Contains user environment values.
/etc/group	Contains the basic group attributes.
/etc/passwd	Contains the basic user attributes.
/etc/security/user	Contains the extended attributes of users.
/etc/security/environ	Contains the environment attributes of users.
/etc/security/limits	Contains the process resource limits of users.
/etc/security/passwd	Contains password information.
/var/adm/sulog	Contains information about login attempts.
/etc/security/enc/LabelEncodings	Contains label definitions for the Trusted AIX system.

subj Command

Purpose

Generates a list of subjects from a document.

Syntax

subj [*File* ...]

Description

The **subj** command searches one or more English-language files for subjects that might be appropriate in a subject-page index and prints the list of subjects on the standard output. The document should contain formatting commands (from the **nroff**, **troff**, and **mm** commands, among others) to make the best use of the **subj** command.

The **subj** command selects sequences of capitalized words as subjects, except for the first word in each sentence. Thus, if a sentence begins with a proper noun, the capitalization rule does not select this word as a subject. However, since each sentence is expected to begin on a new line, the first word of a sentence that begins in the middle of a line may be erroneously selected. Also, the **subj** command selects modifier-noun sequences from the abstract, headings, and topic sentences (the first sentence in each paragraph). Thus, occasionally a word is incorrectly categorized as a noun or adjective.

The output of the **subj** command may not be appropriate for your needs and should be edited accordingly.

Parameters

Item	Description
File	Specifies the English-language files that the **subj** command searches for appropriate subjects for indexing.

sum Command

Purpose

Displays the checksum and block count of a file.

Syntax

sum [**-i**] [**-r** | **-o**] [*File ...*]

Description

The **sum** command reads the file specified by the *File* parameter and calculates a checksum and the number of 1024-byte blocks in that file. If no options are specified, a byte-by-byte algorithm, such as the BSD 4.3 default algorithm, is used. If no files are named, the standard input is read. The checksum and number of 1024-byte blocks are written to standard output. The **sum** command is generally used to determine if a file that has been copied or communicated over transmission lines is an exact copy of the original.

Flags

Item	Description
-i	Allows the user to compute the checksum without including header information, if the input file is a binary file. If the input file is not a binary file, the checksum includes header information.
-o	Uses the word-by-word algorithm to compute the checksum. The **sum** command with the **-o** flag is compatible with the Version 2 **sum** command in terms of the checksum, but not the number of blocks.
-r	Uses a byte-by-byte algorithm to compute the checksum. Using the **-r** flag is the same as using no options.

Note: The default is no longer the word-by-word computation algorithm; it is the BSD 4.3 default algorithm.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

To display the checksum of, and the number of 1024-byte blocks in, the **file1** and **file2** files, type:

```
sum file1 file2
```

If the checksum of the **file1** file is 32830, the checksum of the **file2** file is 32481, and the **file1** file contains one block, and the **file2** contains four blocks, the **sum** command displays:

```
32830           1       file1
32481           4       file2
```

Files

Item	Description
/usr/bin/sum	Contains the **sum** command.

suma Command

Purpose

Creates a task to automate the download of technology levels and service packs from a fix server.

Syntax

To create, edit, or schedule a SUMA task:

suma { { [**-x**] [**-w**] } | **-s** *CronSched* } [**-a** *Field=Value*]... [*TaskID*]

To list SUMA tasks:

suma -l [*TaskID*]...

To list or edit the default SUMA task:

suma -D [**-a** *Field=Value*]...

To list or edit the SUMA global configuration settings:

suma -c [**-a** *Field=Value*]...

To unschedule a SUMA task:

suma -u *TaskID*

To delete a SUMA task:

suma -d *TaskID*

Description

The **suma** command can be used to perform the following operations on a SUMA task or policy:

- Create
- Edit
- List
- Schedule
- Unschedule
- Delete

The specified operation is performed on the task represented by a unique Task ID. For the create or edit cases on a SUMA task, if the *TaskID* is not specified, the create operation is assumed, and a unique *TaskID* is generated. For the **-l** flag, if *TaskID* is not specified, a list of all SUMA tasks are displayed. For the **-c** flag, if the **-a** flag is not specified, the SUMA global configuration settings are listed.

Flags

Item	Description
-c	Lists or edits the SUMA global configuration settings. The **-a** flag allows one or more configuration setting to be updated to the specified value. When used without the **-a** flag, all SUMA configuration settings are listed. The configuration settings that can be edited with the **-a** flag are as follows: **FIXSERVER_PROTOCOL** When communicating with the fix server, this specifies that the transfer utilizes https (secure). The https protocol is the only supported protocol and cannot be changed. Default value: `https` Allowable value: `https`. **DOWNLOAD_PROTOCOL** When downloading file sets, this specifies whether the transfer utilizes http, or https (secure) transfers. The http protocol takes advantage of multi-threaded performance and utilizes the download director protocol (ddp). The https protocol is single-threaded. Default value: `http` Allowable values: `http`, `https`. **DL_TIMEOUT_SEC** Specifies the time in seconds to wait for a response from the fix server during a download operation. Default value: `180` Allowable values: Whole numbers greater than zero. **HTTP_PROXY and HTTPS_PROXY** Proxy server and port to use for the HTTP or HTTPS transfers. The SUMA command shares the proxy connectivity settings with the Electronic Service Agent. The HTTP or HTTPS proxy service configuration can be set up through the SMIT **Create/Change Service Configuration** menus (use fastpath smitty srv_conn) that allow the server specifications such as IP address, port number, and an optional user ID and password. SUMA no longer supports the settings of the HTTP_PROXY and HTTPS_PROXY parameters. Default value: blank (disabled) Allowable value: blank

Item	Description
-c *(Continued)*	**SCREEN_VERBOSE** Specifies a verbosity level for logging information to stdout and stderr. Used when the **suma** command is run from the command line or the SMIT interface. It is not applicable for scheduled tasks run from cron. Default value: LVL_INFO Allowable values: • LVL_OFF : No information is displayed or logged. • LVL_ERROR : Displays error messages and other highly important messages. • LVL_WARNING : Displays warning messages in addition to LVL_ERROR messages. • LVL_INFO : Displays informational messages in addition to LVL_WARNING messages. • LVL_VERBOSE : Displays verbose informational messages in addition to LVL_INFO messages. • LVL_DEBUG : Displays debug output. This setting is for debugging purposes and should not be used for normal operations. **NOTIFY_VERBOSE** Specifies a verbosity level for the information sent in an email notification. Only applies to scheduled tasks run from cron.Default value: LVL_INFO Allowable values: LVL_OFF, LVL_ERROR, LVL_WARNING, LVL_INFO, LVL_VERBOSE, LVL_DEBUG (refer to the **SCREEN_VERBOSE** setting for value descriptions) **LOGFILE_VERBOSE** Specifies a verbosity level for the information that is logged to the log file (**/var/adm/ras/suma.log**). **Note:** An LVL_OFF setting will still log information to the download log file (**/var/adm/ras/suma_dl.log**).Default value: LVL_VERBOSE Allowable values: LVL_OFF, LVL_ERROR, LVL_WARNING, LVL_INFO, LVL_VERBOSE, LVL_DEBUG (refer to the **SCREEN_VERBOSE** setting for value descriptions) **MAXLOGSIZE_MB** The maximum size (in MB) that a log file is allowed to reach. Default value: 1 Allowable values: Whole numbers greater than zero. **REMOVE_CONFLICTING_UPDATES** Specifies if **lppmgr** should remove conflicting updates that have the same level as base images (**lppmgr -u** flag) when run during a clean action. Default value: yes Allowable values: yes, no **REMOVE_DUP_BASE_LEVELS** Specifies whether **lppmgr** should remove duplicate base levels (**lppmgr -b** flag) when run during a clean action. Default value: yes Allowable values: yes, no
-c *(Continued)*	**REMOVE_SUPERSEDE** Specifies whether **lppmgr** should remove superseded file set updates (**lppmgr -x** flag) when run during a clean action. Default value: yes Allowable values: yes, no **TMPDIR** Specifies the directory to store temporary files. Default value: **/var/suma/tmp** Allowable values: Any directory that currently exists.
-d	Deletes the SUMA task associated with the given *TaskID* and any schedules for this task that were created with the **-s** flag.

Item	Description
-D	Lists or edits the default SUMA task. The **-a** flag allows one or more *Fields* of the default task to be updated to the specified *Value*. When used without the **-a** flag, the default SUMA task will be listed.
-l	Lists SUMA tasks. When used without a *TaskID*, all SUMA tasks will be listed. The *TaskID* can be used to specify one or more task IDs to list.
-s *CronSched*	Schedules a SUMA task. If specified when a new task is being created, a save is implied (-w flag functionality). The *CronSched* is a list of five space-separated entries (minute, hour, day, month, weekday) contained in quotation marks. The valid values for these entries are as follows (see the **crontab** man page for additional details): • Minute: 0 - 59 • Hour: 0 - 23 • Day: 1 - 31 • Month: 1 - 12 • Weekday: 0 - 6 (for Sunday - Saturday)
-u	Unschedules a SUMA task. This removes any scheduling information for the specified *TaskID*.
-w	Writes or saves a SUMA task. If used instead of the **-s** flag, the task is saved, allowing scheduling information to be added later. If used with the **-x** flag, the task is run immediately and also saved.
-x	Specifies that a SUMA task should be run immediately and not scheduled. If used without the **-w** flag, the task is not saved for future use.
-a *Field*=*Value* ...	Assigns the specified *Value* to the specified *Field*. For the create or edit operation on a SUMA task, the following are the supported *Fields* and *Values*. **RqType** When **suma** is run with an **RqType** of **Latest**, the **RqType** is the only required field. See example 1 for the default values that will be used in this case. Other **RqType** values (**TL**, **SP**, **ML**, **PTF**) require specification of additional *Field*=*Value* information. **ML** Specifies a request to download a specific maintenance or technology level. An example is `5300-11`. **TL** Specifies a request to download a specific technology level. An example is `6100-03`. **PTF** Specifies a request to download a PTF. An example is U813941. Only certain PTFs may be downloaded as an individual file set. For example, PTFs containing bos.rte.install, bos.alt_disk_install.rte, or PTFs that come out in between Service Packs. Otherwise, the TL or SP must be downloaded. **SP** Specifies a request to download a specific service pack. An example is `6100-02-04`. **Latest** Specifies a request to download the latest fixes. This **RqType** value returns the latest service pack of the **TL** specified in **FilterML**.

Item	Description
-a *(Continued)*	**RqName** The specific name of the item requested (for example, `6100-03` or `6100-04-03`). The **RqName** field should be blank when **RqType** equals **Latest**. **Repeats** Specifies whether the task is executed once and does not remain on the system, repeats until the item is found, or repeats forever. The **Repeats** field only applies to scheduled tasks run from cron that have an **Action** of **Download**, **Clean**, or **Metadata**. If run from the command line or if **Action** is **Preview**, this field is ignored, and no task is removed. **y** Sets up a repeating task, and requires that the task has been assigned a *CronSched* with the **-s** flag. When the **RqType** equals **TL**, **SP**, **PTF**, or **ML**, the task is removed as soon as the item is found. When **RqType** equals **Latest**, the task is set up to repeat forever. **n** Specifies that the task is executed once and does not remain on the system.

Item	Description
-a *(Continued)*	**DisplayName** Indicates the display name for this **SUMA** task (for example, "Download TL 6100-04 when available"). This is used when viewing existing SUMA tasks in SMIT. **Action** **Preview** Specifies that a download preview is performed. No file sets are downloaded. **Download** Specifies that file sets are downloaded into the **DLTarget** based on the policy. **Clean** Specifies that file sets are downloaded into the **DLTarget** based on the policy, followed by a clean operation. The **lppmgr** command is used to clean file sets that are not needed from the **DLTarget**. The three configurable **lppmgr** flag options listed in the SUMA global configuration settings are: • REMOVE_CONFLICTING_UPDATES • REMOVE_DUP_BASE_LEVELS • REMOVE_SUPERSEDE **Metadata** Specifies that metadata files are downloaded instead of file set updates. The following **RqType** values are supported: **TL** Downloads metadata for a specific technology level. **SP** Downloads metadata for a specific service pack. **Latest** Downloads metadata for all service packs for the technology level that is specified for the FilterML flag. **DLTarget** Contains the directory location where the downloaded files are stored. If this field is not specified, it is given the value **/usr/sys/inst.images** and the files are stored in a directory based on the image type; for example **/usr/sys/inst.images/installp/ppc** or **/usr/sys/inst.images/RPMS/ppc**. **NotifyEmail** Contains one or more e-mail addresses (multiple addresses should be comma-separated) that are sent a notification e-mail after a file set download or preview. A notification is sent only if the task is scheduled for execution at a future time (*CronSched* has been specified).

Item	Description
-a *(Continued)*	**FilterDir** Specifies the name of a fix repository directory to filter against so that duplicate fixes are not downloaded. This allows a directory other than the **DLTarget** to be filtered against. For example, you may filter against a NIM lpp_source without having to download into this directory. If left blank, the **DLTarget** is used. **FilterML** Specifies a technology level to filter against; for example, 6100-03. If not specified, the value returned by **oslevel -r** on the local system is used. **MaxDLSize** The maximum allowable amount of data to be downloaded by any single policy execution, in MB. If it is determined that the download operation exceeds this size, no download occurs. A value of "unlimited" or -1 can be specified to indicate no upper limit on the amount of data to be downloaded. **Extend** Specifying y automatically extends the filesystem where the DLTarget resides. If n is specified and additional space is required for the download, no download occurs. **MaxFSSize** The maximum allowable size to which the **DLTarget** filesystem can be extended, in MB. If it is determined that the download operation exceeds this limit, no download occurs. A value of "unlimited" or -1 can be specified to indicate no upper limit on the size of the filesystem (that is, the filesystem can be expanded until physical disk space is exhausted).

Parameters

Item	Description
TaskID	Specifies a unique numeric identifier that is associated with a task. This is assigned when a task is created.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Examples

1. To list the SUMA global configuration settings, type the following:

```
suma -c
```

Output similar to the following is displayed:

```
FIXSERVER_PROTOCOL=https
DOWNLOAD_PROTOCOL=http
DL_TIMEOUT_SEC=180
DL_RETRY=1
HTTP_PROXY=
HTTPS_PROXY=
SCREEN_VERBOSE=LVL_INFO
NOTIFY_VERBOSE=LVL_INFO
LOGFILE_VERBOSE=LVL_VERBOSE
MAXLOGSIZE_MB=1
REMOVE_CONFLICTING_UPDATES=yes
REMOVE_DUP_BASE_LEVELS=yes
```

```
REMOVE_SUPERSEDE=yes
TMPDIR=/var/suma/tmp
```

2. To edit the SUMA global configuration setting to change the maximum log file size to 2 MB, type the following:

```
suma -c -a MAXLOGSIZE_MB=2
```

3. To list the SUMA task defaults, type the following:

```
suma -D
```

Output similar to the following is displayed:

```
DisplayName=
Action=Download
RqType=Latest
RqName=
Repeats=y
DLTarget=/usr/sys/inst.images
NotifyEmail=root
FilterDir=/usr/sys/inst.images
FilterML=
MaxDLSize=-1
Extend=y
MaxFSSize=-1
```

4. To create and schedule a task that downloads the latest fixes monthly (for example, on the 15th of every month at 2:30 a.m.), type the following:

```
suma -s "30 2 15 * *" -a RqType=Latest \
-a DisplayName="Latest fixes - 15th Monthly"
```

Note: A task ID is returned for this newly created task. This example assumes some of the SUMA task defaults, as displayed in the **suma -D** example, are utilized. For example, when the task default of **DLTarget=/usr/sys/inst.images**, the installp images are downloaded into the **/usr/sys/inst.images/installp/ppc** directory.

5. To view SUMA scheduling information that has been set up by running a **suma -s CronSched** command, type the following:

```
crontab -l root
```

6. To create and schedule a task that checks for a specific TL once a week (for example, every Thursday at 3 a.m.), downloads it when it becomes available, and sends e-mail notifications to users on a remote system, type the following:

```
suma -s "0 3 * * 4" -a RqType=TL -a RqName=6100-04 \
-a NotifyEmail="bob.smith@host2,ann@host2"
```

Note: For this task to make a weekly check for a TL, the **Repeats** field needs to be set to **y**. In this case, after the TL is found, the task is deleted. If **Repeats=n**, only a single check occurs before deleting the task.

7. To create and schedule a task that checks for critical fixes monthly (for example, on the 20th of every month at 4:30 a.m.), type the following:

```
suma -s "30 4 20 * *" -a RqType=Latest -a RqName= \
-a RqLevel=latest -a Repeats=y
```

Note: By setting **Repeats=y**, this task 'repeats forever' and is not deleted after a successful download.

8. To create and schedule a task that downloads the entire AIX Version 7.1 with the 5300-11 Recommended Maintenance package into the **/lppsrc/5311** directory on Monday at 11:00 p.m., and

runs an **lppmgr** clean operation after the download operation to remove any superseded updates, duplicate base levels, and conflicting updates, type the following:

```
suma -s "0 23 * * 1" -a Action=Clean -a RqType=ML -a RqName=5300-11 \
-a DLTarget=/lppsrc/5311
```

Note: Prior to running a task that specifies **Action=Clean**, you can run **suma -c** to verify the SUMA global configuration settings that are used when you run **lppmgr** command. In this case, having REMOVE_SUPERSEDE, REMOVE_DUP_BASE_LEVELS, and REMOVE_CONFLICTING_UPDATES all set to **yes** results in the action previously described.

9. To create and schedule a task that downloads the entire AIX Version 7.1 with the 5300-11 Recommended Maintenance package into the **/tmp/lppsrc/5311** directory on Monday at 11:00 p.m., filtering against any updates already contained in **/lppsrc**, type the following:

   ```
   suma -s "0 23 * * 1" -a RqType=ML -a RqName=5300-11 \
   -a DLTarget=/tmp/lppsrc/5311 -a FilterDir=/lppsrc -a FilterSysFile=/dev/null
   ```

   Note: After the task is successfully completed, the task is removed, because **RqType=TL** is a 'repeat until found' task. However, if **Repeats=n**, only a single check for the 5300-03 TL is made, and if the TL is not found on the fix server, the task is deleted because it has been set up not to repeat.

10. To immediately execute a task that performs a preview to check if an SP exists on the fix server, and to create and save this task for later scheduling if the SP does not yet exist, type the following:

    ```
    suma -x -w -a Action=Preview -a RqType=SP -a RqName=6100-04-02
    ```

    Note: A task ID is returned for this newly created task.

11. To immediately execute the newly created task from the above example (assume task ID 23 was returned) and attempt to download the SP and save the **Action=Download** setting for task ID 23, type the following:

    ```
    suma -x -w -a Action=Download 23
    ```

    Note: Because this task is being run from the command line, and not scheduled through cron, the **Repeats** field are ignored and this task is not deleted regardless of whether the SP is found.

12. To schedule task ID 23 to repeatedly check for a specific SP once a week (for example, every Thursday at 3 a.m.), and download it when it becomes available, type the following:

    ```
    suma -s "0 3 * * 4" -a Repeats=y 23
    ```

    Note: This task is deleted when the SP is found.

13. To unschedule a task that removes its scheduling information from the crontab file in the **/var/spool/cron/crontabs** directory, type the following:

    ```
    suma -u 23
    ```

14. To delete a task that also removes its scheduling information if it exists, type the following:

    ```
    suma -d 23
    ```

15. To list multiple SUMA tasks, where 4 and 23 represent task IDs, type the following:

    ```
    suma -l 4 23
    ```

16. To list all SUMA tasks, type the following:

    ```
    suma -l
    ```

17. To create and schedule a task that checks monthly (for example, on the 15th of every month at 2:30 a.m.) for the latest service pack on the specified **FilterML**, and download any that are not already in the **/tmp/latest** repository, type the following:

```
suma -s "30 2 15 * *" -a RqType=Latest -a FilterML=6100-02 \
-a DLTarget=/tmp/latest -a FilterDir=/tmp/latest
```

Note: A task ID is returned for this newly created task.

Location

/usr/suma/bin/suma

Files

Item	Description
/usr/suma/bin/suma	Contains the **suma** command.
/usr/sbin/suma	Link to **/usr/suma/bin/suma**.
/var/adm/ras/suma.log	Contains detailed results from running the **suma** command.
/var/adm/ras/suma_dl.log	Contains a list of files that have been downloaded.
/var/spool/cron/crontabs	Directory that contains the crontab file for scheduling.

suspendvsd Command

Purpose

suspendvsd – Deactivates an available virtual shared disk.

Syntax

suspendvsd {-a | *vsd_name*...}

Description

The **suspendvsd** command brings the specified virtual shared disks from the active state to the suspended state. They remain available. Read and write requests which were active while the virtual shared disk was active are suspended and held. Subsequent read and write operations are also held. If the virtual shared disk is in the suspended state, this command leaves it in the suspended state.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Suspend a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Flags

-a
Specifies that all the virtual shared disks in the active state are to be suspended.

Parameters

vsd_name
Specifies a virtual shared disk. If the virtual shared disk is not in the active state, you get an error message.

Security

You must have root authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Standard Output

Current RVSD subsystem run level.

Examples

To bring the virtual shared disk **vsd1vg1n1** from the active state to the suspended state, enter:

```
suspendvsd vsd1vg1n1
```

Location

/opt/rsct/vsd/bin/suspendvsd

svmon Command

Purpose

Captures and analyzes a snapshot of virtual memory.

Syntax

Command report

svmon -C *commands* [**-O** *options*] [**-t** *count*] [**-i** *interval* [*numintervals*]] [**-@** [**ALL** | *wparnames*]]

Detailed segment report

svmon -D *sids* [**-O** *options*] [**-i** *interval* [*numintervals*]]

Global report

svmon -G [**-O** *options*] [**-i** *interval* [*numintervals*]] [**-@** [**ALL** | *wparnames*]]

Process report

svmon -P [*pids*] [**-O** *options*] [[**-t** *count*] [**-i** *interval* [*numintervals*]] [**-@** [**ALL** | *wparnames*]]

Segment report

svmon -S [*sids*] [**-O** *options*] [**-t** *count*] [**-i** *interval* [*numintervals*]] [**-@** [**ALL** | *wparnames*]]

User report

svmon -U [*lognames*] [**-O** *options*] [**-t** *count*] [**-i** *interval* [*numintervals*]] [**-@** [**ALL** | *wparnames*]]

Workload management class report

svmon -W [*classnames*] [**-O** *options*] [**-t** *count*] [**-i** *interval* [*numintervals*]] [**-@** [**ALL** | *wparnames*]]

Workload management tier report

svmon -T [*tiers*] [**-O** *options*] [**-a** *supclassname*] [**-t** *count*] [**-i** *interval* [*numintervals*]] [**-@** [**ALL** | *wparnames*]]

XML report

svmon X [**-o** *filename*] [**-i** *interval* [*numintervals*]] [**-c** < *comment* >] [**-O** *options*]

Description

The **svmon** command displays information about the current state of memory. However, the displayed information does not constitute a true snapshot of memory because the **svmon** command runs at user level with interrupts enabled.

If you specify no flag, the **svmon** command, by default, reports real memory at the system level.

You can see memory consumption details and generate the following types of reports. To see more information about a type of report, select one of the following links:

- Command report
- Detailed segment report
- Global report
- Process report
- Segment report
- User report
- Workload management class report
- Workload management tier report
- XML report

The output of these reports can be in compact format or long format. To generate compact format report, specify the **-O** flag. If you do not specify the **-O** flag, the report is in long format.

Command report

The command report displays the statistics of memory use for the specified command. To print this report, specify the **-C** flag. The command report can be in compact format or in long format:

Item	Description
Compact report	A one line summary for each command. To set compact report as the default format, specify the **-O** flag.
Long report	A multiple lines report for each command that contains a summary, a size-per-page report, and the details of the segments. To set long report as the default format, do not specify the **-O** flag.

Detailed segment report

The detailed segment report displays detailed information about the primary segments that are specified. To print the detailed segment report, specify the **-D** flag.

The detailed segment report is in long report format only.

Global report

The global report displays the statistics of the real memory and paging space that are in use for the whole system. If you do not specify any flag, the global report is the default format of report that the **svmon** command generates.

To print the global report, specify the **-G** flag.

The global report can be in compact format or long format:

Item	Description
Compact report	A report on only the main metrics of the system. This report is one line with a maximum of 160 characters.
Long report	A summary of memory, page size, and affinity domain. The report is multiple lines, which is the default format of global report. By default, the following metrics are displayed: • The **memory** metric displays the memory consumption of the machine. • The **Page Size** metric displays the memory consumption of the **Page Size**. • The **Affinity Domain** metric reports the memory affinity by affinity domain.

Note: Pinned memory pages in the Global report of the **svmon** command includes kernel locked pages when the kernal lock (`vmm_klock_mode` option) is enabled. For more information about the kernal lock option, refer to the **vmo -h vmm_klock_mode** command documentation.

Process report

The process report displays the memory use for the specified active process. If you do not specify a list of processes, the **svmon** command displays the memory use statistics for all active processes.

To print the process report, specify the **-P** flag.

The process report can be in compact format or long format:

Item	Description
Compact report	A one line report for each process. To set the compact report as the default format, specify the **-O** flag.
Long report	A multiple lines summary for each process. To set the long report as the default format, do not specify the **-O** flag. This report contains a summary for each process, a per-page-size report, and the details of the segments.

Note: The **svmon** command does not show the decrease in the count for the memory usage when the application releases the memory. When the memory is released from the application, it goes back to the memory free list of the per-process. The **svmon** command accounts for the memory that is released as the allocated memory for that application.

Segment report

The segment report displays the statistics of memory use for the specified segments. To display the statistics for all of the defined segments, do not specify any list.

To print the segment report, specify the **-S** flag.

The segment report includes metrics for each specified segment. The report contains several lines of metrics for each segment.

User report

The user report displays the statistics of memory use for the specified users (login names). To display the statistics for all of the users, do not specify any list of login names.

To print the user report, specify the **-U** flag.

The user report can be in compact format or long format:

Item	Description
Compact report	A one line report for each user. To set the compact report as the default format, specify the **-O** flag.
Long report	A multiple lines summary for each user. To set the long report as the default format, do not specify the **-O** flag. This report contains a summary for each user, a per-page-size report, and the details of the segments.

Workload management class report

The workload management class report displays statistics of memory use for the specified workload management classes. To display the statistics for all of the defined classes, do not specify any class.

To print the workload management class report, specify the **-W** flag.

Restriction: This report is available only when the Workload Manager is running. If the Workload Manager is not running, the following message is displayed and no statistics are reported:

```
WLM must be started
```

If the Workload Manager is running in passive mode, the **svmon** command displays the following message before displaying the statistics:

```
WLM is running in passive mode
```

The workload management class report can be in compact format or long format:

Item	Description
Compact report	A one line report for each class. To set the compact report as the default format, specify the **-O** flag.
Long report	A multiple lines summary for each class. To set the long report as the default format, do not specify the **-O** flag. This report contains a summary for each class, a per-page size report, and the details of the segments.

Workload management tier report

The workload management tier report displays information about the tiers, such as the tier number, the superclass name, and the total number of pages in real memory from segments belonging to the tier.

To print the tier report, specify the **-T** flag. Only the long report format is supported.

Restriction: This report is available only when the Workload Manager is running. If the Workload Manager is not running, the following message is displayed and no statistics are reported:

```
WLM must be started
```

If the Workload Manager is running in passive mode, the **svmon** command displays the following message before displaying the statistics:

```
WLM is running in passive mode
```

XML report

You can use the **svmon** command with an **-X** flag to generate a report in XML format. The XML report contains data of the global environment, the processes, the segments, the users, the workload management classes, and the commands running on the system.

The report is by default printed to the standard output. To print the output to a file named *filename*, specify the **-O** *filename* flag. The extension of the output file will be **.svm**.

The **.svm** file uses an XML Schema Definition (XSD) that the **/usr/lib/perf/svmon_schema.xsd** file defines. You can use the XML data in the XML reports to build custom applications because the schema is self-documented.

In the XML report, if you do not specify the **-O affinity** argument, or set it to the off value, only the domain affinity at system level is reported.

Flags

If no command line flag is given, then the **-G** flag is the default.

Item	Description
`-@` [ALL | *wparnames*]	Displays report for the workload partitions. The **-@ ALL** option specifies to display the report for all of the WPAR starting with the global report, and to process all of the available WPAR, sorting them by the name. When you specify a list of WPAR names in the *wparnames* parameter, the WPAR information is displayed in a header, and the report is displayed without adding WPAR information. All information displayed is restricted to the WPAR that was processed and has meaning only inside the WPAR. For example, the `pid` displayed is virtual `pid`, which is the `pid` inside the WPAR. The same rule applies to the `svmon` options. Each WPAR name in the list is processed in the given order and each `svmon` report is separated by the `WPARname` header. When you do not specify a list, the `svmon` command adds WPAR information to existing reports. The **pid** section and **segments** section of the report contain the WPAR name when one is available. Virtual `pid` information might also be displayed. When all of the keywords are used, the `svmon` command processes all of the available WPAR, sorting them by the WPAR name. **Note:** The **-@** flag is not supported when executed within a workload partition.
-a *supclassname*	Restricts the scope to the subclasses of the *supclassname* parameter (in the Tier report that is returned with the **-T** flag).
-c < *comment* >	Adds a comment, specified by the *comment* parameter, into the XML report. Use the **-c** flag with the **-X** flag.
-C *commands*	Displays memory use statistics for the processes running the commands that are specified by the *commands* parameter.

Item	Description
-D *sids*	Displays memory use statistics for the segments that the *sids* parameter specifies, and a detail status of all of the frames of each segment.
-G	Displays a global report.
-i *interval* [*numintervals*]	Displays statistics repetitively. The **svmon** command collects and prints statistics in the interval that the *interval* parameter specifies. The *numintervals* parameter specifies the number of repetitions. If the *numintervals* parameter is not specified, the **svmon** command runs until you interrupt it (Ctrl+C). **Tip:** The observed interval might be larger than the specified interval because it might take a few seconds to collect statistics for some options.
-o *filename*	Specifies the output file with the *filename* parameter for XML reports. Use this flag with the **-X** flag.
-O *options*	Changes the content and presentation of the reports that the **svmon** command generates. You can specify values to the *options* parameter to modify the output. **Tip:** To overwrite the default values that are defined previously by the **-O** *options* flag, you can define the **.svmonrc** configuration file in the directory where the **svmon** command is launched.
-P [*pids*]	Displays the memory-usage statistics for the processes that the *pids* parameter specifies.
-S [*sids*]	Displays the memory-usage statistics for segments that the *sids* parameter specifies. The *sids* parameter is a hexadecimal value. The segment IDs (SIDs) that are specified must be of primary segments. If you do not specify a list of SIDs, the statistics of memory use are displayed for all of the defined segments.
-t *count*	Displays the top object in the *count* parameter to be printed.
-T [*tiers*]	Displays the memory-usage statistics of all of the classes of the tier numbers that the *tiers* parameter specifies. If you do not specify a list of tiers, the statistics of memory use are displayed for all of the defined tiers.
-U [*lognames*]	Displays the memory-usage statistics for the login name that the *lognames* parameter specifies. If you do not specify a list of login identifiers, the statistics of the memory use are displayed for all of the defined login identifiers.
-W [*classnames*]	Displays the memory-usage statistics for the Workload Manager class that the *classnames* parameter specifies. If you do not specify a list of class names, the statistics of memory usage are displayed for all of the defined class names.
-X	Generates the XML report.

Parameters

Item	Description
commands	Specifies the commands to be reported in the command report (**-C**). The value of the *commands* parameter is a string. You can specify more than one command. The value of the *commands* parameter is the exact base name of an executable file.

Item	Description
options	Specifies the content and presentation of each report. Use this parameter with the **-O** flag. The values of the *options* parameter must be separated by commas, or enclosed in quotation marks (“ ”) and separated by commas or spaces. The following values are valid to the *options* parameter. **Tip:** The **scope** specifies the reports that support the value. • **activeuser** = [on | off] The **activeuser** argument specifies that the **svmon** command displays only the active user. – **Default value**: off – **Scope**: User report (**- U**) You can specify the following values to the **activeuser** option: **on** Displays only the active user. **off** Displays all of the user. • **affinity** = [on | detail | off] The **affinity** argument specifies that the **svmon** command displays the memory affinity at process level or segment level. – **Default value**: off – **Scope**: Global report (**-G**), process report (**-P**), and segment report (**-S**) You can specify the following values to the **affinity** option: **on** Displays memory affinity at process level **detail** Displays memory affinity at segment level **off** Does not display the memory affinity In the XML report, if you do not specify the **-O affinity** argument, or set it to the off value, only the domain affinity at system level is reported. **Note:** 1. Use the **-O affinity = detail** argument with caution. 2. The **summary** argument with the value of *longreal* or *longname* is not supported with the **affinity** argument. • **commandline** = [on | off] The **commandline** argument specifies that the **svmon** command displays the command that is used for the current report. – **Default value**: off – **Scope**: All reports You can specify the following values to the **commandline** option: **on** Displays the command that is used for the current report **off** Does not display the command that is used for the current report
options	*(Continued description of the valid values for the options parameter).* • **file_mem_scan** = [on | off] If the segment information for some files, such as remote files is not updated by the file system, by default, value of the **svmon** command does not collect the segment information for those files. By turning **file_mem_scan**=**on**, the **svmon** command scans the entire system's segment table to gather segment information of those files. – **Default value**: off – **Scope**: Command report (**-C**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**), global report when affinity is on (-G -O affinity = on) You can specify the following values for the **file_mem_scan** option: **on** Displays the report with client segments for all files including the files for which the segment information is not updated by the file system. **off** Displays the report with client segments for all files excluding the files for which the segment information is not updated by the file system. **Note:** If you use the value of **file_mem_scan** = **on**, the performance might be impacted based on the number of files opened while running the command and the number of segments in the system.

Item	Description
options	*(Continued description of the valid values for the options parameter).* • **filename** = [on | off] The **filename** argument specifies that the **svmon** command displays the file names of each file segment. – **Default value**: off – **Scope**: Command report (**-C**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**) You can specify the following values to the **filename** option: **on** Displays the file names of each file segment **off** Does not displays the file name of each file segment **Note:** Use the **filename** argument with caution. • **filtercat** = [off exclusive kernel shared unused unattached] The **filtercat** argument specifies that the **svmon** command filters the segments by category. – **Default value**: off – **Scope**: Command report (**-C**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**) You can specify the following values to the **filtercat** option to filter the segments by category: **kernel** Filters the kernel segments. **exclusive** Filters the exclusive segments. The exclusive segments are used by only one process, except the shared-memory segments that are always reported as either shared or unattached. **shared** Filters the shared segments. The shared segments are used by more than one process, or shared-memory segments used by at least one process. **unused** Filters the unused segments. The unused segments are not used by any processes. **unattached** Filters the unused shared-memory segments. The unattached segments are shared-memory segments that are not used by any process. **off** Deactivates the filter. The **off** option is the same as the command **-O filtercat = "kernel exclusive shared unused"**. **Note:** The **filtercat** option changes the value of the reported basic metrics in the summary header because it adds or removes segments from the report.

Item	Description
options	*(Continued description of the valid values for the options parameter).* • **filterpgsz** = [off s m L S] The **filterpgsz** argument specifies that the **svmon** command filters the segments by page size. – **Default value**: off – **Scope**: Command report (**-C**), detailed segment report (**-D**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**) You can specify the following values to the **filterpgsz** option to filter the segments by page size: Filters the segments that are 4 KB (small) in page size **m** Filters the segments that are 64 KB (medium) in page size **L** Filters the segments that are 16 MB (large) in page size **S** Filters the segments that are 16 GB (supreme) in page size **off** Deactivates the **filterpgsz** option **Note:** The **filterpgsz** argument changes the values of the reported metrics in the summary header, because it adds or removes segments from the report. To filter segments of different page sizes, you can specify various parameters in the form of <*min_size*><*max_size*>. For example, to filter the segments with small page size and the segments with small and medium page sizes, enter the following command: `svmon -O filterpgsz="sm s"` • **filterprop** = [off notempty data text] The **filterprop** argument specifies that the **svmon** command filters the segments report by property. – **Default value**: off – **Scope**: Command report (**-C**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**) You can specify the following values to the **filterprop** option to filter the segments by property: **notempty** Filters the segments with value that is in use and is not equal to zero **data** Filters the data segments, which are computational **text** Filters the text segments, which are not computational **off** Deactivates the **filterprop** option **Note:** The **filterprop** argument changes the value of the reported basic metrics in the summary header because it adds or removes segments from the report.

Item	Description
options	*(Continued description of the valid values for the options parameter).* • **filtertype** = [off working persistent client] The **filtertype** argument specifies that the **svmon** command filters the segments by type. – **Default value**: off – **Scope**: Command report (**-C**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**) You can specify the following values to the **filtertype** option to filter the segments by type: **working** Filters the working segments **persistent** Filters the persistent segments, such as the segments on journaled file system (JFS) **client** Filters the client segments, such as the segments on enhance journaled file system (JFS2) or network file system (NFS) **off** Deactivates the **filtertype** option, which is the same as the **-O filtertype = "working persistent client"** command **Note:** The **filtertype** argument changes the value of the reported basic metrics in the summary header, because it adds or removes segments from the report. • **format** = [80 | 160 | nolimit] The **format** argument specifies the maximum width, in characters, for the output of the **svmon** command. – **Default value**: 80 – **Scope**: Command report (**-C**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**) You can specify the following values to the **format** option: **80** Limits the width of the output to 80 characters. In a process report, some fields are truncated. In a segment report, some fields are displayed on separate lines. **160** Limits the width of the output to 160 characters. In a process report, some fields are truncated. In a segment report, some fields are displayed on separate lines. **nolimit** Does not limit the width in character. Does not truncate fields or display them in separate lines. Some columns of the report might be shifted. **Tip:** You can use the **summary** argument to force the value of the **format** option to 160 characters. • **frame** = [on | off] The **frame** argument specifies that the **svmon** command displays the information per frame. – **Default value**: off – **Scope**: Detailed segment report (**-D**) You can specify the following values to the **frame** option: **on** Displays the information per frame **off** Displays the report automatically

Item	Description
options	*(Continued description of the valid values for the options parameter).* • **mapping** = [on | off] The **mapping** argument specifies that the **svmon** command displays the source segments that are associated with the segments that are created by the **mmap** subroutine (also known as the **mmap** segments). When the source segments do not pertain to the process address space and the **mapping = on** value is specified, the source segments are integrated into the report and are flagged with an asterisk (*). – **Default value**: off – **Scope**: Command report (**-C**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**) You can specify the following values to the **mapping** option: **on** Displays the source segments that are associated to the segments created by the **mmap** subroutine **off** Does not display the source segments that are associated with the segments created by the **mmap** subroutine **Note:** The **mapping** argument changes the values of the reported metrics in the summary header because it adds or removes segments from the report. • **maxbufsize=*size*[KB | MB | GB]** The **maxbufsize** argument modifies the memory buffer size to store the data related to the segment identifiers. When you run the **svmon** command with the **-P** flag and when the process has more segments, the svm (snapshot of virtual memory) process might fail because of less buffer size. In such cases, use the **-O maxbufsize** flag to increase the buffer size. The **maxbufsize** argument overrides the default buffer size value and uses the specified value. In AIX 7.2.4, or later, the default buffer size is 2 MB. In AIX 7.2.3, or earlier, the default buffer size 512 KB. • **mpss** = [on | off] The **mpss** argument breaks down the value of the mixed page size segment into individual page sizes. – **Default value**: off – **Scope**: Command report (**-C**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**) You can specify the following values to the **mpss** option: **on** Breaks down the value of the mixed page size segment into individual page sizes **off** Does not break down the value of the mixed page size segment • **overwrite** = [on | off] The **overwrite** argument overwrites the XML file that the **svmon** command produced. – **Default value**: on – **Scope**: XML report (**-X**) You can specify the following values to the **overwrite** option: **on** Overwrites the XML file that the **svmon** command generated **off** Does not overwrite the XML file

Item	Description
options	*(Continued description of the valid values for the options parameter).* • **pgsz** = [on | off] The **pgsz** argument specifies that the **svmon** command displays the sections per page size. – **Default value**: off – **Scope**: Command report (**-C**), process report (**-P**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**) You can specify the following values to the **pgsz** option: **on** Displays the sections per page size **off** Displays the report automatically • **pidlist** = [on | number | off] The **pidlist** argument specifies that the **svmon** command displays a list of process IDs (PIDs) or the number of different PIDs for each segment. – **Default value**: off – **Scope**: Command report (**-C**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**) You can specify the following values to the **filename** option: **on** Displays a list of process IDs for each segment. For special segments, a label is displayed instead a list of process IDs. The following labels are displayed: – **System segment**: Labels the segments that are flagged as system segments – **Unused segment**: Labels the segments that are not used by any existing processes. For example, persistent segments that are relative to the files that are no longer in use. – **Unattached segment**: Labels the shared-memory segments that are not used by any existing processes. – **Shared-library text**: Labels the segments that contain a shared library. The shared library can be used by most of the processes. This label prevents the display of a long list of processes. **number** Displays the number of different process IDs for each segment. **off** Does not displays the list or number of process IDs for each segment.
options	*(Continued description of the valid values for the options parameter).* • **process** = [on | off] The **process** argument specifies that the **svmon** command displays the list of the processes that belong to the entity. – **Default value**: off – **Scope**: Command report (**-C**), user report (**-U**), and workload management class report (**-W**) You can specify the following values to the **process** option: **on** Displays the list of the processes that belong to the entity **off** Does not display the list of processes that belong to the entity • **range** = [on | off] The **range** argument specifies that the **svmon** command displays the ranges of pages within the segments that have been allocated. – **Default value**: off – **Scope**: Command report (**-C**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**) You can specify the following values to the **range** option: **on** Displays the ranges of pages within the allocated segments **off** Does not display the ranges of pages within the allocated segments

Item	Description

- **segment** = [on | category | off]

  The **segment** argument specifies that the **svmon** command displays the segment statistics for entities.

  – **Default value**: off

  – **Scope**: Command report (**-C**), process report (**-P**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**)

  You can specify the following values to the **segment** option:

  on
  Displays a unique segment list. The segments are sorted by the values of the **sortseg** argument.

  category
  Groups the segments in three categories: system, exclusive, and shared. The segments in each category are sorted by the values of the **sortseg** argument.

  off
  Does not display the segment lists.

- **shmid** = [on | off]

  The **shmid** argument displays the shared-memory ID that is associated with a shared-memory segment.

  Restriction: The **shmid** argument cannot work with a workload partition.

  – **Default value**: off

  – **Scope**: Command report (**-C**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**)

  You can specify the following values to the **shmid** option:

  on
  Displays the shared-memory ID associated to a shared-memory segment

  off
  Does not display the shared-memory ID associated to a shared-memory segment

  Note: Use the **shmid** argument with caution.

Item	Description
options	*(Continued description of the valid values for the options parameter).* • **sortentity** = [inuse | pin | pgsp | virtual] The **sortentity** argument specifies the method for the **svmon** command in sorting the reports. – **Default value**: inuse – **Scope**: Command report (**-C**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W** You can specify the following values to the **sortentity** option to sort the reports: **inuse** Sorts the reports in decreasing order of real memory consumption **pin** Sorts the reports in decreasing order of pinned memory consumption **pgsp** Sorts the reports in decreasing order of paging space consumption **virtual** Sorts the reports in decreasing order of virtual memory consumption • **sortseg** = [inuse | pin | pgsp | virtual] The **sortseg** argument specifies the method for the **svmon** command in sorting the segment reports. – **Default value**: inuse – **Scope**: Command report (**-C**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**) You can specify the following values to the **sortseg** option to sort the segment reports: **inuse** Sorts the segments in decreasing order of real memory consumption **pin** Sorts the segments in decreasing order of pinned memory consumption **pgsp** Sorts the segments in decreasing order of paging space consumption **virtual** Sorts the segments in decreasing order of virtual memory consumption • **subclass** = [on | off] The **subclass** specifies that the **svmon** command displays the statistics of memory use for the subclass of the workload management classes. – **Default value**: off – **Scope**: Workload management tier report (**-T**) and workload management class report (**-W**) You can specify the following values to the **subclass** options: **on** Displays the statistics of memory use of the workload management classes' subclasses **off** Does not display the statistics of memory use of the workload management classes' subclasses

Item	Description
options	*(Continued description of the valid values for the options parameter).* • **summary** = [basic | longreal | ame | longame] The **summary** argument specifies the format to display the summary for the **svmon** command. – **Default value**: basic – **Scope**: Command report (**-C**), global report (**-G**), process report (**-P**), user report (**-U**), and workload management class report (**-W**) summary = [ame | longame] is available only with global report **(-G).** You can specify the following values to the **summary** option: **basic** Displays the basic headers for the **svmon** command **longreal** Displays the real memory information in a long format (160 columns per line). **Note:** The **summary** argument with the value of `longreal` is supported along with the **-G** flag only. **ame** Displays the Active Memory Expansion information (in an Active Memory Expansion enabled system). **longame** Displays the Active Memory Expansion information (in an Active Memory Expansion enabled system) in a long format. • **svmonalloc** = [on | off] The **svmonalloc** argument specifies that the **svmon** command displays the maximum size of the memory that it dynamically allocated during its processing. – **Default value**: off – **Scope**: All reports You can specify the following values to the **svmonalloc** options: **on** Displays the maximum size of the allocated memory **off** Does not display the maximum size of the allocated memory • **threadaffinity**= [on | off] The **threadaffinity** argument specifies that the **svmon** command displays the home SRADIDs (Scheduler Resource Allocation Domain Identifier) and the thread SRAD (Scheduler Resource Allocation Domain) affinity statistics for the threads of a process. – **Default value**: off – **Scope**: Process report (**-P**) You can specify the following values to the **threadaffinity** option: **on** Displays the home SRADIDs and thread SRAD affinity statistics for the threads of a process. **off** Does not display the home SRADIDs and thread SRAD affinity statistics for the threads of a process. • **timestamp** = [on | off] The **timestamp** argument specifies that the **svmon** command displays the timestamp at the beginning of the report. – **Default value**: off – **Scope**: Command report (**-C**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**) You can specify the following values to the **timestamp** option: **on** Displays the time stamp at the beginning of the report **off** Does not display the time stamp at the beginning of the report

Item	Description
options	*(Continued description of the valid values for the options parameter).* • **tmem** = [on | off] The **tmem** argument specifies the **svmon** command to append the true memory details. – **Default value**: on – **Scope**: Global report **(-G).** You can specify the following values to the **tmem** option. **on** Displays the true memory information at the end of the report **off** Does not display the true memory information. **Note:** The summary argument must have the value of ame. • **unit** = [auto | page | KB | MB | GB | TB] The **unit** argument modifies the metrics unit of the report. – **Default value**: page – **Scope**: Command report (**-C**), process report (**-P**), segment report (**-S**), workload management tier report (**-T**), user report (**-U**), and workload management class report (**-W**) You can specify the following values to the **unit** option: **auto** Expresses the values in the most appropriate unit with at most three significant digits. The unit used in the report is specified for each metric. **page** Expresses the values in 4 KB page units. The unit used in the report is specified in the report header. **KB** Expresses the values in kilobytes (KB) **MB** Expresses the values in megabytes (MB) **GB** Expresses the values in gigabytes (GB) **TB** Expresses the values in terabytes (TB) **Tip:** To overwrite the default values that are defined previously by the **-O** *options* flag, you can define the **.svmonrc** configuration file in the directory where the **svmon** command is launched.
count	Specifies the top object to be printed. Use the *count* parameter with the **-T** flag.
interval	Specifies the interval for the **svmon** command to collect and print statistics. Use the *interval* parameter with the **-i** flag.
numintervals	Specifies the number of repetitions for the **svmon** command to collect and print statistics when the *interval* parameter is specified. Use the *numintervals* parameter with the **-i** *interval* option. If the *numintervals* parameter is not specified, the **svmon** command runs until you interrupt it (Ctrl+C).
ALL	Specifies that the **-@** flag displays the report for all of the WPAR starting with the global report, and then process all of the available WPAR, sorting them by the WPAR name.
wparnames	Specifies the workload partitions whose information is to be displayed. When you specify the **-@** *wparnames* option, all of the information displayed is restricted to the WPAR that the *wparnames* parameter specifies, and has meaning only inside the WPAR. Each WPAR name in the list is processed in the given order and each svmon report is separated by the WPARname header.
sids	Specifies the segment IDs (SIDs). The SIDs must be primary segments.
pids	Specifies the process IDs (PIDs). The value of the *pids* parameter is a decimal value. If you do not supply any list of process IDs (PIDs), the statistics of memory use are displayed for all active processes. Use the *pids* parameter with the **-P** flag.
lognames	Specifies the login names. The value of the *lognames* parameter is a string. It is an exact login name. If you do not specify any lists of login identifiers, the statistics of the memory use are displayed for all of the defined login identifiers. Use the *lognames* parameter with the **-U** flag.
classnames	Specifies the Workload Manager class. The value of the *classnames* parameter is a string. It is the exact name of a class. For a subclass, the name should be in the form *superclassname.subclassname*.
tiers	Specifies a tier number for the classes. If you do not specify a list of tiers, the statistics of memory use are displayed for all of the defined tiers. Use the *tiers* parameter with the **-T** flag.
supclassname	Specifies the name of the superclass that the subclasses are restricted to. You cannot specify a list of classes for this flag.

Item	Description
filename	Specifies the name of the output file. It is an alpha-numeric string. The suffix of the output file name is **.svm**. It is automatically added to the file name if you do not specify the suffix. Use the *filename* parameter with the **-o** flag and the **-X** flag.
comment	Specifies the string to add in the <CollectionHeader><Comment> tag of the XML report. Use the *comment* parameter with the **-X** flag and the **-c** flag.

Security

Any user can run the **svmon** command. If the user is not a root user, the view will be limited to the user's own processes.

If RBAC is activated and the **aix.system.stat** role that is attributed to the user, the user can see the same view that the root user does.

Examples

1. To display global statistics in a one line format every minute for 30 minutes, enter the following command:

```
# svmon -G -O summary=longreal -i 60 30
```

2. To display global statics with automatic unit selection, a time stamp, per page size data, and detailed affinity information, enter the following command:

```
# svmon -G -O unit=auto,timestamp=on,pgsz=on,affinity=detail
```

3. To display global statistics for the system and all of its WPAR in a compact format, enter the following command:

```
# svmon -G -O summary=longreal -@ ALL
```

4. To display the memory consumption in megabytes (MB) of all processes in a compact report, enter the following command:

```
# svmon -P -O summary=basic,unit=MB
```

5. To display the memory consumption of all processes according to the number of virtual pages, and sort the segments for each process by the number of pages in the paging space, enter the following command:

```
# svmon -P -O segment=on,sortentity=virtual,sortseg=pgsp
```

6. To display the memory consumption of process 123456 in full detail, enter the following command:

```
# svmon -P 123456 -O
segment=on,pidlist=on,range=on,mapping=on,shmid=on,filename=on,affinity=detail
```

7. To display the top 10 system segments sorted by the number of pages in real memory, enter the following command:

```
# svmon -S -t 10 -O filtercat=kernel,sortseg=inuse
```

8. To display all of the segments that are not attached to a process, enter the following command:

```
# svmon -S -O filtercat=unattached
```

9. To display only 16 MB segments with their address ranges, enter the following command:

```
# svmon -S -O filterpgsz=L -O range=on
```

10. In the global WPAR, to display the WPAR name that each segment belongs to, enter the following command:

```
# svmon -S -@
```

11. To display the memory consumption of all Oracle processes in a compact report for only the shared segments, enter the following command:

```
# svmon -C oracle -O summary=basic,filtercat=shared
```

12. To display the top 10 users running the processes that consume the most memory every minute, enter the following command:

```
# svmon -U -t 10 -O summary=basic -i 60
```

13. To display the memory use for the `Mysupclass` superclass with its subclasses, enter the following command:

```
# svmon -W Mysupclass -O subclass=on
```

14. To display the memory use for the `0` tier subclasses of the `Mysupclass` superclass, enter the following command:

```
# svmon -T 0 -a Mysupclass
```

15. To display the frames that belong to the `36cfb` segment with frame level details, enter the following command:

```
# svmon -D 36cfb -O frame=on
```

16. To generate an XML report in the **lpar01.svm** file, enter the following command:

```
# svmon -X -o lpar01.svm
# svmon -X -o lpar01
```

17. To generate an XML report with affinity domain details, enter the following command:

```
# svmon -X -o lpar_affinity -O affinity=on
```

18. To generate an XML report with affinity domain details at the segment level, enter the following command:

```
# svmon -X -o lpar_affinitydet -O affinity=detail
```

19. To display global statistics with memory compression details along with true memory snapshot at the end, enter the following command:

```
# svmon -G -O summary=ame
```

20. To display global statistics with memory compression details with true memory details turned-off, enter the following command

```
# svmon -G -O summary=ame,tmem=off
```

21. To display global statistics with Active Memory Expansion details (in an Active Memory Expansion enabled system) in a one line format, enter the following command

```
# svmon -G -O summary=longame
```

22. To display the home SRADIDs and thread SRAD affinity statistics for the threads of a process, enter:

```
# svmon -P 1 -O threadaffinity=on
```

swap Command

Purpose

Provides a paging space administrative interface.

Syntax

swap [**-a** *device*] | [**-d** *device*] | [**-s**] | [**-l**]

Description

The functions provided by the swap command are display of characteristics, addition of paging space and removal of paging space.

Flags

Item	Description
-a *device*	Activates the paging space. Performs the same function the **swapon** command.
-d *device*	Deactivates the paging space. Performs the same function as the **swapoff** command.
-l	Lists the status of paging space areas in a list form. The output has 4 columns, containing the following information: **device** Path name of the page space. **maj/min** The major/minor device number for the device. **total** Total size in megabytes for the area. **free** Amount of available space.
-s	Prints summary information about total paging space usage and availability. The following information is displayed in the output (amounts of paging space are listed in 4K byte blocks). **allocated** Total amount of paging space area currently allocated. **used** Total amount of paging space area currently being used. **available** Total amount of free paging space. These numbers include paging spaces from all configured areas as listed by the **-l** option on active paging space. **Note:** There is a paging space limit of 64 GB per device.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To print summary information on total paging space, enter:

```
swap -s
```

2. To list the status of the paging space areas in a list form, enter:

```
swap -l
```

3. To activate a particular paging space device paging01, enter:

```
swap -a /dev/paging01
```

Files

Item	Description
/usr/sbin/swap	Contains the System V **swap** command.

swapoff Command

Purpose

Deactivates one or more paging spaces.

Syntax

swapoff *DeviceName* { *DeviceName* ...}

Description

The **swapoff** command deactivates one or more paging spaces. The paging spaces are specified by *DeviceName*.

Note: There is a paging space limit of 64 GB per device.

To be deactivated:

- The paging space must have been previously activated through the **swapon** command.
- There must exist enough space in the remaining paging spaces. The remaining paging device should have enough space to accommodate the current system-wide paging space usage and the **npswarn** value.

Note: This command is not supported when executed within a workload partition.

Exit Status

Item	Description
Value	**Description**
0	Deactivation is successful, the paging state is set to the INACTIVE state.

Item	Description
1	The following message displays: `swapoff: Cannot deactivate paging space DeviceName`
2	There is not enough space in the remaining paging spaces, the deactivation is not done and the following message displays: `"swapoff: Cannot deactivate paging space DeviceName :` `There is not enough space in the file system."`
3	An I/O error occurred on user pages of a paging space, the following message displays: `swapoff: Deactivation of paging space DeviceName suspended:` `I/O errors encountered on user backing pages.` The recommended action is: • Check the error log. • Deactivate the paging space for the next reboot using the **chps** command. • Reboot the system.
4	An I/O error occurred on system pages of a paging space, the following message displays: `swapoff: Deactivation of paging space DeviceName suspended:` `I/O errors encountered on system backing pages. The system may crash.` The recommended action is: • Check the error log. • Deactivate the paging space for the next reboot using the **chps** command. • Reboot the system.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

swapon Command

Purpose

Activates a paging space.

Syntax

swapon -a | *devicename*

Description

The **swapon** command activates a paging space. It is used during early system initialization to make the initial paging space available. During a later phase of system initialization, the **swapon -a** command is used to make other devices available so that paging and swapping activity is interleaved across several devices. If the option **auto=yes** then the **swapon -a** command makes all devices specified in the **/etc/swapspaces** available that aren't explicitly excluded from being automatically swapped on by

their stanza. Calls to the **swapon** command normally occur in the system multiuser initialization **/etc/rc** file.

The *devicename* parameter specifies a specific device to be made available. The second form gives individual block devices as given in the system swap configuration table. The call makes this space and other defined spaces available to the system for paging and swap allocation. The system swap configuration table is the set of all devices specified in the **/etc/swapspaces** file.

Note: The maximum number of active paging spaces is 16. In addition, there is a paging space limit of 64 GB per device.

Note: This command is not supported when executed within a workload partition.

Flags

Item	Description
-a	Causes all devices present in the **/etc/swapspaces** file to be made available.

Security

The Role Based Access Control (RBAC) Environment and Trusted AIX: This command implements and can perform privileged operations. Only privileged users can execute such privileged operations.

To review the list of privileges and the authorizations associated with this command, refer to the **/etc/security/privcmds** database.

Examples

1. To cause all devices present in the **/etc/swapspaces** file to be made available, enter:

   ```
   swapon -a
   ```

   All devices present in the **/etc/swapspaces** file are now available.
2. To cause the **/dev/paging03** and **/dev/paging04** devices to be available for paging and swapping, enter:

   ```
   swapon /dev/paging03 /dev/paging04
   ```

   The **/dev/paging03** and **/dev/paging04** devices are now available.

Files

Item	Description
/etc/rc	System multiuser initialization
/dev/paging	Device entries for paging/swap space
/etc/swapspaces	Contains a list of swap devices.

swcons Command

Purpose

Redirects, temporarily, the system console output to a specified device or file.

Syntax

swcons [**-p** *Log_File*] [**-s** *Log_Size*] [**-t** *Tag_Verbosity*] [**-v** *Log_Verbosity*] *PathName*

Description

The **swcons** command temporarily switches the system console output to a different target during system operation. This command only switches system informational-, error-, and intervention-required message output to the specified destination. The **swcons** command does not affect the operation of the system console device that is providing a login by way of the **getty** command.

The device or file specified when using this command remains the target for console output until changed by another **swcons** command, until the next start of the system, or until the console driver detects an error when accessing the designated device or file. If an open or write error is detected on the device or file specified by the **swcons** command, the console device driver switches all output back to the device or file that provided console support when the system was last started.

The *PathName* parameter must be a fully qualified path name to a device or file that is to receive system console message output. If the *PathName* parameter specifies a file that does not exist, the **swcons** command creates the file. If the file does exist, the **swcons** command appends any new console message output to the contents of the file.

> **Attention:** Use of the **swcons** command to switch console output to an NFS mounted file system or a diskless/dataless client might cause the operating system to hang.

Flags

Item	Description
-p *Log_File*	Specifies the full path name to use for the console output log file.
-s *Log_Size*	Specifies the size, in bytes, of the console output log file.
-t *Tag_Verbosity*	Specifies the verbosity level for console output tagging. Zero disables tagging; 1 through 9 enable tagging. For additional information about console output logging and tagging, see the **console Special File** in the *Files Reference* book.
-v *Log_Verbosity*	Specifies the verbosity level for console output logging. Zero disables logging; 1 through 9 enable logging.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To change the system console message output to a file called `console.out` in the `/tmp` directory, enter:

   ```
   swcons /tmp/console.out
   ```

2. To change the system console message output to a terminal with the logical name `tty3`, enter:

   ```
   swcons /dev/tty3
   ```

3. To change the system-console message output back to the device or file that supported the console output at system start time, enter:

   ```
   swcons
   ```

Files

Item	Description
/dev/console	Specifies the special file for system console access.
/usr/sbin/swcons	Contains the **swcons** command file.

swrole Command

Purpose

Switches to a specified role session.

Syntax

swrole { **ALL** | *Role* [,*Role*] ... } [*Argument* ...]

Description

The **swrole** command creates a new role session with the roles that is specified by the *Role* parameter. The *Role* parameter must be composed of the names of roles in the **roles** attribute of the user. Before creating a new role session, the **swrole** command performs authentication according to the **auth_mode** attribute of the **chrole** command for the specified roles. If any of the specified roles requires authentication, the user must be successfully authenticated for the action to be performed. If none of the specified roles require authentication, no authentication is requested.

The **swrole** command creates a new role session with the specified roles added to the active role set of the session. The **ALL** keyword specifies that a role session is created with all the roles that are assigned to the user. Role sessions are limited to eight roles per session. If a user has more than eight roles, only the first eight roles are assigned to the role session when the **ALL** keyword is specified. Creation of a new role session preserves the user environment for the current session.

Any argument, such as a flag or a parameter, which is specified by the *Arguments* parameter, must relate to the login shell that is defined for the user. The arguments are passed to the login shell that is created for the role session. For example, if the login shell for a user is **/usr/bin/ksh**, any of the flags that are allowed for the **ksh** command can be specified.

To restore the previous session, type `exit` or press the Ctrl-D. The action ends the shell created by the **swrole** command and returns the user to the previous shell and environment.

Each time the **swrole** command is run, an entry is made in the **/var/adm/rolelog** file. The **/var/adm/rolelog** file records the following information: date, time, system name, login name and role name. The **/var/adm/rolelog** file also records whether or not the role initiation attempt is successful: a plus sign (+) indicates a successful role initiation, and a minus sign (-) indicates an unsuccessful role initiation.

The **swrole** command is functional only when the system is operating in enhanced Role Based Access Control (RBAC) mode. If the system is not in enhanced RBAC mode, the command displays an error message and returns failure.

Examples

1. To assume the `RoleAdmin` and `FSAdmin` roles as a user who has been assigned the roles, enter the following command:

   ```
   swrole RoleAdmin,FSAdmin
   ```

2. To run the **backup** command as a role that has the appropriate authorization, enter the following command:

   ```
   swrole FSAdmin "-c /usr/sbin/backup -9 -u"
   ```

swts Command

Purpose

Switches a thin server to a different COSI.

Syntax

swts -c *Image* [**-n** |**-t** *Time*] [**-v**] *ThinServer*

Description

The `swts` command switches a thin server to a different Common Operating System Image (COSI). If specified with the `-t` flag, the thin server switches to a new common image at the time specified by the *Time* parameter. The value for *Time* must be a valid cron tab entry. Refer to the `crontab` command for creating valid cron time entries.

The `swts` command can be run on either a NIM master or a thin server. When a thin server is switched to a new common image, files in the `/inst_root` directory for the thin server will be synced with the new common image.

Flags

Item	Description
`-c` *Image*	Specifies the common image that the thin server switches to.
-n	Specifies option to allow a thin server to switch to a new common OS image that was setup by the NIM administrator with the **-c** flag. The user running from the thin server will only need to execute the **swts** command without any argument to switch the common OS images.
`-t` *Time*	Specifies a cron entry that allows thin servers to be switched over at a more convenient time.
`-v`	Enables verbose debug output when the `swts` command runs.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

Access Control: You must have root authority to run the `swts` command.

Examples

1. To switch the `cosi1` common image of a thin server named `lobo` to a common image named `cosi2`, enter:

```
swts -c cosi2 lobo
```

The `lobo` thin server is re-initialized and `cosi2` is its new operating system.

2. To switch the `cosi1` common image of a thin server named `lobo` to a common image named `cosi2` at midnight on Sunday, December 25, enter:

```
swts -c cosi2 -t "0 0 25 12 0" lobo
```

The `lobo` thin server will continue to use the `cosi1` common image until midnight on Sunday, December 25, when it switches to `cosi2`.

Location

`/usr/sbin/swts`

Files

Item	Description
`/etc/niminfo`	Contains variables used by NIM.

sync Command

Purpose

Updates the i-node table and writes buffered files to the hard disk.

Syntax

sync

Description

The **sync** command runs the **sync** subroutine. If the system must be stopped, run the **sync** command to ensure file system integrity. The **sync** command writes all unwritten system buffers to disk including modified i-nodes, delayed block I/O, and read-write mapped files.

> **Note:** The writing, although scheduled, is not necessarily complete upon return from the **sync** subroutine.

synclvodm Command

Purpose

Rebuilds the logical volume control block, the device configuration database, and the device special files.

Syntax

synclvodm [**-c** | **-D** | **-F** | **-k** | **-K** | **-P** | **-R** | **-v**] *VolumeGroup LogicalVolume ...*

Description

The **synclvodm** command rebuilds the logical volume control block, the device configuration database, and the device special files (for the volume group and logical volumes), so that they are synchronized with the volume group descriptor areas on the physical volumes.

During normal operations, the device configuration database remains consistent with the logical volume manager information in the logical volume control blocks and the volume group descriptor areas on the physical volumes. If for some reason the device configuration database is not consistent with Logical Volume Manager information, the **synclvodm** command can be used to resynchronize the database. The

volume group must be active for the resynchronization to occur (see **varyonvg**). If logical volume names are specified, only the information related to those logical volumes is updated. If logical volume names are not specified, every logical volume in the volume group is updated.

> **Attention:** Do not remove the **/dev** entries for volume groups or logical volumes. Do not change the device configuration database entries for volume groups or logical volumes using the object data manager.
>
> **Note:** To use this command, you must either have root user authority or be a member of the **system** group.

Flags

Item	Description
-c	Treats naming conflicts as fatal errors. If this flag is not specified, the command generates a warning message for any naming conflicts, and automatically renames the logical volume by default. A logical volume naming conflict occurs when the logical volume name is already in use by another device. A volume group naming conflict occurs when the volume group major number cannot be reserved in the device configuration database.
-D	Does not remove or recreate the logical volume minor numbers and device special files. If not specified, the command removes and recreates the logical volume minor numbers and device special files by default.
-F	Does not synchronize the device configuration database entries for the physical volumes in the volume group. If this flag is not specified, the command removes the device configuration database entries for all physical volumes in the volume group, and recreates those entries based on the information in the volume group descriptor area by default.
-k	Takes the volume group lock when the **synclvodm** command is running. If this flag is not specified, the volume group lock is taken only if the parent process does not have the lock.
-K	Does not take the volume group lock when the **synclvodm** command is running. Use this flag when the caller is a shell script, and is managing the volume group lock in the shell script with the **putlvodm -k** and **-K** flags. The default behavior is to take the volume group lock unless the parent process has the lock.
-P	Preserves the permission bits for the special files of logical volume device. The **-P** flag overrides the **-D** flag. The **-P** flag is ignored for original type volume groups. If this flag is not set, the ownership of the logical volume special file is set to `root`, and the group is set to `system`.
-R	Restores the user, group, and permissions for the logical volume device special files to the values previously set by the **mklv** and **chlv** commands using the **-U**, **-G**, and **-P** flags. The **-R** flag is ignored for original type volume groups, or when the **-D** flag is specified.
-v	Displays the output from the **synclvodm** command in verbose mode.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Examples

To synchronize the device configuration database with the logical volume manager information for rootvg, enter the following:

```
synclvodm rootvg
```

Files

Item	Description
/usr/sbin/synclvodm	Contains the **synclvodm** command.

syncroot Command

Purpose

Synchronizes a non-shared portion of installed software with a shared part.

Syntax

/usr/sbin/syncroot [[-a] [-i] | [-F] [-r]] [-p] [-v] [-X]

Flags

Item	Description
-a	Performs additional installation only. Does not downlevel the **installp** file sets (that is, uninstall, reject, force overwrite). Not valid with the -r flag.
-i	Only updates **installp** file sets. Not valid with the -r flag.
-F	Forces copy **RPM** files. Not valid with the -i flag.
-r	Only updates **RPM** files. Not valid with the -i flag.
-p	Previews operation. Do not actually performs the synchronization.
-v	Specifies the verbose mode.
-X	Expands file systems if necessary and possible.

Notes:

- If you are logged into a version 6 workload partition, on a version 7 global system, and run the **syncroot** command, the operation will fail with the following error:

```
syncroot: Processing root part installation status.
Your global system is at a higher version than the WPAR.
Please log out of the WPAR and execute the migwpar command.
syncroot: Returns Status = FAILURE
```

- For shared workload partitions (WPARs), if the installation history of the source system is different from the installation history of the target system, the **restwpar** command and the **syncroot** command might fail for few filesets. You might see a failure message that is similar to the following example for the **syncroot** command, at the end of the restwpar operation:

```
syncroot: Error synchronizing installp software
syncroot: Returns Status = FAILURE
```

  You must restore or migrate the shared WPAR to a logical partition (LPAR) that has the installation history similar to the installation history of the source LPAR.

Security

Access Control: Only the root user can run this command.

Examples

1. To update all **installp** filesets in the root part, enter:

```
# syncroot -i
```

2. To perform an update of all **RPM** files and expand space automatically (if needed and possible), enter:

```
# syncroot -r -X
```

syncvg Command

Purpose

Synchronizes logical volume copies that are not current.

Syntax

syncvg [**-f**] [**-i**] [**-H**] [**-P** *NumParallelLps*] { **-l** | **-p** | **-v** } *Name* ...{ [**-a** { *all* | *pid1,pid2,...* }] [**-r** { *all* | *pid1,pid2,...* }] [**-t** { *all* | *pid1,pid2,...* }] [**-n** *vgName*] [**-T** *SyncRate* [-d { all | *pid1,pid2,...* }]] [**-q**] [**-Q**] }

Description

The **syncvg** command synchronizes the physical partitions, which are copies of the original physical partition, that are not current. The **syncvg** command can be used with logical volumes, physical volumes, or volume groups, with the *Name* parameter representing the logical volume name, physical volume name, or volume group name. The synchronization process can be time consuming, depending on the hardware characteristics and the amount of data.

When the **-f** flag is used, a good physical copy is chosen and propagated to all other copies of the logical partition, whether or not they are stale. Using this flag is necessary in cases where the logical volume does not have the mirror write consistency recovery.

Unless disabled, the copies within a volume group are synchronized automatically when the volume group is activated by the **varyonvg** command.

Note: For the **syncvg** command to be successful, at least one good copy of the logical volume should be accessible, and the physical volumes that contains this copy should be in ACTIVE state. If the **-f** option is used, the above condition applies to all mirror copies.

If the **-P** option is not specified, **syncvg** will check for the *NUM_PARALLEL_LPS* environment variable. The value of *NUM_PARALLEL_LPS* will be used to set the number of logical partitions to be synchronized in parallel.

Flags

Item	Description
-a { *all* | *pid1,pid2,...* }	Pauses one or more sync operations. The following parameters can be passed to this option: **all** Pause all sync operations. **pid1,pid2,...** A comma separated list of process ID (PID) to pause.
-f	Specifies a good physical copy is chosen and propagated to all other copies of the logical partition, whether or not they are stale.

Item	Description
-H	Postpones writes for this volume group on other active concurrent cluster nodes until this sync operation is complete. When using the **-H** flag, the **-P** flag does not require that all the nodes on the cluster support the **-P** flag. This flag is ignored if the volume group is not varied on in concurrent mode.
-i	Reads the names from standard input.
-l	Specifies that the *Name* parameter represents a logical volume device name.
-n *vgName*	Manages sync operations for a specific volume group. This option is only valid with the **-a**, **-r**, **-t** , **-q** and **-Q** options. **vgName** Volume group name.
-p	Specifies that the *Name* parameter represents a physical volume device name.
-P *NumParallelLps*	Numbers of logical partitions to be synchronized in parallel. The valid range for *NumParallelLps* is 1 to 32. *NumParallelLps* must be tailored to the machine, disks in the volume group, system resources, and volume group mode. When a volume group is varied on in concurrent mode, all other cluster nodes that have this volume group varied must be at least AIX 4.3.0, otherwise **syncvg** will ignore this option and continue. **Note:** See Description above for more information.
[**-q**]	Queries sync operations. A verbose list of sync operation Process Identifiers (PIDs) is returned. This flag also outputs the sync rate for each sync operation. If the *SyncRate* option is not specified using the **-T** flag, this flag displays the current sync rate of the sync operation.
[**-Q**]	Queries sync operations. A comma separated list of sync operation PIDs is returned. This flag also returns the sync rate for each sync operation. If the *SyncRate* option is not specified using the **-T** flag, this flag displays the current sync rate of the sync operation.
{**-r** *all* | *pid1,pid2,...* }	Resumes one or more sync. The following parameters can be passed to this option: **all** Resumes all sync operations. **pid1,pid2,...** A comma separated list of PIDs to resume.
{**-t** *all* | *pid1,pid2,...* }	Terminates one or more sync. The following parameters can be passed to this option: **all** Terminates all sync operations. **pid1,pid2,...** A comma separated list of PIDs to terminate.

Item	Description
[**-T** *SyncRate* [-d { all | *pid1,pid2,...* }]]	Throttles the sync rate of the current sync operation or throttles one or more sync operations that are in progress. The following parameters can be passed to this option: **SyncRate** Specifies the sync rate, in MB/sec, to throttle. The **syncvg** command synchronizes one Logical Track Group (LTG) at a time. This parameter must be specified in multiples of the LTG size of the volume group. If the **SyncRate** parameter is not specified in the multiples of the LTG size, the **syncvg** command rounds up to the nearest LTG size of the volume group. If you do not specify the **-d** flag, the **syncvg** command throttles the sync rate of the current sync operation. **-d all** Throttles the sync rate for all the sync operations that are in progress. **-d pid1,pid2,...** A comma-separated list of PIDs that throttle the sync rate.
-v	Specifies that the *Name* parameter represents a volume group device name.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To synchronize the copies on physical volumes hdisk4 and hdisk5, enter:

```
syncvg -p hdisk4 hdisk5
```

2. To synchronize the copies on volume groups vg04 and vg05, enter:

```
syncvg -v vg04 vg05
```

3. To display the synchronization status, enter:

```
syncvg -q
An output that is similar to the following example is displayed.
VG Name    Status        Sync Rate     PID            Command
tvg2       SYNCING       128M          8323316         /bin/ksh /usr/sbin/syncvg -l tvg2lv1
tvg2       SYNCING       1M            7536758         /bin/ksh /usr/sbin/syncvg -l tvg2lv3
tvg2       SYNCING       256M          6815782         /bin/ksh /usr/sbin/syncvg -l tvg2lv2
tvg1       SYNCING       2G            7995416         /bin/ksh /usr/sbin/syncvg -l tvg1lv2
tvg1       SYNCING       5M            2949162         /bin/ksh /usr/sbin/syncvg -l tvg1lv3
tvg1       SYNCING       1G            7274582         /bin/ksh /usr/sbin/syncvg -l tvg1lv1
```

4. To pause the **syncvg** command and then display the synchronization status, enter:

```
syncvg -a all
syncvg -q
An output that is similar to the following example is displayed.
VG Name    Status        Sync Rate     PID          Command
tvg2       PAUSE         128M          8323316       /bin/ksh /usr/sbin/syncvg -l tvg2lv1
tvg2       PAUSE         1M            7536758       /bin/ksh /usr/sbin/syncvg -l tvg2lv3
tvg2       PAUSE         256M          6815782       /bin/ksh /usr/sbin/syncvg -l tvg2lv2
tvg1       PAUSE         2G            7995416       /bin/ksh /usr/sbin/syncvg -l tvg1lv2
tvg1       PAUSE         5M            2949162       /bin/ksh /usr/sbin/syncvg -l tvg1lv3
vg1        PAUSE         1G            7274582       /bin/ksh /usr/sbin/syncvg -l tvg1lv1
```

5. To synchronize the current **syncvg** operation with a sync rate of 512 MB/sec on a volume group named vg00, enter:

```
syncvg -T 512 -v vg00
```

Files

Item	Description
/usr/sbin/syncvg	Contains the **syncvg** command.
/tmp	Directory where the temporary files are stored and while the command is running.

syncwpar Command

Purpose

Synchronizes software between a global system and a workload partition.

Syntax

Shared WPAR synchronization

/usr/sbin/syncwpar [[-a] [-i] | [-F] [-r]] [-p] [-v] [-X] { -A | -f *wparnamesfile* | *wparname* }

Detached WPAR synchronization

/usr/sbin/syncwpar -D [-d *device*] [**-p**] [**-v**] [**-X**] { **-A** | **-f** *wparnamesfile wparname* }

Detached WPAR interim fix operations

/usr/sbin/syncwpar -D { -E *<path to fix>* | -R *<ifix label>*} { **-A** | **-f** *wparnamesfile* | *wparname* }

Versioned WPAR device data synchronization

/usr/sbin/syncwpar -c *wparname*

Description

The `syncwpar` command synchronizes the software that is installed in the global shared parts (usually the **/usr** and **/opt**) with the workload partition *root* part.

If you specify the **-D** flag, the **syncwpar** command recovers the system software that is in a detached workload partition (WPAR) with writable **/usr** directory, and that has diverged from the system software in the global environment. If you do not specify the **-D** flag, the **syncwpar** command runs only on shared WPAR that have a read-only **/usr** directory.

Note: The **syncwpar** command cannot be used to synchronize software levels in AIX 5.2 or AIX 5.3 versioned WPAR. Software in versioned WPAR is independent from the software in the global environment.

The `syncwpar` command operates on a single WPAR when you specify the *wparname* parameter, on a list of WPAR when you specify the *wparname* parameter with the `-f` *wparnamesfile* parameter, or on all system WPAR when you specify the `-A` flag.

Restriction: Running the `syncwpar` command on application workload partitions is restricted.

Note: If you run the **syncwpar** command to sync a version 6 workload partition, on a version 7 global system, the **syncwpar** command will call the **migwpar** command and migrates the workload partition.

Flags

Item	Description
`-a`	Performs additional installation only. Does not downlevel the `installp` filesets (that is, uninstall, reject, force overwrite). Not valid with the `-r` flag.
-c	Synchronizes the predefined storage device data in a specified versioned workload partition. The **-d** device flag is not required to synchronize the device data. Synchronization of the device data helps to resolve problems while configuring a storage device in a WPAR.
-D	Synchronizes software in the detached system workload partitions that have a writable **/usr** directory. The default is to synchronize software in only shared system workload partitions that have a read-only **/usr** directory.
`-i`	Only updates the `installp` filesets. Not valid with the `-r` flag.
`-F`	Forces the RPM files to be copied. Not valid with the `-i` flag.
`-r`	Updates only the RPM files. Not valid with the `-i` flag.
`-p`	Previews the operation. Does not actually perform the synchronization.
`-v`	Specifies the verbose mode.
`-X`	Expands file systems if necessary and possible.
`-A`	Synchronizes all of the available system workload partitions with the global system.
`-f`	Specifies the file containing a list of workload partitions in the *wparnamesfile* parameter.
`-d`	Synchronizes the software in a detached WPAR `wpar`, using the specific software installation directories. The **-d** flag is valid only when used with the **-D** flag. • When the **-d** flag is specified, the images in the directory are used to apply the base installation or updates to the detached WPARs. It is important that the install or update images in the specified location are the same as the ones that were last used to install or update the global system so that the resulting software levels match. • When the **-d** flag is not specified, the synchronization rejects or commits the levels of software in the detached WPARs.
`-R`	Removes the specified interim fix from the WPARs. The argument is the label of the **ifix** parameter that must be removed. The flag is valid only for detached system workload partitions.
`-E`	Installs the specified interim fix into the detached system workload partitions. The argument is the full path to the **ifix** parameter. The flag is valid only for detached system workload partitions.

Parameters

Item	Description
wparnamesfile	Specifies the file that contains a list of workload partition names.
wparname	Specifies the name of a workload partition.
device	Specifies the name of a device.

Security

Access Control: Only the root user can run this command.

Examples

1. To synchronize all of the software on workload partition `mywpar`, enter the following command:

```
syncwpar mywpar
```

2. To synchronize all WPAR, enter the following command in the global environment:

```
# syncwpar -A
```

3. To synchronize WPAR that is named `mywpar` and to expand the file system automatically, enter the following command:

```
# syncwpar -X mywpar
```

4. To synchronize software in the detached WPAR named `privatewpar` using the `/mysw` software installation directory, enter the following command:

```
# syncwpar -D -d /mysw privatewpar
```

5. To install the **myfix.epkg.Z** interim fix to all the detached system workload partitions, enter the following command:

```
# syncwpar -D -E /tmp/myfix.epkg.Z -A
```

6. To remove an interim fix with the label **myfix** from all the detached system workload partitions, enter the following command:

```
# syncwpar -D -R myfix -A
```

syscall Command

Purpose

Performs a specified subroutine call.

Syntax

syscall [**-n**] *Name* [*Argument1 ... ArgumentN*] [**;** *Name* [*Argument1 ... ArgumentN*]] ...

Description

The **syscall** command executes a system call interface program, which performs the subroutine call specified by the *Name* parameter. If you specify the **-n** flag, the **syscall** command performs the call **n** times. Arguments specified by the *Argument* parameter are passed to the subroutine without error checking. The *Argument* parameter can be expressed in the following formats:

Item	Description
0x *nnn*	Hexadecimal constant *nnn*.
0 *nnn*	Octal constant *nnn*.
nnn	Decimal constant *nnn*.
+*nnn*	Decimal constant *nnn*.
-*nnn*	Decimal constant *nnn*.

Item	Description
"*string*	The character string "*string*".
'*string*	The character string "*string*".
string	The character string "*string*".
#*string*	The length of the character string "*string*".
&&*n*	The address of the *n*th argument to this subroutine. (*n*=0 is the subroutine name.)
&*n*	The address of the *n*th byte in an internal 10KB buffer.
$*n*	The result of the *n*th subroutine. (*n*=0 is the first subroutine.)
string	Anything else is a literal character string.

The **syscall** command prints a message and exits for unknown subroutines and for subroutines that return a value of -1.

Note: The **syscall** command understands the **sleep** subroutine as a special case subroutine.

Flags

Item	Description
-n	Specifies the number of times the **syscall** command performs the specified subroutine.
;	Separates multiple subroutines (up to a maximum of 20) issued by the same invocation of the **syscall** command.

Examples

To simulate the C program fragment:

```
output=open("x", 401, 0755);
```

```
write(output, "hello", strlen("hello"));
```

enter:

```
syscall open x 401 0755 \; write \$0 hello \#hello
```

Note: Special shell characters must be escaped.

Files

Item	Description
/usr/bin/syscall	Contains the **syscall** command.

sysck Command

Purpose

Checks the inventory information during installation and update procedures.

Syntax

sysck { **-i** | **-u** } [**-R** *RootPath*] [**-N**] [**-v**] [**-s** *SaveFile*] [**-O** { **r** | **s** | **u** }] **-f** *File ProductName* { **tcbck** *Flags* }

All of the **tcbck** command flags are valid with this command.

Description

Note: All of the **tcbck** command flags are valid with the **sysck** command. This feature provides compatibility with Version 3.1.

The **sysck** command checks file definitions against the extracted files from the installation and update media and updates the Software Vital Product Data (SWVPD) database. The **sysck** command does not recognize the following special characters in file names: grave accent (`), quotation marks (', "), backslash (\), caret (^), parentheses ((,)), vertical bar (|), braces ({, }), brackets ([,]), greater than and less than symbols (<, >), colon (:), and comma (,). If a file name contains one of these characters, the **sysck** command fails.

The **sysck** command is primarily used during the installation and update of software products.

When invoked with the **-i** flag, the **sysck** command checks the attributes of an extracted file with its file definitions, updates the SWVPD, and attempts to fix some errors if they exist.

The *File* parameter is the name of the stanza file that contains the file definitions. An example of such a file is the **/etc/security/sysck.cfg** file, although the **syschk** command does not use this file. The **sysck** command checks the size, links, symlinks, owner, group, and mode attributes of a file for which the type attribute is set to **FILE**. When invoked with the **-v** flag as well as the **-i** flag, **sysck** also checks the checksum value of a file.

The **sysck** command updates the file name, product name, type, checksum, and size of each file in the SWVPD database.

To fix errors, the **sysck** command resets the attribute of the installed or updated file to the defined value in the *File* stanza file, except for some attributes as described in "Fixing Errors".

When invoked with the **-u** flag, the **sysck** command removes the entry from the SWVPD database for each file that is part of the software product *ProductName*. The **sysck** command also deletes any hard links and symbolic links for each file, as defined in the SWVPD database.

Flags

Item	Description
-f *File*	Specifies the name of the stanza file that contains the file definitions.
-i	Checks for the correct installation of a software product's files. Updates the SWVPD database with the file definitions, and attempts to fix some errors if found.
-N	Specifies that the SWVPD database should not be updated.
-O {**r**|**s**|**u**}	Specifies which part of the SWVPD is to be updated, as follows: **r** Specifies the root part of the SWVPD. **s** Specifies the **/usr/share** part of the SWVPD. **u** Specifies the **/usr** part of the SWVPD (default).

Item	Description
-R *RootPath*	Use *RootPath* as root instead of "**/**".

Item	Description
-s *SaveFile*	Takes a snapshot of what is currently in the VPD and saves it in stanza format to the file specified by *SaveFile*. Called with the **-u** option. No action is taken in the database with this flag. Must be used with the **-f** option. For example: `sysck -i -s /tmp/save.inv -f /tmp/real.inv bos.rte.shell`
-u	Deletes file entries from the SWVPD and deletes hard links and symbolic links.
-v	Verifies that the checksum is correct.
ProductName	Specifies the installable software product or option that is being checked.

Environment Variables

Item	Description
INUTREE	The environment variable **INUTREE** has only the following four valid values: **NULL** Same as **INUTREE** not being set. **M** Specifies the root part of the SWVPD. **S** Specifies the **/usr/share** part of the SWVPD. **U** Specifies the **/usr** part of the SWVPD (default). **INUTREE** can be used instead of the **-O** *Tree* flag.
INUNOVPD	The environment variable **INUNOVPD** can be null or can be set to 1. If it is set to 1 then **sysck** does not update the SWVPD. **INUNOVPD** can be used instead of the **-N** flag.
INUVERIFY	If the environment variable **INUVERIFY** is set to 1 **sysck** verifies that the checksum attributes in the stanza file are correct. **INUVERIFY** can be used instead of the **-v** flag.

File Definitions

Item	Description
acl	The access control list for the file. If the value is blank, the **acl** attribute is removed. If no value is specified, the command computes a value, according to the format described in Access Control Lists. This attribute should grant x (execute) access only to the root user and members of the security group. The command should **setuid** to the root user and have the trusted computing base attribute.
class	The logical group of the file. A value must be specified because it cannot be computed. The value is *ClassName* [*ClassName*].
checksum	The checksum of the file. If the value is blank, the **checksum** attribute is removed. If no value is specified, the command computes a value, according to the format given in the **sum** command. The value is the output of the **sum -r** command, including spaces.
group	The file group. If the value is blank, the **group** attribute is removed. If no value is specified, the command computes a value, which can be a group ID or a group name.
mode	The file mode. If the value is blank, the **mode** attribute is removed. If no value is specified, the command computes a value, which can be an octal number or a string (**rwx**), and have the **TCB**, **SUID**, **SGID**, and **SVTX** attributes.

Item	Description
owner	The file owner. If the value is blank, the **owner** attribute is removed. If no value is specified, the command computes a value, which can be a user ID or a user name.
size	The size of the file in bytes. If the value is blank, the **size** attribute is removed. A **VOLATILE** value in the size field indicates that the file size will change (so no checksum value can be given). A **NOSIZE** value indicates that the file has 0 length. If no value is specified, the command computes a value, which is a decimal number.
target	Allows symbolic links and hard links to exist as separate stanzas in the inventory. The **target** file definition refers to the full path name of the source of the link, for example: `/etc/foo --> /usr/bar` The **target** is `/usr/bar`.
type	The type of file. This value cannot be blank. If no value is specified, the command computes a value, which can be the **FILE**, **DIRECTORY**, **FIFO**, **BLK_DEV**, **CHAR_DEV**, **LINK**, **MPX_DEV**, and **SYMLINK** keywords.
xacl	An addition to the extended-access control list. A value must be specified as a single entry in an extended-access control list because the value cannot be computed. This attribute is valid only if the **-i** flag is used. For information about the format, see the **acl** file definition above.

Fixing Errors

To fix errors, the **sysck** command resets the attribute of the installed or updated file to the defined value defined in the *File* stanza file except for the following attributes, for which the **sysck** command acts as described:

Item	Description
links	Creates any missing hard links. If a link exists to another file that is not listed in this definition, the link is deleted.
program	If this attribute is included in the *File* stanza file, **sysck** invokes the program. A message is printed if an error occurs, but no additional action is taken.
symlinks	Creates any missing symbolic links. If a link exists to another file that is not listed in this definition, the link is deleted.

Security

Privilege Control: Only the root user can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. A product that uses the **installp** command to install ships an inventory file in its image. To add the definitions to the inventory database and check permissions, links, checksums, etc., enter:

   ```
   sysck -i -f dude.rte.inventory dude.rte
   ```

   where `dude.rte.inventory` would look like the following:

   ```
   /usr/bin/dude.exec:
      class = apply,inventory,dude.rte
   ```

```
    owner = bin
    group = bin
    mode = 555
    type = FILE
    size = 2744
    checksum = "04720       3"
```

2. To remove any links to files for a product that has been removed from the system and remove the files from the inventory database, enter:

   ```
   sysck  -u  -f dude.rte.inventory dude.rte
   ```

Files

Item	Description
/etc/objrepos/inventory	Specifies names and locations of files in a software product on the root.
/usr/lib/objrepos/inventory	Specifies names and locations of files in a software product on the **/usr** file system.
/usr/share/lib/objrepos/inventory	Specifies names and locations of files in a software product on the **/usr/share** file system.

syscorepath Command

Purpose

Specifies a single system-wide directory where all core files of any processes will be dumped.

Syntax

syscorepath [**-p** *DirectoryName*] [**-g**] [**-c**]

Description

The **syscorepath** command enables a system administrator to set up a single system-wide directory in which to dump core files from any processes. This can ease administrative tasks in managing file-system space and provides a single, known directory in which to find core files. By default, the core file is created in the working directory of the process being core-dumped.

The directory should have read and write privileges for all users on the system. If a user does not have permission to write in the directory, a core file will not be created. Core files will be given unique names based on the process ID and time, so a core file will be named **core.*pid*.*ddhhmmss***, where *pid* is the process ID, *dd* is the day of the month, *hh* is the hour in 24-hour format, *mm* is minutes, and *ss* is seconds.

Note: The settings made by the **syscorepath** command do not persist across system reboots. However, the settings made by the **chcore** command persist across system reboots.

Flags

Item	Description
-c	Unsets the current directory specified as the repository for core files. Subsequent core files will be created in the working directory of the process.
-g	Displays current directory specified as the repository for core files.

Item	Description
-p *DirectoryName*	Specifies the directory to use as a repository for core files. *DirectoryName* must be a valid directory name.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Standard Errors

EPERM
User does not have permission.

ENOTDIR
Specified *DirectoryName* is not a directory.

ENAMETOOLONG
Specified *DirectoryName* is too long.

Security

Only the root user can run this command.

Examples

1. To set **/core** as the repository for core files, type:

```
syscorepath -p /core
```

2. To display the current repository for core files, type:

```
syscorepath -g
```

3. To unset the directory used as the repository for core files, type:

```
syscorepath -c
```

Files

Item	Description
/usr/bin/syscorepath	Contains the **syscorepath** command.

sysdumpdev Command

Purpose

Displays and modifies the information and settings that are related to traditional system dump and firmware-assisted system dump.

Syntax

sysdumpdev -P { **-p** *device* | **-s** *device* } [**-q**] [**-i**]

sysdumpdev [**-p** *device* | **-s** *device*] [**-q**]

sysdumpdev [**-d** *directory* | **-D** *directory* | **-e** | **-I** | [**-k** | **-K**] | **-l** | **-p** *device* | **-q** | **-s** *device* | **-z**]

sysdumpdev [**-i**]

sysdumpdev -L { **-v** | -S *device* }

sysdumpdev [**-t** { *traditional* | *fw-assisted* }] [**-f** {*disallow*, *allow, require* }]

Description

The **sysdumpdev** command changes the primary or secondary dump device designation in a system that is running. The primary and secondary dump devices are designated in a system configuration object. The new device designations are in effect until you run the **sysdumpdev** command again, or you restart the system.

If you run the **sysdumpdev** command with the **no** flag, the **sysdumpdev** command identifies the current attributes of the primary and secondary dump devices and writes the attribute values to the ODM object class and NVRAM. The default primary dump device is **/dev/hd6**. The default secondary dump device is **/dev/sysdumpnull**. If the system has 4 GB or more of memory, then the default dump device is **/dev/lg_dumplv**, and **/dev/lg_dumplv** is a dedicated dump device. AIX Version 7.1, and later, extends firmware assisted dump capabilities to make it as the default system dump method if it is supported by the platform.

Note:

- A mirrored paging space might be used as a dump device.
- Do not use a diskette drive as your dump device.
- If you use a paging device, only use hd6, the primary paging device. The AIX operating system supports using any paging device in the root volume group (rootvg) as the secondary dump device.
- If you use a removable device such as a tape or DVD, be aware that the dump does not span volumes. Thus, the dump must fit on a single volume.
- You can configure an iSCSI software initiator device in the root volume group (rootvg) as the dump device for a firmware-assisted system dump, for AIX Version 6.1 with the 6100-01 Technology Level.
- Remote dumps for thin servers are supported for AIX 6.1. You must define the relative dump resource on the NIM master to see the dump resource on the NIM client as an iSCSI disk that can only be used to configure the primary dump device. Only firmware-assisted system dump can be configured on an iSCSI disk device.
- For AIX Version 6.1 with the 6100-06 Technology Level, you can configure a firmware-assisted dump of kernel memory.

For AIX 6.1 and later versions, all dumps are compressed. You should use the **savecore** command to copy dumps from the dump device to a file.

The **sysdumpdev** command supports firmware-assisted system dump for the following features:

- Return of dump size estimation
- Display of information about most recent dump
- Detection of a new dump

The sysdumpdev command also provides the dump type including the traditional dump type or the *fw-assisted* dump type.

The **-t** flag specifies the type of dump. Its possible values are *traditional* and *fw-assisted*.

The **-f** flag specifies the full memory system dump mode. This mode is relevant only for the firmware-assisted system dump. In this mode, the dump is performed independently of the operating system. All of the partition memory is saved to the dump.

Running sysdumpdev in Non-rootvg Volume Groups

You can use a dump-logical volume outside the root volume group, if it is not a permanent dump device and for a traditional system dump only. For example, if the **-P** flag is not specified. However, if you choose a paging space, the dump device cannot be copied unless it is in rootvg. If the dump device must be copied, only rootvg is active before paging is started.

The primary dump devices must always be in the root volume group for permanent dump devices. The secondary device might be outside the root volume group unless it is a paging space.

Flags

Item	Description
-d *directory*	Specifies the *directory* the dump is copied to at system boot. If the copy fails at boot time, you can use the **-d** flag to ignore the system dump.
-D *directory*	Specifies the *directory* the dump is copied to at system boot. If the copy fails at boot time, you can use the **-D** flag to copy the dump to an external media. **Note:** When using the **-d** *directory* or **-D** *directory* flags, the following error conditions are detected: • *directory* does not exist. • *directory* is not in the local journaled file system. • *directory* is not in the **rootvg** volume group.
-e	Estimates the size of the dump (in bytes) for the current running system. The size that is shown is the estimated size of the compressed dump.

Item	Description
-f{ *disallow* | *allow_kernel* | *require_kernel* | *allow_full* | *require_full* }	Specifies whether firmware-assisted system dump does allow, require or forbid the dump of either the kernel memory or the full memory. In kernel memory or full memory mode, the dump is performed independently of the operating system. All of the kernel relevant memory is saved to a kernel memory system dump. All of the partition memory is saved to a full memory system dump. The **-f** flag has the following variables: • The *disallow* variable specifies that neither the full memory system dump mode nor the kernel memory system dump mode is allowed. It is the selective memory mode. • The *allow_full* variable specifies that the full memory system dump mode is allowed but is performed only when operating system cannot properly handle the dump request. • The *require_full* variable specifies that the full memory system dump mode is allowed and is always performed. When the full memory dump is allowed, the dump size estimation specified with the **-e** flag corresponds to the memory size with the applied compression factor.
-i	Indicates that the **sysdumpdev** command was called from a system function. This flag is only used by system utilities. The **-i** flag will not make the requested change if the effected value has already been modified by other than an automatic IBM function; that is, the **-i** flag will not override a previous change.
-I	Resets the indications of previous changes. After the **-I** flag is specified, changes are allowed with the **-i** flag.
-k	If your machine has a key mode switch, it is required to be in the service position before a dump can be forced with the dump key sequences.

Item	Description
-K	If your machine has a key mode switch, the reset button or the dump key sequences will force a dump with the key in the normal position, or on a machine without a key mode switch. **Note:** On a machine without a key mode switch, a dump can not be forced with the key sequence without this value set.
-l	Lists the current value of the primary and secondary dump devices, copy directory, and **forcecopy** attribute. The **-l** flag also displays the current dump type. The following list indicates the possible values that are displayed: • **fw-assisted**: The preferred dump type is firmware-assisted system dump. • **fw-assisted (suspend)**: The preferred dump type is firmware-assisted system dump, but the primary dump device is either not configured or it does not support firmware-assisted system dump. In the latter case, a traditional system dump is triggered. • **traditional**: Only the traditional system dump is available after the **sysdumpdev -t traditional** command. It might also because the firmware-assisted system dump is not supported on this system. To support firmware-assisted system dump, there must be sufficient memory when the system starts up, and POWER6 or later hardware and the supported firmware must be installed.
-L	Displays statistical information about the most recent system dump. This includes date and time of last dump, number of bytes written, and completion status. The **-L** flag shows both the compressed size and the uncompressed size of the dump. The compressed size is the size of what was actually written to the dump device. If no previous dump was recorded in nonvolatile memory, this flag scans the dump devices for the existing dump. **Note:** 1. The dump sizes shown might not reflect the exact size of the dump on the media. There can be a small difference because of disk and copy block sizes. 2. If the dump has failed due to an I/O error, the major and minor device numbers will be those for the failing device.
-P	Makes permanent the dump device specified by **-p** or **-s** flags. The **-P** flag can only be used with the **-p** or **-s** flags.
-p *device*	Temporarily changes the primary dump device to the specified device. The device can be a logical volume, writable DVD, or a tape device or an iSCSI disk configured by NIM for remote dump.
-q	Suppresses all messages to standard output. If this flag is used with the **-l**, **-z**, or **-L** flag, the **-q** flag will be ignored.
-s *device*	Device Temporarily changes the secondary dump device to the specified device. The same devices valid for the **-p** flag are valid here.
-S *device*	Scans a specific dump device for a valid compressed dump. The dump must be from an AIX release with parallel dump support. This flag can be used only with the -L flag.

Item	Description
-t{ *traditional* | *fw-assisted* }	Specifies the type of dump to perform. The **-t** flag has the following variables: • The *traditional* variable specifies that the traditional system dump is performed. In this dump type, the dump data is saved before the system reboot. Under any of the following circumstances, you can only specify the *traditional* variable: – Firmware-assisted system dump is not supported. – Memory is not sufficient when the system starts. – POWER6 or later hardware is not installed. You cannot use the traditional system dump on an iSCSI software initiator dump device. • The *fw-assisted* variable specifies that the firmware-assisted system dump is performed. In this dump type, the dump data is saved in parallel with the system reboot. If the system starts in a low memory configuration, you must explicitly enable the full memory dump using the **-f** flag, especially in iSCSI software initiator configuration where firmware-assisted system dump cannot fall back on the traditional system dump if the full memory dump is not allowed. If you specify the *fw-assisted* variable but the primary dump device is either not configured or it does not support firmware-assisted system dump, a traditional system dump is triggered. When the firmware-assisted system dump type is not allowed at configuration time, or is not enforced at dump request time, a traditional system dump is performed. In addition, because the scratch area is only reserved at initialization, a configuration change from traditional system dump to firmware-assisted system dump is not effective until the system is rebooted.
-v	When the dump status is not 0, this option will display available dump debug information. The debug data, when available, is used by service to diagnose dump failures. This flag can only be used with the **-L** flag.
-z	Determines if a new system dump is present. If one is present, a string containing the size of the dump in bytes and the name of the dump device will be written to standard output. If a new system dump does not exist, nothing is returned. After the **sysdumpdev -z** command is run on an existing system dump, the dump will no longer be considered recent.

If no flags are used with the **sysdumpdev** command, the default dump devices are used.

Security

Access Control: Only the root user can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Error Codes

Note: A nonzero dump status indicates a failed dump. The following values are the possible dump status values and their corresponding light-emitting diode (LED) values:

Dump status	Description	LED value
0	Dump completed successfully	0C0

Dump status	Description	LED value
-1	No dump device defined	0C8
-2	Dump device too small	0C4
-3	Dump crashed or did not start	0C5
-4	I/O error	0C1

Examples

1. To display current dump device settings, enter the following command:

   ```
   sysdumpdev -l
   ```

   For information about the types of dump that this command shows, see the **-l** flag description under the Flags section.
2. To designate logical volume hd7 as the primary dump device, enter the following command:

   ```
   sysdumpdev -p /dev/hd7
   ```
3. To designate tape device `rmt0` as the secondary dump device, enter the following command:

   ```
   sysdumpdev -s /dev/rmt0
   ```
4. To display information from the previous dump invocation, enter the following command:

   ```
   sysdumpdev -L
   ```
5. To permanently change the database object for the primary dump device to `/dev/newdisk1`, enter the following command:

   ```
   sysdumpdev -P -p /dev/newdisk1
   ```
6. To determine if a new system dump exists, enter the following command:

   ```
   sysdumpdev -z
   ```

   If a system dump has occurred recently, an output that is similar to the following is displayed:

   ```
   4537344 /dev/hd7
   ```
7. To specify the directory that a dump is copied to after a system crash, if the dump device is `/dev/hd6`, enter the following command:

   ```
   sysdumpdev -d /tmp/dump
   ```

   This attempts to copy the dump from `/dev/hd6` to `/tmp/dump` after a system crash. If there is an error during the copy, the system continues to boot and the dump is lost.
8. To specify the directory that a dump is copied to after a system crash, if the dump device is `/dev/hd6`, enter the following command:

   ```
   sysdumpdev -D /tmp/dump
   ```

   This attempts to copy the dump from `/dev/hd6` to the `/tmp/dump` directory after a crash. If the copy fails, you are prompted with a menu. You can copy the dump manually to some external media through this menu.
9. To scan a dump device for a dump, enter the following command:

   ```
   sysdumpdev -L -S /dev/hd6
   ```

sysdumpstart Command

Purpose

Provides a command line interface to start a kernel dump to the primary or secondary dump device.

Syntax

sysdumpstart [**-p**] [**-t** *traditional* | **-f** { *disallow* | *require_kernel* | *require_full* }]

sysdumpstart [**-s**] [**-t** *traditional*]

Description

The **sysdumpstart** command provides a command line interface to start a kernel dump to the primary or secondary dump device. When the dump completes, the system halts. Use the **kdb** command to examine a kernel dump. Use the **sysdumpdev** command to reassign the dump device.

During a kernel dump, the following values can be displayed on the three-digit terminal display as follows:

Item	Description
0c0	Indicates that the dump completed successfully.
0c1	Indicates that an I/O occurred during the dump.
0c2	Indicates that the dump is in progress.
0c4	Indicates that the dump is too small.
0c5	Indicates a dump internal error .
0c8	Indicates that the dump was disabled. In this case, no dump device was designated in the system configuration object for dump devices. The **sysdumpstart** command halts, and the system continues running.
0c9	Indicates that a dump is in progress.
0ca	Indicates that a firmware-assisted system dump is not finished yet. System startup resumes after the dump completes.
0cb	Indicates that a dump is in progress.
0cc	Indicates that the system switched to the secondary dump device after attempting a dump to the primary device.

You could also use the System Management Interface Tool (SMIT) **smit sysdumpstart** fast path to run this command.

You can specify the **-t traditional** flag that allows to force a traditional system dump when the firmware-assisted system dump is configured.

Restriction:

- If traditional system dump is the current configuration, the **sysdumpstart** command cannot start a firmware-assisted system dump.
- If firmware-assisted system dump is the current configuration with an iSCSI software initiator dump device, the **sysdumpstart** command cannot start a traditional system dump.

You can specify the **-f** flag that allows to override the current full memory dump configuration.

Flags

Item	Description
-f{ *disallow* | *require_kernel* | *require_full*}	Specifies if neither the kernel memory dump nor the full memory dump is allowed. If allowed, this flag specifies where the kernel memory dump or full memory dump is required. The **-f** flag has the following keywords: • Specify the *disallow* keyword to start a firmware-assisted system dump of selective memory. • Specify the *require_kernel* keyword to start a firmware-assisted system dump of kernel memory. • Specify the *require_full* keyword to start a firmware-assisted system dump of full memory.
-p	Initiates a system dump and writes the results to the primary dump device.
-s	Initiates a system dump and writes the results to the secondary dump device.
-t *traditional*	Forces a traditional system dump independently to the current configuration.

Security

Access Control: Only the root user can run this command.

Examples

1. To start a kernel dump to the primary dump device, enter the following command:

```
sysdumpstart -p
```

2. To start a kernel dump to the secondary dump device, enter the following command:

```
sysdumpstart -s
```

sysline Command

Purpose

Displays system status on the status line of a terminal.

Syntax

/usr/bin/sysline [**-b**] [**-c**] [**-d**] [**-e**] [**-h**] [**-i**] [**-j**] [**-l**] [**-m**] [**-p**] [**-q**] [**-r**] [**-s**] [**-w**] [**-D**] [**-H** *Remote*] [**+N**]

Description

The **sysline** command runs in the background and periodically displays system status information on the status line of the terminal. Not all terminals contain a status line. If no flags are specified, the **sysline** command displays the following status items:

- Time of day
- Current number of processes which may be run
- Number of users (followed by a u)
- Number of executable processes (followed by an r)
- Number of suspended processes (followed by an s)
- Number of users who have logged on and off since the last status report

Finally, if new mail has arrived, a summary of it is printed. If there is unread mail in your mailbox, an asterisk appears after the display of the number of users. The display is normally in reverse video (if your terminal supports this in the status line) and is right-justified to reduce distraction. Every fifth display is done in normal video to give the screen a chance to rest.

If you have a file named **.who** in your home directory, then the contents of that file is printed first. One common use of this feature is to alias the **chdir**, **pushd**, and **popd** commands to place the current directory stack in **/.who** after it changes the new directory.

If you have a file named **.syslinelock** in your home directory, then the **sysline** command will not update its statistics and write on your screen, it will just go to sleep for a minute. This is useful if you want to momentarily disable **sysline**. Note that it may take a few seconds from the time the lock file is created until you are guaranteed that **sysline** will not write on the screen.

Flags

Item	Description
-b	Beeps once every half hour and twice every hour.
-c	Clears the status line for five seconds before each redisplay.
-D	Prints out the current day/date before the time.
-d	Prints status line data in human readable format, debug mode.
-e	Prints out only the information. Suppresses the control commands necessary to put the information on the bottom line. This option is useful for putting the output of the **sysline** command onto the mode line of an **emacs** window.
-H *Remote*	Prints the load average on the remote host *Remote.* If the host is down, or is not sending *rwhod* packets, then the down time is printed instead. If the prefix **ucb** is present, then it is removed.
-h	Prints out the host machine's name after the time.
-i	Prints out the process ID of the **sysline** command process onto standard output upon startup. With this information you can send the alarm signal to the **sysline** process to cause it to update immediately. The **sysline** command writes to the standard error, so you can redirect the standard output into a file to catch the process ID.
-j	Left-justifies the **sysline** command output on terminals capable of cursor movement on the status line.
-l	Suppresses the printing of names of people who log in and out.
-m	Suppresses mail check.
+N	Updates the status line every *N* seconds. The default is 60 seconds.
-p	Suppresses the report of the number of processes that are executable and suspended.
-q	Suppresses the printout diagnostic messages if something goes wrong when starting up.
-r	Suppresses reverse video display.
-s	Prints the short form of a line by left-justifying **iff** (if and only if) escapes are not allowed in the status line. Some terminals (the Televideos and Freedom 100 for example) do not allow cursor movements (or other "intelligent" operations) in the status line. For these terminals, the **sysline** command normally uses blanks to cause right-justification. This flag disables the adding of blanks.
-w	Prints the status on the current line of the terminal, suitable for use inside a one line window (Window mode).

Examples

To display the day and date, the number of processes which may be run, the number of users, and to clear the screen five seconds before it updates, enter:

```
sysline -Dcr
```

Note: This will only work on screens which have status line capabilities.

Files

Item	Description
/etc/utmp	Contains the names of users who are logged in.
/dev/kmem	Contains the process table.
/var/spool/rwho/whod.*	Contains who/Uptime information for remote hosts.
${HOME}/.who	Specifies information to print on the bottom line.
${HOME}/.syslinelock	Specifies that when it exists, **sysline** does not print.

syslogd Daemon

Purpose

Logs system messages.

Syntax

syslogd [**-a**] [**-d**] [**-s**] [**-f** *ConfigurationFile*] [**-m** *MarkInterval*] [**-r**] [**-R**] [**-n**] [**-N**] [**-p** *LogName*] [**-M all**] [**-A AdditionalLog**] [-e]

Description

The **syslogd** daemon reads a datagram socket and sends each message line to a destination described by the **/etc/syslog.conf** configuration file. The **syslogd** daemon reads the configuration file when it is activated and when it receives a hangup signal.

The **syslogd** daemon creates the **/etc/syslog.pid** file, which contains a single line with the command process ID used to end or reconfigure the **syslogd** daemon.

A terminate signal sent to the **syslogd** daemon ends the daemon. The **syslogd** daemon logs the end-signal information and terminates immediately.

Each message is one line. A message can contain a priority code, marked by a digit enclosed in < > (angle braces) at the beginning of the line. Messages longer than 900 bytes may be truncated.

The **/usr/include/sys/syslog.h** include file defines the facility and priority codes used by the configuration file. Locally written applications use the definitions contained in the **syslog.h** file to log messages via the **syslogd** daemon.

Note: The maximum file size for the **syslogd** log file cannot exceed 2GB.

Flags

-a
Suppresses the reverse host name lookup for the messages coming from the remote host and logs the IP address of the remote host in the log files.

-d
Turns on debugging.

-e
Specifies enhanced rotation. All compressed and uncompressed files that are available in the log directory and that are created by the **syslogd** daemon are considered for rotation.

-f *ConfigurationFile*
Specifies an alternate configuration file.

-m *MarkInterval*
Specifies the number of minutes between the **mark** command messages. If you do not use this flag, the **mark** command sends a message with **LOG_INFO** priority sent every 20 minutes. This facility is not enabled by a **selector** field containing an asterisk (*****), which selects all other facilities.

-M all
Specifies not to suppress duplicate messages in logfile. This flag is valid only if used with the all argument.

-s
Specifies to forward a "shortened" message to another system (if it is configured to do so) for all the forwarding syslog messages generated on the local system.

-r
Suppresses logging of messages received from remote hosts.

-R
Disables the facility to receive messages from the network using the internet domain socket.

-n
Suppresses the "`Message forwarded from <log_host_name>: `" string added to the beginning of the syslog message that is forwarded to a remote log host.

-N
Suppresses logging of priority and facility information for each log message.

-p
Specifies an alternate path name for the datagram socket.

-A AdditionalLog
Specifies additional logs that the syslogd daemon checks. By default, the syslogd daemon checks the `/dev/log` file for messages. If this flag is specified, it also checks the additional files for messages. The additional logs might be in the **chroot** path.

Configuration File

The configuration file informs the **syslogd** daemon where to send a system message, depending on the message's priority level and the facility that generated it.

If you do not use the **-f** flag, the **syslogd** daemon reads the default configuration file, the `/etc/syslog.con` file.

The **syslogd** daemon ignores blank lines and lines beginning with a number sign (#).

Format

Lines in the configuration file for the **syslogd** daemon contain a **selector** field, an **action** field, and an optional **rotation** field, separated by one or more tabs or spaces.

The **selector** field names a facility and a priority level. Separate facility names with a , (comma). Separate the facility and priority-level portions of the **selector** field with a . (period). Separate multiple entries in the same selector field with a ; (semicolon). To select all facilities, use an * (asterisk).

The **action** field identifies a destination (file, host, or user) to receive the messages. If routed to a remote host, the remote system will handle the message as indicated in its own configuration file. To display messages on a user's terminal, the **destination** field must contain the name of a valid, logged-in system user.

The **rotation** field identifies how rotation is used. If the **action** field is a file, then rotation can be based on size or time, or both. One can also compress and/or archive the rotated files.

Facilities

Use the following system facility names in the **selector**field:

Facility	Description
kern	Kernel
user	User level
mail	Mail subsystem
daemon	System daemons
auth	Security or authorization
syslog	**syslogd** daemon
lpr	Line-printer subsystem
news	News subsystem
uucp	uucp subsystem
local0 through **local7**	Local use
*****	All facilities

Priority Levels

Use the following message priority levels in the **selector** field. Messages of the specified priority level and all levels above it are sent as directed.

Priority	Description
emerg	Specifies emergency messages (**LOG_EMERG**). These messages are not distributed to all users. **LOG_EMERG** priority messages can be logged into a separate file for reviewing.
alert	Specifies important messages (**LOG_ALERT**), such as a serious hardware error. These messages are distributed to all users.
crit	Specifies critical messages not classified as errors (**LOG_CRIT**), such as improper login attempts. **LOG_CRIT** and higher-priority messages are sent to the system console.
err	Specifies messages that represent error conditions (**LOG_ERR**), such as an unsuccessful disk write.
warning	Specifies messages for abnormal, but recoverable, conditions (**LOG_WARNING**).
notice	Specifies important informational messages (**LOG_NOTICE**). Messages without a priority designation are mapped into this priority message.
info	Specifies informational messages (**LOG_INFO**). These messages can be discarded, but are useful in analyzing the system.
debug	Specifies debugging messages (**LOG_DEBUG**). These messages may be discarded.
none	Excludes the selected facility. This priority level is useful only if preceded by an entry with an ***** (asterisk) in the same **selector** field.

Destinations

Use the following message destinations in the **action** field.

File Name
Full path name of a file opened in append mode

@*Host*
Host name, preceded by the at sign (**@**)

***User*[, *User*][...]**
User names

*****
All users

centralizedlog LogSpaceName/LogStreamName
PowerHA pureScale logstream

Note: You must have PowerHA pureScale appliance to use the *centralizedlog LogSpaceName/LogStreamName* message destination.

Rotation

Use the following rotation keywords in the **rotation** field.

rotate
This keyword must be specified after the **action** field.

size
This keyword specifies that rotation is based on size. It is followed by a number and either a **k** (kilobytes) or **m**(megabytes).

time
This keyword specifies that rotation is based on time. It is followed by a number and either a **h**(hour) or **d**(day) or **w**(week) or **m**(month) or **y**(year).

files
This keyword specifies the total number of rotated files. It is followed by a number. If not specified, then there are unlimited number of rotated files.

compress
This keyword specifies that the saved rotated files will be compressed.

archive
This keyword specifies that the saved rotated files will be copied to a directory. It is followed by the directory name.

Effect of command line flags on syslogd rotation:

The -e flag:
This flag is used to enhance the **syslogd** rotation policy. When this flag is used, all the compressed and uncompressed files are considered during rotation.

If your log file rotation frequency is only determined by time, you can reset the timer by entering the following command:

```
refresh -s syslogd
```

The next rotation that is based on the time of the previous rotation does not occur when this command is run during the scheduled time interval.

Examples

1. To log all mail facility messages at the debug level or above to the file **/tmp/mailsyslog**, enter the following command:

   ```
   mail.debug /tmp/mailsyslog
   ```

2. To send all system messages except those from the mail facility to a host named `rigil`, enter the following command:

   ```
   *.debug;mail.none @rigil
   ```

3. To send messages at the **emerg** priority level from all facilities, and messages at the **crit** priority level and above from the mail and daemon facilities, to users `nick` and `jam`, enter the following command:

```
*.emerg;mail,daemon.crit nick, jam
```

4. To send all mail facility messages to all users' terminal screens, enter the following command:

```
mail.debug *
```

5. To log all facility messages at the debug level or above to the file **/tmp/syslog.out**, and have the file rotated when it gets larger then 500 kilobytes or if a week passes, limit the number of rotated files to 10, use compression and also use **/syslogfiles** as the archive directory, enter the following command:

```
*.debug /tmp/syslog.out rotate size 500k time 1w files 10 compress archive /syslogfiles
```

6. To set the rotation schedule for the `syslog.out` file to rotate only every five days, enter the following command:

```
*.debug /var/log/syslog.out rotate time 5d
```

   You can reset the timer at any time before the next rotation by entering the following command:

```
refresh -s syslogd
```

   After you reset the timer, the next rotation occurs after the scheduled interval of time that starts at the time when the refresh command is entered.

Files

/etc/syslog.conf
Controls the output of **syslogd**.

/etc/syslog.pid
Contains the process ID.

t

The following AIX commands begin with the letter *t*.

tab Command

Purpose

Changes spaces into tabs.

Syntax

tab [**-e**] [*File ...*]

Description

The **tab** command reads the file specified by the *File* parameter or standard input, and replaces spaces in the input with tab characters wherever the **tab** command can eliminate one or more spaces. If you specify a file with the *File* parameter, the **tab** command writes the resulting file back to the original file. If the input is standard input, the **tab** command writes to standard output. The **tab** command assumes that tab stops are set every eight columns, starting with column nine. The file name specified for the *File* parameter cannot exceed **PATH_MAX**-9 bytes in length.

Flag

Item	Description
-e	Replaces only those spaces at the beginning of a line up to the first non-space character.

Example

To replace space characters in the `File` file with tab characters, enter:

```
tab File
```

File

Item	Description
/usr/bin/tab	Contains the **tab** command.

tabs Command

Purpose

Sets tab stops on terminals.

Syntax

tabs [*TabSpec ...*] [**+m** [*Number*]] [**-T***Terminal ...*]

Description

The **tabs** command specifies tab stops on terminals that support remotely settable hardware tab characters. Tab stops are set according to the *TabSpec* parameter, and previous settings are erased.

When you use the **tabs** command, always refer to the leftmost column number as 1, even if your workstation refers to it as 0.

If you do not specify the *TabSpec* parameter, the default value is **-8**.

The following preset formats can be specified for the *TabSpec* parameter:

Item	Description
-a	Sets the tabs to 1, 10, 16, 36, and 72 (IBM System/370 Assembler first format).
-a2	Sets the tabs to 1, 10, 16, 40, and 72 (IBM System/370 Assembler second format).
-c	Sets the tabs to 1, 8, 12, 16, 20, and 55 (COBOL normal format).
-c2	Sets the tabs to 1, 6, 10, 14, and 49 (COBOL compact format, columns 1-6 omitted). With this code, the first column position corresponds to card column 7. One space gets you to column 8, and a tab gets you to column 12. Files using this code should include a format specification of: `<:t-c2 m6 s66 d:>`
-c3	Sets the tabs to 1, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62, and 67 (COBOL compact format with more tabs than **-c2**). These tabs provide the recommended format for COBOL. Files using this code should include a format specification of: `<:t-c3 m6 s66 d:>`
-f	Sets the tabs to 1, 7, 11, 15, 19, and 23 (FORTRAN).
-p	Sets the tabs to 1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57, and 61 (PL/I).
-s	Sets the tabs to 1, 10, and 55 (SNOBOL).
-u	Sets the tabs to 1, 12, 20, and 44.

In addition to the preset formats, the *TabSpec*s parameter can include:

Item	Description
-*Number*	Sets regularly repeating tabs at every *Number* column. (The standard operating system tab setting is **-8**. The **-8** setting is required when using the **nroff** command with the **-h** flag.) Another special case is the **-0** setting, which implies no tabs at all. If more than 20 tabs are set, you must run the **tabs** command twice to clear them.
Number1, *Number2*,...	Sets tabs at the specified column numbers (a comma-separated list in ascending order). You can specify up to 40 numbers. If any number except the first has a plus-sign prefix, the prefixed number is added to the previous number for the next setting. Thus, the tab list specified by **1,10,20,30** provides the same tab settings as the tab list specified by **1,10,+10,+10**.
-*Filep*	Reads the first line of the *Filep* file for a format specification. If the **tabs** command finds a format specification, the **tabs** command sets tabs as specified. If the **tabs** command does not find a format specification, it sets tabs to the system default (**-8**).

It is sometimes convenient to maintain text files with nonstandard tab stop settings (tab stops that are not set at every eighth column). Such files must be converted to a standard format. This is often done by replacing all tab characters with the appropriate number of space characters, before they can be processed by any commands. A format specification occurring in the first line of a text file specifies how tab characters are to be expanded in the remainder of the file.

A format specification consists of a sequence of parameters separated by blanks and surrounded by **<:** and **:>**. Each parameter consists of a letter key, possibly followed immediately by a value. The following parameters are recognized:

Item	Description
ttabs	Specifies the tab stop settings for a file. The value of *tabs* must be one of the following: • A list of column numbers separated by commas, indicating tab stops set at the specified columns. • A - (dash) followed immediately by an integer *n*, indicating tab stops set at intervals of *n* columns, that is, at 1+*n*, 1+2**n*, and so on. • A - (dash) followed by the name of a preset tab stop specification. Up to 40 numbers are allowed in a comma-separated list of tab stop settings. If any number (except the first one) is preceded by a plus sign, it is taken as an increment to be added to the previous value. Therefore, the formats **t1, 10, 20, 30** and **t1, 10, +10, +10** are considered identical. Standard tab stops are specified by **t-8**, or, equivalently, **t1, 9, 17, 25**. This is the tab stop setting that most system utilities assume, and is the most likely setting to find at a terminal. The specification **t-0** specifies no tab stops at all. The preset tab stop specifications that are recognized are as follow: **a** 1, 10, 16, 36, 72 Assembler, IBM System/370, first format **a2** 1, 10, 16, 40, 72 Assembler, IBM System/370, second format **c** 1, 8, 12, 16, 20, 55 COBOL, normal format **c2** 1, 6, 10, 14, 49 COBOL compact format (columns 1-6 omitted). Using this code, the first typed character corresponds to card column 7; one space gets you to column 8; and a tab gets you to column 12. Files using this tab stop setup should include a format specification as follows: `<:t-c2 m6 s66 d:>` **c3** 1, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62, 67 COBOL compact format (columns 1-6 omitted) with more tab stops than **c2**. This is the recommended format for COBOL. The appropriate format specification is: `<:t-c3 m6 s66 d:>` **f** 1, 7, 11, 15, 19, 23 FORTRAN **p** 1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57, 61 PL/I **s** 1, 10, 55 SNOBOL **u** 1, 12, 20, 44 UNIVAC 1100 Assembler
ssize	Specifies a maximum line size. The value of *size* must be an integer. Size checking is performed after tab characters have been expanded, but before the margin is adjusted.
mmargin	Specifies the number of space characters to be added to the beginning of each line. The value of *margin* must be an integer.
d	Indicates that the line containing the format specification is to be deleted from the converted file. The *d* parameter takes no value.

Item	Description
e	Indicates that the current format is valid only until another format specification is encountered in the file. The *e* parameter takes no value.

Default values, which are assumed for parameters not supplied, are **t-8** and **m0**. If the *s* parameter is not specified, no size checking is performed. If the first line of a file does not contain a format specification, the above defaults are assumed for the entire file. The following is an example of a line containing a format specification:

```
<:t5,10,15 s72:>
```

If a format specification can be disguised as a comment, it is not necessary to code the *d* parameter.

Flags

Item	Description
-T*Terminal*	Identifies the terminal so the **tabs** command can set tabs and margins correctly. The *Terminal* variable is one of the terminals specified in the **greek** command. Supported values for the *Terminal* variable include: **ANSI** Any ANSI terminal, such as a VT100 terminal. **hp** Hewlett-Packard hardcopy terminals. **2621** Hewlett-Packard 2621. **2640** Hewlett-Packard 2640. **2645** Hewlett-Packard 2645. Additional hardcopy terminals supported by the **tabs** command include: • 1620 • 1620-12 • 1620-12-8 • 1700 • 1700-12 • 1700-12-8 • 300 • 300-12 • 300s • 300s-12 • 40-2 • 4000a • 4000a-12 • 43 • 450 • 450-12 • 450-12-8 • tn1200 • tn300 • oki If you do not provide the **-T** flag, the value of the environment variable **TERM** is used. If the **-T** flag is provided with no value or if **-T** and **TERM** have invalid values, the error message `unknown terminal` is displayed and the command terminates.
+m *Number*	Moves all tabs to the right the number of columns specified by the *Number* variable. This flag also sets the left margin to the column specified by the *Number* variable. If **m** is specified without a value, the default value for the Number variable is 10. The leftmost margin on most workstations is defined by **+m0**. The first column for tabs is defined as column 0 not column 1. **Note:** If the same flag occurs more than once, only the last flag takes effect.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To set tabs every four spaces, enter:

```
tabs -4
```

2. To set tabs every ten spaces on a VT100 terminal, enter:

```
tabs -10 -TANSI
```

File

Item	Description
/usr/bin/tabs	Contains the **tabs** command.

tail Command

Purpose

Displays the last few lines of a file.

Syntax

Standard Syntax

tail [**-f**] [**-c** *Number* | **-n** *Number* | **-m** *Number* | **-b** *Number* | **-k** *Number*] [*File*]

To Display Lines in Reverse Order

tail [**-r**] [**-n** *Number*] [*File*]

Description

The **tail** command writes the file specified by the *File* parameter to standard output beginning at a specified point. If no file is specified, standard input is used. The *Number* variable specifies how many units to write to standard output. The value for the *Number* variable can be a positive or negative integer. If the value is preceded by + (plus sign), the file is written to standard output starting at the specified number of units from the beginning of the file. If the value is preceded by - (minus sign), the file is written to standard output starting at the specified number of units from the end of the file. If the value is not preceded by + (plus sign) or - (minus sign), the file is read starting at the specified number of units from the end of the file.

The type of unit used by the *Number* variable to determine the starting point for the count is determined by the **-b**, **-c**, **-k**, **-m**, or **-n** flag. If one of these flags is not specified, the **tail** command reads the last ten lines of the specified file and writes them to standard output. This is the same as entering **-n 10** at the command line.

The **-m** flag provides consistent results in both single- and double-byte character environments. The **-c** flag should be used with caution when the input is a text file containing multibyte characters, because output can be produced that does not start on a character boundary.

Flags

Item	Description
-b *Number*	Reads the specified file beginning at the 512-byte block location indicated by the *Number* variable.
-c *Number*	Reads the specified file beginning at the byte location indicated by the *Number* variable.
-f	If the input file is a regular file or if the *File* parameter specifies a FIFO (first-in-first-out), the **tail** command does not terminate after the last specified unit of the input file has been copied, but continues to read and copy additional units from the input file as they become available. If no *File* parameter is specified and standard input is a pipe, the **-f** flag is ignored. The **tail -f** command can be used to monitor the growth of a file being written by another process.
-k *Number*	Reads the specified file beginning at the 1KB block location indicated by the *Number* variable.
-m *Number*	Reads the specified file beginning at the multibyte character location indicated by the *Number* variable. Using this flag provides consistent results in both single- and double-byte character-code-set environments.
-n *Number*	Reads the specified file from the first or last line location as indicated by the sign (+ or - or none) of the *Number* variable and offset by the number of lines *Number*.
-r	Displays the output from the end of the file in reverse order. The default for the **-r** flag prints the entire file in reverse order. If the file is larger than 20,480 bytes, the **-r** flag displays only the last 20,480 bytes.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To display the last 10 lines of the `notes` file, enter:

   ```
   tail notes
   ```

2. To specify the number of lines to start reading from the end of the `notes` file, enter:

   ```
   tail -n 20 notes
   ```

3. To display the `notes` file a page at a time, beginning with the 200th byte, enter:

   ```
   tail -c +200 notes | pg
   ```

4. To follow the growth of a file, enter:

   ```
   tail -f accounts
   ```

   This displays the last 10 lines of the `accounts` file. The **tail** command continues to display lines as they are added to the `accounts` file. The display continues until you press the Ctrl-C key sequence to stop it.

File

Item	Description
/usr/bin/tail	Contains the **tail** command.

talk Command

Purpose

Converse with another user.

Syntax

talk {*User* | *User***@***Host* | *Host***!***User* | *Host***.***User* | *Host***:***User* } [*Tty*] [*Pty*]

Description

The **/usr/bin/talk** command allows two users on the same host or on different hosts to have an interactive conversation. The **talk** command opens both a send window and a receive window on each user's display. Each user is then able to type into the send window while the **talk** command displays what the other user is typing.

To initiate a conversation, a local user executes the **talk** command and specifies a remote user's login ID. The remote user's login ID can contain NLS characters. If the remote user is on a remote host, the name of the host must also be specified in one of the following ways:

```
User@Host
Host!User
Host.User
Host:User
```

When using full domain names, the only valid form for specifying the user and host is *User***@***Host*. For example, `michael@host17.dev.ibm.com` initiates a conversation with user `michael` at host `host17` in the `dev.ibm.com` domain.

When the local user initiates the conversation, a message is sent to the remote user, inviting a conversation. If the local user also specifies `tty`, the invitation message is sent only to the specified terminal. Otherwise, the invitation is sent to the remote user's login terminal. This usually is the console, but it may be another terminal. Once this invitation is received, the **talk** command displays two windows on the local user's terminal and displays progress messages until the remote user responds to the invitation.

> **Note:** If the remote user is running AIXwindows and has no other terminals open, the **talk** command cannot send an invitation.

To have the conversation, the remote user also has to execute the **talk** command from any terminal and specify the local user's account name and host name, if appropriate. When the remote user accepts the invitation, the **talk** command displays two windows on each user's terminal. One window displays what is typed by the local user; the other window displays what is typed by the remote user. To end the conversation, either user can press the Interrupt (Ctrl-C) key sequence and the connection is closed. The Interrupt key sequence can be displayed and modified using the **stty** command.

If the users involved in the conversation are using National Language Support (NLS) capabilities, their terminals must support the printing of NLS characters. The same is true for conversations using Kanji capabilities; the terminals being used must support the printing of Kanji characters.

The **talk** command requires a valid address to which to bind. The host name of the remote machine must be bound to a working network interface, which is usable by other network commands, such as the **ping** command. If a machine has no network interface, that is a standalone machine, it must bind its host name to the loopback address (127.0.0.1) in order for the **talk** command to work. For example, two

users named `local` and `remote` on a standalone machine could initiate a conversation, using the **talk** command, by entering:

```
talk remote@loopback
```

To which user `remote` responds:

```
talk local@loopback
```

To disallow **talk** command invitations, the remote user can issue the **mesg** command.

> **Note:** The **talk** command uses the Talk 4.3 protocol.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To talk to a user logged in on a remote host, enter:

   ```
   talk dale@host2
   ```

   In this example, the local user wants to talk with user `dale` who is logged in on `host2`.
2. To talk to a user only if that user is logged in on the console of a remote host, enter:

   ```
   talk dale@host2 console
   ```

   User `dale` receives this message only if logged in on the console at `host2`.

talkd Daemon

Purpose

Provides the server function for the **talk** command.

Syntax

/usr/sbin/talkd [**-s**]

Description

> **Note:** The **talkd** daemon is normally started by the **inetd** daemon. It can also be controlled from the command line, using SRC commands.

The **/usr/sbin/talkd** daemon is the server that notifies a user (the recipient) that another user (the caller) wants to initiate a conversation. The daemon sets up the conversation if the recipient accepts the invitation. The caller initiates the conversation by executing the **talk** command specifying the recipient. The recipient accepts the invitation by executing the **talk** command specifying the caller.

The **talkd** daemon listens at the socket defined in the **/etc/services** file. When the **talkd** daemon receives a LOOK_UP request from a local or remote **talk** process, the **talkd** daemon scans its internal invitation table for an entry that pairs the client process (the local or remote **talk** process) with a caller.

If no entry exists in the invitation table, the **talkd** daemon assumes that the client process is the caller. The **talkd** daemon then receives the client process' ANNOUNCE request. The **talkd** daemon broadcasts

an invitation on the remote computer where the recipient first logged in (unless the caller specifies a particular tty device). This terminal usually is the console, but it may be another terminal.

Otherwise, the invitation is sent to the terminal that the second user first logged in to. This usually is the console, but it may be another terminal.

If an entry does exist in the **talkd** daemon's internal invitation table, the **talkd** daemon assumes that the client is the recipient. The **talkd** daemon returns the appropriate rendezvous address to the **talk** process for the recipient. The recipient process then establishes a stream connection with the caller process.

> **Note:** The **talkd** daemon uses the Talk 4.3 protocol. The subserver name for the AIX protocol is **ntalk**.

Changes to the **talkd** daemon can be made using the System Management Interface Tool (SMIT) or System Resource Controller (SRC), by editing the **/etc/inetd.conf** or /**etc**/**services** file. Entering `talkd` at the command line is not recommended. The **talkd** daemon is started by default when it is uncommented in the **/etc/inetd.conf** file.

The **inetd** daemon get its information from the /**etc**/**inetd.conf** file and the /**etc**/**services** file.

After changing the **/etc/inetd.conf** or /**etc**/**services** file, run the **refresh -s inetd** or **kill -1** *InetdPID* command to inform the **inetd** daemon of the changes to its configuration file.

Debugging messages are sent to the **syslogd** daemon.

> **Note:** The **talkd** daemon should be controlled using the System Management Interface Tool (SMIT) or by changing the **/etc/inetd.conf** file.

Manipulating the talkd Daemon with the System Resource Controller

The **talkd** daemon is a subserver of the **inetd** daemon, which is a subsystem of the System Resource Controller (SRC). The **talkd** daemon is a member of the **tcpip** SRC subsystem group. This daemon is enabled by default in the **/etc/inetd.conf** file and can be manipulated by the following SRC commands:

Item	Description
startsrc	Starts a subsystem, group of subsystems, or a subserver.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
lssrc	Gets the status or a subsystem, group or subsystems, or a subserver.

Flags

Item	Description
-s	Turns on socket-level debugging.

Examples

1. To start the **talkd** daemon, enter the following:

   ```
   startsrc  -t ntalk
   ```

   This command starts the **talkd** subserver.
2. To stop the **talkd** daemon normally, enter the following:

   ```
   stopsrc  -t ntalk
   ```

   This command allows all pending connections to start and existing connections to complete but prevents new connections from starting.
3. To force stop the **talkd** daemon and all **talkd** connections, enter the following:

   ```
   stopsrc -f -t ntalk
   ```

This command terminates all pending connections and existing connections immediately.

4. To display a short status report about the **talkd** daemon, enter the following:

   ```
   lssrc -t ntalk
   ```

   This command returns the daemon's name, process ID, and state (active or inactive).

Files

Item	Description
/etc/utmp	Contains data about users currently logged in.

tapechk Command

Purpose

Performs consistency checking on the streaming tape device.

Syntax

tapechk [**-?**] *Number1 Number2*

Description

The **tapechk** command performs rudimentary consistency checking on an attached streaming tape device. Some hardware malfunctions of a streaming tape drive can be detected by simply reading a tape. The **tapechk** command provides a way to perform tape reads at the file level.

Because the streaming tape drive cannot backspace over physical data blocks or files, the **tapechk** command rewinds the tape to its starting position prior to each check. This command either checks data for the next number of files specified by the *Number1* parameter or skips the next number of files specified by the *Number2* parameter. If you do not specify any parameters, the **tapechk** command rewinds the tape and checks only the first physical block.

The **tapechk** command uses the device in the **TAPE** environment variable if it is defined. Otherwise, the default tape device is **/dev/rmt0**.

> **Note:** The **backup** command allows you to archive files selectively or as an entire file system. It writes data as a continuous stream terminated by a file mark, regardless of the number of files specified. The **tapechk** command perceives each stream of data as a single file, which is important when you specify numeric parameters.

Although you can use the **tapechk** command on any streaming tape cartridge, it is primarily designed for checking tapes written by the **backup** command.

Flag

Item	Description
-?	Explains the format of the **tapechk** command. **Note:** If you specify the **-?** flag, it must be specified before the *Number1* and *Number2* parameters.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Example

To check the first three files on a streaming tape device, enter:

```
tapechk 3
```

File

Item	Description
/usr/sbin/tapechk	Contains the **tapechk** command.

tar Command

Purpose

Manipulates archives.

Syntax

X/Open Standards:

tar {-c|-r|-t|-u|-x} [-B] [-d] [-E] [-F] [-h] [-i] [-l] [-m] [-o] [-p] [-s] [**-U**] [-v] [-w]
[-*Number*] [-f *Archive*] [-b *Blocks*]
[-S [*Feet*] [*Feet @Density*] [*Blocksb*]] [-L *InputList*] [-X *ExcludeList*]
[-N *Blocks*] [-R] [-D] [-C *Directory*] [-Z] *File* | *Directory* ...

Berkeley Standards:

tar { c | r | t | u | x } [b B d D E f F h i l L X m N o p R s S **U** v w Z [*0-9*]]
[*Blocks*] [*Archive*] [*InputList*] [*ExcludeFile*]
[[*Feet*] | [*Feet@Density*] | [*Blocksb*]] *Directory* | *File* ...

Description

Note:

1. The **ustar** header format allows unlimited (2^64 -1) file sizes.
2. The **tar** command does not preserve the sparse nature of any file that is sparsely allocated. Any file that was originally sparse before the restoration will have all space allocated within the filesystem for the size of the file.

The **tar** command manipulates archives by writing files to, or retrieving files from an archive storage medium. The files used by the **tar** command are represented by the *File* parameter. If the *File* parameter refers to a directory, then that directory and recursively all files and directories within it are referenced as well.

The **tar** command looks for archives on the default device (usually tape), unless you specify another device with the **-f** *Archive* flag. When specifying path names that are greater than 100 characters for the United States Tape Archiver (USTAR) format, remember that the path name is composed of a prefix buffer, a / (slash), and a name buffer.

The **tar** command supports the length of **path+filename** only till the system defined **PATH_MAX** limit. Any length of **path+filename** input greater than **PATH_MAX** limit is not archived

When writing to an archive, the **tar** command uses a temporary file (the /**tmp**/**tar***** file) and maintains in memory a table of files with several links. You receive an error message if the **tar** command cannot create the temporary file, or if there is not enough memory available to hold the link tables.

Two groups of flags exist for the **tar** command: the required flags and the optional flags. The required flags control the actions of the **tar** command and include the **-c**, **-r**, **-t**, **-u**, and **-x** flags. At least one required flag must be selected for the **tar** command to function. Having selected a required flag, you can select an optional flag but none are necessary to control the **tar** command.

Note:

1. When the storage device is an ordinary file or a block special file, the **-u** and **-r** flags backspace. However, raw magnetic tape devices do not support backspacing. So when the storage device is a raw magnetic tape, the **-u** and **-r** flags rewind the tape, open it, and then read it again.
2. Records are one block long on block magnetic tape, but they are typically less than half as dense on raw magnetic tape. As a result, although a blocked raw tape must be read twice, the total amount of tape motion is less than when reading one-block records from a block magnetic tape once.
3. The structure of a streaming tape device does not support the addition of information at the end of a tape. Consequently when the storage device is a streaming tape, the **-u** and **-r** flags are not valid options. An attempt to use these flags results in the following error message:

   ```
   tar: Update and Replace options not valid for a
   streaming tape drive.
   ```

4. No recovery exists from tape errors.
5. The performance of the **tar** command to the IBM9348 Magnetic Tape Unit Model 12 can be improved by changing the default block size. To change the block size, enter the following at the command line:

   ```
   chdev -1 <device_name> -a block_size=32k
   ```

For more information on using tape devices see the **rmt** special file.

Flags

Flags for the **tar** command are in two groups, the required and the optional. You must supply at least one required flag to control the **tar** command.

Table 21. Required Flags

Required Flags	Description
-c	Creates a new archive and writes the files specified by one or more *File* parameters to the beginning of the archive.
-r	Writes the files specified by one or more *File* parameters to the end of the archive. This flag is not valid for any tape devices because such devices do not support the addition of information at the end of a tape.
-t	Lists the files in the order in which they appear in the archive. Files can be listed more than once.

Table 21. Required Flags (continued)

Required Flags	Description
-u	Adds the files specified by one or more *File* parameters to the end of the archive only if the files are not in the archive already, or if they have been modified since being written to the archive. The **-u** flag is not valid for any tape devices because such devices do not support the addition of information at the end of a tape.
-U	Allows archival and extraction of Extended Attributes. The Extended Attributes include Access control list (ACL) also.
-x	Extracts the files specified by one or more *File* parameters from the archive. If the *File* parameter refers to a directory, the **tar** command recursively extracts that directory from the archive. If you do not specify the *File* parameter, the **tar** command extracts all of the files from the archive. When an archive contains multiple copies of the same file, the last copy extracted overwrites all previously extracted copies. If the file being extracted does not already exist on the system, the file is created. If you have the proper permissions, the **tar** command restores all files and directories with the same owner and group IDs as they have on the tape. If you do not have the proper permissions, the files and directories are restored with your owner and group IDs. It is not possible to ask for any occurrence of a file other than the last.

Table 22. Optional Flags

Optional Flags	Description
-B	Forces input and output blocking to 20 blocks per record. With this option, the **tar** command can work across communications channels where blocking may not be maintained.
-b *Blocks*	Specifies the number of 512 bytes blocks per record. Both the default and the maximum is 20, which is appropriate for tape records. Due to the size of interrecord gaps, tapes written with large blocking factors can hold much more data than tapes with only one block per record. The block size is determined automatically when tapes are read (the **-x** or **-t** function flags). When archives are updated with the **-u** and **-r** functions, the existing record size is used. The **tar** command writes archives using the specified value of the *Blocks* parameter only when creating new archives with the **-c** flag. For output to ordinary files with the **-f** flag, you can save disk space by using a blocking factor that matches the size of disk blocks (for example, the **-b4** flag for 2048-byte disk blocks).

Table 22. Optional Flags (continued)	
Optional Flags	**Description**
-C *Directory*	Causes the **tar** command to perform a **chdir** subroutine to the directory specified by the *Directory* variable. Using the **-C** flag allows multiple directories that are not related by a close common parent to be archived, using short relative path names. For example, to archive files from the **/usr/include** and **/etc** directories, you might use the following command: `tar c -C /usr/include File1 File2 -C /etc File3 File4` You can use multiple **-C** options when you extract files from the archive. When you use multiple **-C** options, each instance of the **-C** *Directory* is relative to the one that is listed before it in the command. For example, the second **-C** *Directory* is relative to the first **-C** *Directory*. If an archive contains a file with an absolute path name, for example `/home/dir1/filename`, the file is extracted into the directory that is specified by the **-C** *Directory* by removing the leading slash (/) from the filepath or filename. The **-C** *Directory* flag must appear after all other flags and can appear in the list of file names given.
`-D`	Suppress recursive processing when directories are specified.
-d	Makes separate entries for block files, special character files, and first-in-first-out (FIFO) piped processes. Normally, the **tar** command will not archive these special files. When writing to an archive with the **-d** flag, the **tar** command makes it possible to restore empty directories, special files, and first-in-first-out (FIFO) piped processes with the **-x** flag. **Restriction:** Although anyone can archive special files, only a user with root user authority can extract them from an archive (FIFO can also be extracted by non-root users).
-E	Avoids truncation of the long user and group names during addition of files to new or existing archive.
-F	Checks the file type before archiving. Source Code Control Systems (SCCS), Revision Control Systems (RCS), files named **core**, **errs**, **a.out**, and files ending in **.o** (dot o) are not archived.

Table 22. Optional Flags (continued)	
Optional Flags	**Description**
-f *Archive*	Uses the *Archive* variable as the archive to be read or written. When this flag is not specified, the **tar** command uses a system-dependent default file name of the form **/dev/rmt0**. If the *Archive* variable specified is - (minus sign), the **tar** command writes to standard output or reads from standard input. If you write to standard output, the **-c** flag must be used.
-h	Forces the **tar** command to follow symbolic links as if they were normal files or directories. Normally, the **tar** command does not follow symbolic links.
-i	Ignores header checksum errors. The **tar** command writes a file header containing a checksum for each file in the archive. When this flag is not specified, the system verifies the contents of the header blocks by recomputing the checksum and stops with a directory checksum error when a mismatch occurs. When this flag is specified, the **tar** command logs the error and then scans forward until it finds a valid header block. This permits restoring files from later volumes of a multi-volume archive without reading earlier volumes.
-L *InputList*	The *Inputlist* argument to the -L option should always be the name of the file that lists the files and directories that need to be archived or extracted.
-l	Writes an error message to standard output for each file with a link count greater than 1 whose corresponding links were not also archived. For example, if **file1** and **file2** are hard-linked together and only **file1** is placed on the archive, then the **-**l flag will issue an error message. Error messages are not displayed if the **-l** flag is not specified.
-m	Uses the time of extraction as the modification time. The default is to preserve the modification time of the files.
-N *Blocks*	Allows the **tar** command to use very large clusters of blocks when it deals with streaming tape archives. Note however, that on input, the **tar** command cannot automatically determine the block size of tapes with very long block sizes created with this flag. In the absence of a **-N** *Blocks* flag, the largest block size that the **tar** command can automatically determine is 20 blocks.

Table 22. Optional Flags (continued)

Optional Flags	Description
-o	Provides backwards compatibility with older versions (non-AIX) of the **tar** command. When this flag is used for reading, it causes the extracted file to take on the User and Group ID (UID and GID) of the user running the program, rather than those on the archive. This is the default behavior for the ordinary user.
-p	Restores fields to their original modes, ignoring the present umask. The **setuid**, **setgid**, and tacky bit permissions are also restored to the user with root user authority. This flag restores files and directories to their original mode.
`-R`	Use recursion when directories are specified. Ignored when used with the `-D` option.
-s	Tries to create a symbolic link If the **tar** command is unsuccessful in its attempt to link (regular link) two files with the **-s** flag.
-S *Blocks* **b**, **-S** *Feet*, **-S** *Feet*@*Density*	Specifies the number of 512KB blocks per volume (first format), independent of the tape blocking factor. You can also specify the size of the tape in feet by using the second form, in which case the **tar** command assumes a default *Density* variable. The third form allows you to specify both tape length and density. Feet are assumed to be 11 inches long to be conservative. This flag lets you deal more easily with multivolume tape archives, where the **tar** command must be able to determine how many blocks fit on each volume. **Note:** 1. Tape drives vary in density capabilities. The *Density* variable calculates the amount of data a system can fit on a tape. 2. When using 1/4-inch tape devices, be sure to take into account the number of tracks on the tape device when specifying the value for the *Feet* variable. For example, a 4-track,1/4-inch tape drive with a 600-foot tape and a density of 8000 bpi can be specified using the **-S** *Feet*@*Density* flag as follows: `-S 2400@8000` where 600 feet multiplied by 4 tracks equals 2400 feet.
`-U`	Archives or restores named extended attributes and ACLs. When listing, this option will display the names of any named extended attributes and the type of any ACLs associated with each file that are part of the archive image.

Table 22. Optional Flags (continued)	
Optional Flags	**Description**
-v	Lists the name of each file as it is processed. With the **-t** flag, **-v** gives more information about the tape entries, including file sizes, times of last modification, User Number (UID), Group Number (GID), and permissions.
-w	Displays the action to be taken, followed by the file name, and then waits for user confirmation. If the response is affirmative, the action is performed. If the response is not affirmative, the file is ignored.
-*Number*	Uses the **/dev/rmt***Number* file instead of the default. For example, the **-2** flag is the same as the **-f/dev/rmt2** file.
-X *ExcludeList*	Excludes the file names or directories given in the *ExcludeList* from the tar archive being created, extracted or listed. The *ExcludeList* shall contain only one filename or directory per line which are to be excluded from the tar archive being created, extracted from or listed. The -X option can be specified multiple times and it takes precedence over all other options.
-Z	Archives the Encrypted File System (EFS) information of encrypted files or directories. The EFS information is extracted by default. When you specify the **-t** and **-v** flags along with the **-Z** flag, an **e** indicator is displayed after the file mode for encrypted files and directories that were archived with the **-Z** flag, and a hyphen (-) is displayed after the file mode for other files. **Restriction:** Archives created with the **-Z** flag can be restored only on AIX 6.1 or later releases.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To write the `file1` and `file2` files to a new archive on the default tape drive, enter:

```
tar -c file1 file2
```

2. To extract all files in the /tmp directory from the archive file on the /dev/rmt2 tape device and use the time of extraction as the modification time, enter:

```
tar -xm -f/dev/rmt2 /tmp
```

3. To create a new archive file that contains the file1 file and pass the archive file to the **dd** command to be written to the /dev/rmt1 device, enter:

```
tar -cvf - file1 | dd of=/dev/rmt1 conv=sync
```

4. To display the names of the files in the out.tar disk archive file on the current directory, enter:

```
tar -vtf out.tar
```

5. To expand the compressed **tar** archive file, fil.tar.z, pass the file to the **tar** command, and extract all files from the expanded **tar** archive file, enter:

```
zcat fil.tar.Z | tar -xvf -
```

6. To archive the contents of /usr/include and /usr/bin files using short relative path names, enter:

```
cd /usr
tar -cvf/dev/rmt0 -C./include . -C ../bin .
```

Requirement: When specifying multiple instances of the **-C** flag with relative path names, the user must take the previous **-C** flag request into account.

7. To archive to an 8-mm device when using the **-S** flag, enter:

```
tar -cvf /dev/rmt0 -S 4800000b /usr
```

Restriction: When archiving to an 8-mm device, avoid using the **-S** *Feet* and **-S** *Feet*@*Density* flags, because the 8-mm device does not use the concept of density when writing to a tape.

8. To archive a list of all C files that is listed in the file through the *InputList* argument of the **-L** option, enter:

```
tar -cvf fl.tar -L fl_list
```

Where fl_list is a file consisting a list of all .c files in it. This can be obtained as follows:

```
ls *.c > fl_list
```

9. To archive a list of all C files by setting a variable using the **-L** option, enter:

```
ls *.c > fl_list
fl=fl_list
tar -cvf var.tar -L $fl
```

10. To avoid the truncation of long user or group names during creation of the archive, enter:

```
tar -cvEf file.tar file
```

11. To create a new archive file that contains the file1 file with ACL and EA, enter:

```
tar -cvUf /tmp/tar.ar file1
```

Berkeley Options

The following are examples of the Berkeley options using the **tar** command:

Tip: With Berkeley options the arguments to the flags should be given in exact order in which the flags are given below. For example:

```
tar cvfbL test.tar 20 infile
```

where `test.tar` is archive tar file, `20` is number of blocks, and `infile` is *Inputlist* for the archive.

1. To archive all directories and complete filenames listed in input list file `infile` into `ar.tar`, enter :

   ```
   tar cvfL ar.tar infile
   ```

   Where `infile` contains the pathnames of files that are to be archived.
2. To archive files within directories listed in the input list file `infile` into `ar.tar`, enter:

   ```
   tar cvRfL ar.tar infile
   ```
3. To extract directories and complete files specified in the input list file `infile` from an archive named `ar.tar`, enter:

   ```
   tar xvfL ar.tar infile
   ```
4. To extract files from within directories and complete files specified in the input list file `infile` from an archive named `ar.tar`, enter:

   ```
   tar xvRfL ar.tar infile
   ```

Files

Item	Description
/dev/rmt0	Specifies the default tape device.
/bin/tar	Specifies the symbolic link to the **tar** command.
/usr/bin/tar	Contains the **tar** command.
/tmp/tar*	Specifies a temporary file.

Tip: In AIX 3.2, the entire **/bin** directory is a symbolic link to **/usr/bin**.

tbl Command

Purpose

Formats tables for the **nroff** and **troff** commands.

Syntax

tbl [**-TX**] [**—**] [*File*... | **-**]

Description

The **tbl** command is a preprocessor that formats tables for the **nroff** and **troff** commands. It reads one or more files. If no *File* parameter or **-** (minus sign) is specified as the last parameter, the command reads standard input by default. It copies the input unchanged to standard output, except for text between lines containing **.TS** and **.TE**. The **tbl** command reformats such text, which describes tables, without altering the **.TS** and **.TE** lines.

Depending on the target output device, the output formatted by the **nroff** command may need to be post-processed by the **col** command to produce correct output.

Note: To minimize the volume of data passed through pipelines, enter the **tbl** command first when using it with the **eqn** or **neqn** command.

Input Format

The **tbl** command processes text that is displayed within the following format:

```
[ {.DS .DF} ]
.TS
Options ;
Format .
Data
.TE
[.DE]
```

To include short tables in an **mm** macro document, enclose them within the **.DS** (or **.DF**) and **.DE** macro pair.

Options

Following are the available global options for the input format:

Option	Purpose
center or **CENTER**	Centers the line.
expand or **EXPAND**	Expands to line length.
box or **BOX**	Encloses in a box.
allbox or **ALLBOX**	Boxes all entries.
doublebox or **DOUBLEBOX**	Encloses in two boxes.
tab(*Character***)** or **TAB(***Character***)**	Changes the tab character to the *Character* value.
linesize(*Number***)** or **LINESIZE(***Number***)**	Makes all lines the thickness of the point size specified by the *Number* value.
delim(*XY***)** or **DELIM(***XY***)**	Recognizes the *X* and *Y* variables as **eqn** command delimiters.
;	Denotes end of options.

Format

The *Format* variable in the Input Format describes the format of text. Each format line (the last of which must end with a period) describes all remaining lines of the table. A single-key letter describes each column of each line of the table. Follow this key letter with specifiers that determine the font and point size of the corresponding item, indicate where vertical bars are to displayed between columns, and determine such things as column width and intercolumn spacing. The following are the available key letters:

Item	Description
l or **L**	Left-adjusts column.
r or **R**	Right-adjusts column.
c or **C**	Centers column.

Item	Description
n or **N**	Numerically aligns column. **Note:** Numerically aligned data, **n** or **N** format specification, are based upon the locale that is specific for *RADIXCHAR*, which is assumed to be a single character. The alignment can also be determined using the \& (backslash, ampersand) character sequence independent of the presence of any *RADIXCHAR* characters. If more than one *RADIXCHAR* character is displayed in a numerically aligned field, the last one is used for alignment. If no *RADIXCHAR* characters are displayed in a particular column, the alignment is based on the last ASCII arabic numeral. If there is no ASCII numeral and no *RADIXCHAR* character in a column, the data is centered.
a or **A**	Left-adjusts subcolumn.
s or **S**	Spans item horizontally.
t or **T**	Pushes vertical spans to top.
v or **V**	Adjusts vertical line spacing.
^	Spans item vertically.
u or **U**	Moves item half-line up.
z or **Z**	Indicates zero-width item.
-	Indicates horizontal line.
=	Indicates double horizontal line.
|	Indicates vertical line.
||	Indicates double vertical line.
b or **B**	Indicates boldface item.
i or **I**	Indicates italic item.
f*Character* or **F***Character*	Changes to the font specified by the *Character* variable.
p*Number* or **P***Number*	Changes to the size specified by the *Number* variable.
w(*Number***)** or **W(***Number***)**	Sets minimum column width equal to the *Number* variable value.
NumberNumber	Spaces between columns.
e or **E**	Makes equal-width columns.
.	Ends format.

Data

Handling data within the input format, especially for tables, uses the following line commands:

Item	Description
T{**...T**}	Indicates text block, as follows: *Data***<**TAB**>T**{ *Text Block* **T**}**<**TAB**>***Data*

Item	Description
_	Writes short horizontal line.
\R*X*	Repeats the *X* parameter value across a column.
\^	Indicates that the item listed previously spans downward into this row.
.T&	Starts new format.
.TS H, **.TH**, and **.TE**	Allows multi-page tables with column headings repeated on each page. (This is a feature of the **mm** macros.)

Parameters

Item	Description
File	Specifies the files that the **tbl** command will be processing.

Flags

Item	Description
-TX	Uses only full vertical line motions, making the output suitable for line printers and other devices that do not have partial vertical line motions.
—	(double dash) Indicates the end of flags.
-	Forces input to be read from standard input.

Examples

The following example shows coded input, and associated table output of the **tbl** command. The @ (at sign) is used in input to represent an input tab character.

Input

```
.TS
center box ;
cB s s
cI | cI s
^ | c c
l | n n .
Household Population
_
Town@Households
@Number@Size
=
Bedminster@789@3.26
Bernards Twp.@3087@3.74
Bernardsville@2018@3.30
Bound Brook@3425@3.04
Bridgewater@7897@3.81
Far Hills@240@3.19
.TE
```

tc Command

Purpose

Interprets text into the **troff** command output for the Tektronix 4015 system.

Syntax

tc [**-t**] [**-e**] [**-a** *Number*] [**-o** *List* | **-s** *Number*] [**—**] [*File* | **-**]

Description

The **tc** command interprets input as output from the **troff** command. The **tc** command reads one or more English-language files. If no file is specified or the **-** (minus sign) flag is specified as the last parameter, standard input is read by default. The standard output of the **tc** command is intended for a Tektronix 4015 (a 4014 terminal with ASCII and APL character sets). The various typesetter sizes are mapped into the 4014's four sizes. The entire **troff** command character set is drawn using the 4014 character generator, with overstruck combinations where necessary.

At the end of each page, the **tc** command waits for a new-line character from the keyboard before continuing to the next page. While it waits, the following commands are recognized:

Item	Description
!*Command*	Sends the value of the *Command* variable to the shell.
-e	Does not erase before each page.
-*Number*	Skips backward the specified number of pages.
-a*Number*	Sets the aspect ratio to the value of the *Number* variable.
?	Prints a list of available options.

Note: The **tc** command does not distinguish among fonts.

Parameters

Item	Description
File	Specifies the English-language text files to be interpreted as output from the **troff** command.

Flags

Item	Description
-a *Number*	Sets the aspect ratio to the specified number. The default is 1.5.
-e	Does not erase before each page.
-o *List*	Prints only the pages enumerated in the *List* variable. The list consists of pages and page ranges (for example, 5-17) separated by commas. The range *Number***-** goes from the *Number* variable value to end; the range **-***Number* goes from the beginning to and including the page specified by the *Number* variable.
-s *Number*	Skips the first specified number of pages.
-t	Does not wait between pages when directing output into a file.
-	Reads from standard input.
—	(double dash) Indicates the end of flags.

Example

To use the **tc** command in a pipeline with the **troff** command, enter:

```
troff [Flag...] [File...] | tc
```

tcbck Command

Purpose

Audits the security state of the system.

Syntax

Check Mode

tcbck { **-n** | **-p** | **-t**| **-y** } [**-i**] [**-o**] { **ALL** | **tree** | { *Name* ... *Class* ... } }

Update Mode

tcbck -a -f *File* | *PathName Attribute* **=** *Value ...*

OR

tcbck -d -f*File* | { *PathName* ... | *Class* ... }

OR

tcbck -l /dev/*filename* /dev/*filename*

Exit Status

This command returns the following exit values:

0
User definition files are appropriate.

>0
An error occurred or there is an error in one or more user definition files.

The following error codes are returned:

EINVAL (22)
Invalid command line arguments

ENOENT (2)
One or more user definition files do not exist

ENTRUST (114)
Errors in user definitions in the database files

Description

The **tcbck** command audits the security state of the system by checking the installation of the files defined in the **/etc/security/sysck.cfg** file (the `sysck` database). Each file definition in the **/etc/security/sysck.cfg** file can include one or more attributes that describe proper installation. When invoked with no flags and with no parameters, the **tcbck** command prints a synopsis of its syntax.

The `tcbck` database usually defines all the files and programs that are part of the trusted computing base, but the root user or a member of the security group can choose to define only those files considered to be security-relevant.

Note: This command writes its messages to **stderr**.

Flags

Item	Description
-a	Adds or updates file definitions in the `sysck` database.
-d	Deletes file definitions from the `sysck` database.

Item	Description
-f *File*	Specifies that file definitions be read from *File*.
-i	Excludes filesystems under directories listed in the **treeck_nodir** attribute when the **tree** option is specified.
-l	(Lowercase L) Adds entries to the **sysck.cfg** file for **/dev/** files that the administrator would like registered with the Trusted Computing Base.
-n	Specifies the checking mode and indicates that errors are to be reported, but not fixed.
-o	Writes output to syslog.
-p	Specifies the checking mode and indicates that errors are to be fixed, but not reported.
-t	Specifies the checking mode and indicates that errors are to be reported with a prompt asking whether the error should be fixed.
-y	Specifies the checking mode and indicates that errors are to be fixed and reported.

Modes of Operation

The **tcbck** command has two modes of operation: check mode and update mode. A description of each mode follows.

Check Mode

In check mode, the **tcbck** command checks file definitions against the installed files. You can check all the file definitions in the `sysck` database (the **/etc/security/sysck.cfg** file) by specifying the **ALL** value, or all the files in the file system tree by specifying the **tree** value. If you prefer to check specific files, you can use the *Name* parameter to give the path names of individual files or the *Class* parameter to group several files into a logical group that is defined by a class name, such as audit. You must select one of the following: the **ALL** or **tree** values, or one or more files identified by the *Class* or *Name* parameter.

If the **tree** value is the selection criterion, all the files in the file system tree are checked to ensure that all the relevant files are defined in the `sysck` database. Files defined in the `tcbck` database are checked against their definitions. Files not in the `tcbck` database must *not*:

- Have the **trusted computing base** attribute set.
- Be **setuid** or **setgid** to an administrative ID.
- Be linked to a file in the`tcbck` database.
- Be a device special file.

If the **tcbck** command is running in check mode with both the **tree** value and the **-t** flag and an error occurs, the command provides an error message and prompts you for a decision on how or whether the error should be corrected. If you decide not to delete the file or turn off illegal permissions, you are prompted for a decision on updating the database. If you request an update, the system supplies missing information, such as the name of the file, the link, or the unregistered device name.

A flag (**-n**, **-p**, **-t**, **-y**) also must be included to specify check mode and identify the method of error handling. If there is a duplicate stanza in the **/etc/security/sysck.cfg** file, an error is reported, but not fixed.

Updating the Vital Product Database (VPD) involves defining the **type**, **checksum**, and **size** attributes of each file to the VPD manager. This information is used to verify a correct installation. If these attributes are not defined in **-f** *File*, they are computed when the program is installed or updated. The **checksum** attribute is computed with a method specifically defined for the VPD manager. Refer to “Fixing Errors” on page 4011 for more information on file attributes.

The only file definitions modified during an update are the new definitions that indicate a file is part of the trusted computing base (TCB). The *File* parameter is the stanza file that contains the file definitions in **tcbck** format, and is defined in the **/etc/security/sysck.cfg** file. When the update is complete, the files are checked against their file definitions in the stanza file and errors are fixed and reported.

Programs that require **setuid** or **setgid** privilege must be in the `tcbck` database, or these privileges will be cleared when the **tcbck** command runs in Check mode.

Update Mode

In update mode, the **tcbck** command adds (**-a**), deletes (**-d**), or modifies file definitions in the **/etc/security/sysck.cfg** file for the file specified by the *File* parameter, the *PathName* parameter, or the *Class* parameter. The *Class* parameter permits you to group several files into a logical group that is defined by a class name, such as audit. The **tcbck** command also deletes the specified stanzas from the **/etc/security/sysck.cfg** file.

In update mode, the **tcbck** command (**-l**) adds or modifies **/dev/** entry definitions in the **/etc/security/sysck.cfg** file for the specified **/dev** entry. This flag should be run by the administrator to add newly created devices that are trusted to the **sysck.cfg** file. If new devices are not added to the **sysck.cfg** file, the tree option produces warnings of unregistered devices.

The **-l** flag creates a stanza for each **/dev/** entry listed on the command line. The information for the stanza is taken from the current status of the **/dev** entry. The stanza includes:

Device name	/dev/ entry name
File type	Either **FILE**, **DIRECTORY**, **FIFO**, **SYMLINK**, **BLK_DEV**, **CHAR_DEV**, or **MPX_DEV**
Owner ID	Owner name
Group ID	Group name
Permissions	Read/write/execute permissions for owner, group and other. **SUID**, **SGID**, **SVTX** and **TCB** attribute bits
Target	If the file is a symbolic link, the target file will be listed.

File definitions to be added or modified with the **-a** flag can be specified on the command line or in a file as *Attribute*=*Value* statements. The following attributes can be used:

Item	Description
acl	The access control list for the file. If the value is **blank**, the **acl** attribute is removed. If no value is specified, the command computes a value, according to the format described in Access Control Lists.
class	The logical group of the file. A value must be specified, because it cannot be computed. If the value is **blank**, the **class** attribute is removed from the specified file stanza. The value is *ClassName* [*ClassName*].
checksum	The checksum of the file. If the value is **blank**, the **checksum** attribute is removed. If no value is specified, the command computes a value, according to the format given in the **sum** command. The value is the output of the **sum -r** command, including spaces.
group	The file group. If the value is **blank**, the **group** attribute is removed. If no value is specified, the command computes a value, which can be a group ID or a group name.
links	The hard links to this file. If the value is **blank**, the **links** attribute is removed. A value must be specified, because it cannot be computed. The value must be an absolute path name, expressed as *Path* [,*Path* ...].
mode	The File mode. If the value is **blank**, the **mode** attribute is removed. If no value is specified, the command computes a value, which can be an octal number or string (*rwx*), and have the **tcb**, **SUID**, **SGID**, and **SVTX** attributes.
owner	The file owner. If the value is **blank**, the **owner** attribute is removed. If no value is specified, the command computes a value, which can be a user ID or a user name.
program	The associated checking program for the file. If the value is **blank**, the **program** attribute is removed. A value must be specified, because it cannot be computed. The value must be an absolute path name. If flags are specified, the value should be expressed as *Path*, *Flag*.

Item	Description
symlinks	The symbolic links to the file. If the value is **blank**, the **symlinks** attribute is removed. A value must be specified, because it cannot be computed. The value must be an absolute path name, expressed as *Path* [,*Path*..].
size	The size of the file in bytes. If the value is **blank**, the **size** attribute is removed. If no value is specified, the command computes a value. The value is a decimal number.
source	The source for the file. If the value is **blank**, the **source** attribute is removed. If no value is specified, an empty file of the appropriate type is created. The value must be an absolute path name.
type	The type of file. This value cannot be **blank**. If no value is specified, the command computes a value, which can be the **FILE**, **DIRECTORY**, **FIFO**, **BLK_DEV**, **CHAR_DEV**, or **MPX_DEV** keywords.

You can add, delete, or modify the attributes of the **tcbck** command by creating or modifying a **sysck** stanza in the **/etc/security/sysck.cfg** file. The following attributes can be used:

Item	Description
checksum	An alternate checksum command to compute the checksum value of files. The system appends the name of each file to the command. If the value is **blank**, this alternate **checksum** attribute is removed. The value is the command string to be run on each file. The default string is **/usr/bin/sum -r <**.
setgids	An additional list of administrative groups to be checked for **setgid** programs that are not valid (groups with ID numbers greater than 200). If the value is **blank**, the **setgids** attribute is removed. The value is a comma separated list of group names.
setuids	An additional list of administrative users to be checked for **setuid** programs that are not valid (users with ID numbers greater than 200). If the value is **blank**, the **setuids** attribute is removed. The value is a comma separated list of user names.
treeck_nodir	A list of directories to be excluded from verification by the **tcbck** command. If the value is blank, the **treeck_nodir** attribute is removed. The value is a comma separated list of directories. File systems that exist under directories contained in this attribute are *not* excluded. Use the **-i** flag to exclude these file systems. Use this option only when the **tree** option is specified.
treeck_novfs	A list of file systems to be excluded from verification by the **tcbck** command during a check of an installed file system tree. If the value is **blank**, the **treeck_novfs** attribute is removed. The value is a comma separated list of file systems. Use this option only when the **tree** option is specified.

Refer to the **/etc/security/sysck.cfg** file for more information about these attributes and “Examples” on page 4012 for information about a typical stanza.

If *Attributes* are included without values, the command tries to compute the value from the file to be changed. The **type** attribute is mandatory, but the others do not need to be specified.

Fixing Errors

To fix errors, the **tcbck** command usually resets the attribute to the defined value. For the following attributes, the command modifies its actions as described:

Item	Description
checksum	Disables the file by clearing its access control list, but does not stop any further checks.
links	Creates any missing hard links. If a link exists to another file, the link is deleted.

Item	Description
program	Invokes the program, which must exist and have an absolute path name. A message is printed if an error occurs, but no additional action is taken.
size	Disables the file by clearing its access control list, but does not stop any further checks.
source	Copies the source file to the file identified by the *File* parameter. If the source is null, any existing file is deleted and a file of the correct **type** is created.
symlinks	Creates any missing symbolic links. If a link exists to another file, the link is deleted.
type	Disables the file by clearing its access control list, and stops any further checks.

If you used the **-t** flag with the **tcbck** command, you are prompted for a decision on fixing errors. If you answer `yes`, errors are fixed. If you give any other response, errors are not fixed.

Security

Access Control: This command grants execute (x) access only to the root user and members of the security group. The command should be setuid to the root user and have the **trusted computing base** attribute.

Files Accessed:

Mode	File
r	**/etc/passwd**
r	**/etc/group**
r	**/etc/security/user**
rw	**/etc/security/sysck.cfg**
x	**/usr/bin/aclget**
x	**/usr/bin/aclput**
x	**/usr/bin/sum**

Auditing Events:

Event	Information
TCBCK_Check	file, error, status
TCBCK_Update	file, function

Examples

1. To add the **/bin/boo** file with **acl**, **checksum**, **class**, **group**, **owner**, and **program** attributes to the `tcbck` database, type:

   ```
   tcbck -a /bin/boo acl checksum class=audit group owner\
   program=/bin/boock
   ```

   The resulting stanza will contain the attributes given previously, with computed values inserted for those attributes you do not define. The database will contain a stanza like the following:

   ```
   /bin/boo:
          acl =
          checksum = 48235
          class = audit
          group = system
          owner = root
          program = /bin/boock
          type = FILE
   ```

The attribute values are added to the installation definition but are not checked for correctness. The **program** attribute value comes from the command line, the **checksum** attribute value is computed with the **checksum** program, and all the others, except `acl`, are computed from the file i-node.

2. To indicate that the size of a file should be checked but not added to the database, because it can expand during installation, use the **VOLATILE** keyword, as in the following example for the **/etc/passwd** file:

```
/etc/passwd:
        type =  FILE
        owner = root
        group = system
        size  = 1234,VOLATILE
```

3. To delete the **/bin/boo** file definition from the `tcbck` database, type:

```
tcbck -d /bin/boo
```

4. To delete all definitions with a `class` of `audit` from the `tcbck` database, type:

```
tcbck -d audit
```

5. To check all the files in the `tcbck` database, and fix and report all errors, type:

```
tcbck -y ALL
```

6. To exclude the `/calvin` and the `/hobbes` file systems from verification during a security audit of an installed file system tree, type:

```
tcbck -a sysck treeck_novfs=/calvin,/hobbes
```

7. To exclude a directory from verification during a security audit, type:

```
tcbck -a sysck treeck_nodir=/home/john
```

8. To add `jfh` and `jsl` as administrative users and `developers` as an administrative group to be verified during a security audit of an installed file, type:

```
tcbck -a sysck setuids=jfh,jsl setgids=developers
```

9. To create/modify **sysck.cfg** stanza entries for the newly created **/dev** entries `foo` and `bar`, type:

```
tcbck -l /dev/foo /dev/bar
```

Note: By adding these entries you are registering them as part of the Trusted computing base.

Attention: Although the special characters "$" and "?" are allowed in this routine, using them in filenames may result in potential problems such as ambiguous files.

Files

Item	Description
/usr/bin/tcbck	Specifies the path to the **tcbck** command.
/etc/security/sysck.cfg	Specifies the path to the system configuration database.

tcopy Command

Purpose

Copies a magnetic tape.

Syntax

tcopy *Source* [*Destination*]

Description

The **tcopy** command copies magnetic tapes. Source and target file names are specified by the *Source* and *Destination* parameters. The **tcopy** command assumes that there are two tape marks at the end of the tape, and it ends when it finds the double file marks. With only a source tape specified, the **tcopy** command prints information about the size of records and tape files

Examples

To copy from one streaming tape to a 9-track tape, enter:

```
tcopy /dev/rmt0 /dev/rmt8
```

Files

Item	Description
/usr/bin/tcopy	Contains the **tcopy** command.

tcpdump Command

Purpose

Dumps traffic on a network

Syntax

tcpdump [**-a**] [**-A**] [**-B** *buffer_size*] [**-d**] [**-D**] [**-e**] [**-f**] [**-l**] [**-K**] [**-L**] [**-M** *secret*] [**-r** *file*]][**-n**] [**-N**] [**-O**] [**-p**][**-q**] [**-Q** [**-V**]] [**-R**] [**-S**] [**-t**] [**-T**][**-u**] [**-U**] [**-v**] [**-x**] [**-X**] [-c *count*][**-C** *file_size*] [**-F** *file*] [**-G** *rotate_seconds*] [**-i** *interface*] [**-s** *snaplen*] [**-w** *file*][**-E** *addr*] [**-y** *datalinktype*] [**-z** *command*] [**-Z** *user*] [*expression*]

Description

The `tcpdump` command prints the headers of packets on a network interface that match the boolean expression. You can run the command with the `-w` flag to save the packet data in a file for further analysis. You can also run the command with the `-r` flag to read data from a saved packet file instead reading the packets from a network interface. In all cases, only packets that match expression is processed by the `tcpdump` command.

If it is not run with the `-c` flag, `tcpdump` continues capturing packets until it is interrupted by a SIGINT signal (typically control-C) or a SIGTERM signal (typically the `kill(1)` command). If `tcpdump` is run with the `-c` flag, it captures the packets until it is interrupted by a SIGINT or SIGTERM signal or the specified number of packets have been processed.

The `tcpdump` command returns the following counts after capturing all the packets:

packets "received by filter"
Counts all packets regardless of whether they were matched by the filter expression.

packets "dropped by kernel"
The number of packets that were dropped, due to a lack of buffer space.

Allowable Primitives

dst host host
True if the IPv4/v6 destination field of the packet is host, which may be either an address or a name.

src host host
True if the IPv4/v6 source field of the packet is host.

host host
True if either the IPv4/v6 source or destination of the packet is host. Any of the above host expressions can be prepended with the keywords, ip, arp, rarp, or ip6 as in:ip host host which is equivalent to:

```
ether proto \ip and host host
```

If host is a name with multiple IP addresses, each address is checked for a match.

ether dst ehost
True if the ethernet destination address is `ehost`. Ehost may be either a name from `/etc/ethers` or a number (see ethers(3N) for numeric format).

ether src ehost
True if the ethernet source address is `ehost`.

ether host ehost
True if either the ethernet source or destination address is `ehost`.

gateway host
True if the packet used host as a gateway. For example, the ethernet source or destination address was host but neither the IP source nor the IP destination was host. Host must be a name and must be found both by the machine's host-name-to-IP-address resolution mechanisms (host name file, DNS, NIS, etc.) and by the machine's host-name-to-Ethernet-address resolution mechanism (/etc/ethers, and so on). An equivalent expression is ether host ehost and not host host which can be used with either names or numbers for host /ehost. This syntax does not work in IPv6-enabled configuration at this moment.

dst net net
True if the IPv4/v6 destination address of the packet has a network number of net.

src net net
True if the IPv4/v6 source address of the packet has a network number of net.

net net
True if either the IPv4/v6 source or destination address of the packet has a network number of net.

net net mask netmask
True if the IP address matches net with the specific netmask. This might be qualified with src or dst. This syntax is not valid for IPv6 net.

net net/len
True if the IPv4/v6 address matches net with a netmask len bits wide. May be qualified with src or dst.

dst port port
True if the packet is ip/tcp, ip/udp, ip6/tcp orip6/udp and has a destination port value of port. The port can be a number or a name used in `/etc/services` (see tcp(4P) and udp(4P)). If a name is used, both the port number and protocol are checked. If a number or ambiguous name is used, only the port number is checked (For example, dst port 513 prints both tcp/login traffic and udp/who traffic, and port domain prints both tcp/domain and udp/domain traffic).

src port port
True if the packet has a source port value of port.

port port
True if either the source or destination port of the packet is port. Any of the above port expressions can be prepended with the keywords, tcp or udp, as in: tcp src port port which matches only tcp packets whose source port is port.

less length
True if the packet has a length less than or equal to length. This is equivalent to len <= length.

greater length
True if the packet has a length greater than or equal to length. This is equivalent to: len >= length.

ip proto protocol
True if the packet is an IP packet of protocol type protocol. Protocol can be a number or one of the names icmp, icmp6, igmp, igrp, pim, ah, esp, vrrp, udp, or tcp. Note that the identifiers tcp, udp, and icmp are also keywords and must be escaped via backslash (\), which is \\ in the C-shell. Note that this primitive does not chase the protocol header chain.

ip6 proto protocol
True if the packet is an IPv6 packet of protocol type protocol. Note that this primitive does not chase the protocol header chain.

ip6 protochain protocol
True if the packet is IPv6 packet, and contains protocol header with type protocol in its protocol header chain. For example, ip6 protochain 6 matches any IPv6 packet with TCP protocol header in the protocol header chain. The packet may contain, for example, authentication header, routing header, or hop-by-hop option header, between IPv6 header and TCP header. The Berkeley Packet Filter (BPF) code emitted by this primitive is complex and cannot be optimized by BPF optimizer code in `tcpdump`, so this can be somewhat slow.

ip protochain protocol
Equivalent to ip6 protochain protocol. But, this is used for Ipv4.

ether broadcast
True if the packet is an ethernet broadcast packet. The ether keyword is optional.

ip broadcast
True if the packet is an IPv4 broadcast packet. It checks for both the all-zeroes and all-ones broadcast conventions, and looks up the subnet mask on the interface on which the capture is being done.

If the subnet mask of the interface on which the capture is being done is not available, for example, because the interface on which capture is being done has no netmask this check does not work correctly.

ether multicast
True if the packet is an ethernet multicast packet. The ether keyword is optional. This is shorthand for `ether[0] & 1 != 0`.

ip multicast
True if the packet is an IP multicast packet.

ip6 multicast
True if the packet is an IPv6 multicast packet.

ether proto protocol
True if the packet is of ether type protocol.Protocol can be a number or one of the names ip, ip6, arp, rarp, atalk, aarp, decnet, sca, lat, mopdl, moprc, iso, stp, ipx, or netbeui. Note that these identifiers are also keywords and must be escaped via backslash (\).

[In the case of FDDI (e.g., `fddi protocol arp'), Token Ring (e.g., `tr protocol arp'), and IEEE 802.11 wireless LANS (e.g., `wlan protocol arp'), for most of those protocols, the protocol identification comes from the 802.2 Logical Link Control (LLC) header, which is usually layered on top of the FDDI, Token Ring, or 802.11 header. When filtering for most protocol identifiers on FDDI, Token Ring, or 802.11, tcpdump checks only the protocol ID field of an LLC header in so-called SNAP format with an Organizational UnitIdentifier (OUI) of 0x000000, for encapsulated Ethernet; it doesn't check whether the packet is in SNAP format with an OUI of 0x000000. The exceptions are:

iso
`tcpdump` checks the DSAP (Destination Service Access Point) and SSAP (Source Service Access Point) fields of the LLC header.

stp and netbeui
`tcpdump` checks the DSAP of the LLC header.

atalk
`tcpdump` checks for a SNAP-format packet with an OUI of 0x080007 and the AppleTalk etype.

In the case of Ethernet, `tcpdump` checks the Ethernet type field for most of those protocols. The exceptions are:

iso, sap, and netbeui
`tcpdump` checks for an 802.3 frame and then checks the LLC header as it does for FDDI, Token Ring, and 802.11.

atalk
`tcpdump` checks both for the AppleTalk etype in an Ethernet frame and for a SNAP-format packet as it does for FDDI, Token Ring, and 802.11.

aarp
`tcpdump` checks for the AppleTalk ARP etype in either an Ethernet frame or an 802.2 SNAP frame with an OUI of 0x000000;

ipx
`tcpdump` checks for the IPX etype in an Ethernet frame, the IPX DSAP in the LLC header, the 802.3-with-no-LLC-header encapsulation of IPX, and the IPX etype in a SNAP frame.

decnet src host
True if the DECNET source address is host, which may be an address of the form 10.123, or a DECNET host name. [DECNET host name support is only available on Ultrix systems that are configured to run DECNET.]

decnet dst host
True if the DECNET destination address is host.

decnet host host
True if either the DECNET source or destination address is host.

ifname interface
True if the packet was logged as coming from the specified interface.

on interface
Synonymous with the ifname modifier.

rnr num
True if the packet was logged as matching the specified PF rule number (applies only to packets logged by OpenBSD's pf(4)).

rulenum num
Synonomous with the rnr modifier.

reason code
True if the packet was logged with the specified PF reason code. The known codes are: match, bad-offset, fragment, short, normalize, and memory (applies only to packets logged by OpenBSD's `pf(4)`).

action act
True if PF took the specified action when the packet was logged. Known actions are: pass and block (applies only to packets logged by OpenBSD's `pf(4)`)

netbeui
ip, ip6, arp, rarp, atalk, aarp, decnet, iso, stp, ipx.

Abbreviations for:

```
ether proto p
```

where *p* is one of the above protocols.

lat, moprc, mopdl

Abbreviations for:

```
ether proto p
```

where *p* is one of the above protocols. Note that `tcpdump` does not currently know how to parse these protocols.

vlan [*vlan_id*]
True if the packet is an IEEE 802.1Q VLAN packet. If *vlan_id* is specified, only the packets that have the specified *vlan_id* are true. Note that the first `vlan` keyword encountered in expression changes the decoding offsets for the remainder of expression on the assumption that the packet is a VLAN packet.

tcp, udp, icmp
Abbreviations for:

```
ip proto p or ip6 proto p
```

where *p* is one of the above protocols.

iso proto protocol
True if the packet is an OSI packet of protocol type protocol. Protocol can be a number or one of the names clnp, esis, or isis.

clnp, esis, isis
Abbreviations for:

- iso proto p

where *p* is one of the above protocols.

l1, l2, iih, lsp, snp, csnp, psnp
Abbreviations for IS-IS PDU types.

vpi *n*
True if the packet is an ATM packet, for SunATM on Solaris, with a virtual path identifier of *n*.

vci *n*
True if the packet is an ATM packet, for SunATM on Solaris, with a virtual channel identifier of *n*.

lane
True if the packet is an ATM packet, for SunATM on Solaris, and is an ATM LANE packet. Note that the first lane keyword encountered in expression changes the tests done in the remainder of expression on the assumption that the packet is either a LANE emulated Ethernet packet or a LANE LE Control packet. If lane isn't specified, the tests are done under the assumption that the packet is an LLC-encapsulated packet.

llc
True if the packet is an ATM packet, for SunATM on Solaris, and is an LLC-encapsulated packet.

oamf4s
True if the packet is an ATM packet, for SunATM on Solaris, and is a segment OAM F4 flow cell (VPI=0 & VCI=3).

oamf4e
True if the packet is an ATM packet, for SunATM on Solaris, and is an end-to-end OAM F4 flow cell (VPI=0 & VCI=4).

oamf4
True if the packet is an ATM packet, for SunATM on Solaris, and is a segment or end-to-end OAM F4 flow cell (VPI=0 & (VCI=3 | VCI=4)).

oam
True if the packet is an ATM packet, for SunATM on Solaris, and is a segment or end-to-end OAM F4 flow cell (VPI=0 & (VCI=3 | VCI=4)).

metac
True if the packet is an ATM packet, for SunATM on Solaris, and is on a meta signaling circuit (VPI=0 & VCI=1).

bcc
True if the packet is an ATM packet, for SunATM on Solaris, and is on a broadcast signaling circuit (VPI=0 & VCI=2).

sc
True if the packet is an ATM packet, for SunATM on Solaris, and is on a signaling circuit (VPI=0 & VCI=5).

ilmic
True if the packet is an ATM packet, for SunATM on Solaris, and is on an ILMI circuit (VPI=0 & VCI=16).

connectmsg
True if the packet is an ATM packet, for SunATM on Solaris, and is on a signaling circuit and is a Q.2931 Setup, Call Proceeding, Connect, Connect Ack, Release, or Release Done message.

metaconnect
True if the packet is an ATM packet, for SunATM on Solaris, and is on a meta signaling circuit and is a Q.2931 Setup, Call Proceeding, Connect, Release, or Release Done message.

expr relop expr
True if the relation holds, where relop is one of >, <, >=, <=, =, !=, and expr is an arithmetic expression composed of integer constants (expressed in standard C syntax), the normal binary operators [+, -, *, /, &, |], a length operator, and special packet data accessors. To access data inside the packet, use the following syntax:

```
proto [ expr : size ]
```

Proto is one of ether, fddi, tr, wlan, ppp, slip, link, ip, arp, rarp, tcp, udp, icmp or ip6, and indicates the protocol layer for the index operation. (ether, fddi, wlan, tr, ppp, slip and link all refer to the link layer.) Note that tcp, udp and other upper-layer protocol types only apply to IPv4, not IPv6 (this will be fixed in the future). The byte offset, relative to the indicated protocol layer, is given by expr. Size is optional and indicates the number of bytes in the field of interest; it can be either one, two, or four, and defaults to one. The length operator, indicated by the keyword len, gives the length of the packet.

For example, `ether[0] & 1 != 0` catches all multicast traffic. The expression `ip[0] & 0xf != 5` catches all IP packets with options. The expression `ip[6:2] & 0x1fff = 0` catches only unfragmented datagrams and frag zero of fragmented datagrams. This check is implicitly applied to the tcp and udp index operations. For instance, tcp[0] always means the first byte of the TCP header, and never means the first byte of an intervening fragment.

Some offsets and field values may be expressed as names rather than as numeric values. The following protocol header field offsets are available: icmptype (ICMP type field), icmpcode (ICMP code field), and tcpflags (TCP flags field).

The following ICMP type field values are available: icmp-echoreply, icmp-unreach, icmp-sourcequench, icmp-redirect, icmp-echo, icmp-routeradvert, icmp-routersolicit, icmp-timxceed, icmp-paramprob, icmp-tstamp, icmp-tstampreply, icmp-ireq, icmp-ireqreply, icmp-maskreq, icmp-maskreply.

The following TCP flags field values are available: tcp-fin, tcp-syn, tcp-rst, tcp-push, tcp-ack, tcp-urg.

Combining Primitives

A parenthesized group of primitives and operators (parentheses are special to the Shell and must be escaped).

```
          Negation (`!' or `not').

          Concatenation (`&&' or `and').

          Alternation (`||' or `or').
```

Negation has highest precedence. Alternation and concatenation have equal precedence and associate left to right. Note that explicits and tokens, not juxtaposition, are now required for concatenation.

If an identifier is given without a keyword, the most recent keyword is assumed. For example, `not host vs and ace` is short for `not host vs and host ace` which should not be confused with `not ( host vs or ace )`

Expression arguments can be passed to `tcpdump` as either a single argument or as multiple arguments, whichever is more convenient. Generally, if the expression contains Shell metacharacters, it is easier to pass it as a single, quoted argument. Multiple arguments are concatenated with spaces before being parsed.

Flags

Item	Description
-a	Attempts to convert network and broadcast addresses to names.
-A	Prints each packet (minus its link level header) in ASCII. Handy for capturing web pages.
-B *buffer_size*	Indicates the buffer size in kilobytes. Smaller values are accepted. If the buffer size is smaller than the minimum value that is set by the BPF, the actual buffer size is ignored and the value that is set by the Berkeley Packet Filter (BPF) is used. If the **-B** option is not specified, the buffer size defaults to 32,768.
-c *Count*	Exits after receiving *Count* packets.
-C *file_size*	Before writing a raw packet to a *savefile*, check whether the file is currently larger than *file_size* and, if so, close the current savefile and open a new one. Save files after the first *savefile* has the name specified with the **-w** flag, with a number after it, starting at 2 and continuing upward. The units of *file_size* are millions of bytes (1,000,000 bytes, not 1,048,576 bytes).
-d	Dumps the compiled packet-matching code to standard output, then stops.
-D	Prints the list of the network interfaces available on the system and on which **tcpdump** can capture packets. For each network interface, a number and an interface name (possibly followed by a text description of the interface) is printed. The interface name or the number can be supplied to the *-i* flag to specify an interface on which to capture.
-dd	Dumps packet-matching code as a C program fragment.
-ddd	Dumps packet-matching code as decimal numbers (preceded with a count).
-e	Prints the link-level header on each dump line.

Item	Description
-E *addr*	Use `spi@ipaddr algo:secret` for decrypting IPsec ESP packets that are addressed to *addr* and contain Security Parameter Index value `spi`. This combination may be repeated with comma or newline separation. **Note:** Setting the secret for IPv4 ESP packets is now supported. Algorithms may be `des-cbc`, `3des-cbc`, `blowfish-cbc`, `rc3-cbc`, `cast128-cbc`, or none. The default is `des-cbc`. The ability to decrypt packets is only present if **libcrypto** is installed and is in LIBPATH. `secret` is the ASCII text for ESP secret key. If preceeded by `0x`, then a hex value is read. The option assumes RFC2406 ESP, not RFC1827 ESP. The option is for debugging purposes only and the use of this option with a true secret key is discouraged. By presenting the IPsec secret key onto command line you make it visible to others, via **ps(1)** and other occasions. In addition to the above syntax, the **tcpdump** command might use the syntax file name to read the specified file. The file is opened upon receiving the first ESP packet, so any special permissions that **tcpdump** may have been given, should already have been given up.
-f	Prints foreign IPv4 addresses numerically rather than symbolically. The test for foreign IPv4 addresses is done by using the IPv4 address and netmask of the interface on which capture is being performed. This option does not work correctly if that address or netmask is not available.
-F *file*	Use *file* as input for the filter expression. An additional expression given on the command line is ignored.
-G *rotate_seconds*	Rotates the dump file that is specified with the **-w** option every *rotate_seconds* seconds. If used in conjunction with the **-C** option, file names take the form of *file <count>*, if the value specified in the *size* variable is reached first. Otherwise, the **tcpdump** command rotates the file when the value specified in the *rotate_seconds* variable is elapsed.

Item	Description
-i *interface*	Listens on *interface*. If unspecified, **tcpdump** searches the system *interface* list for the lowest numbered, configured up *interface* (excluding loopback). Ties are broken by choosing the earliest match. An *interface* number as printed by `-D` flag can be used as the *interface* argument.
-K	Skips verification of TCP checksum on interfaces that perform TCP checksum calculation in hardware. If this flag is not used, all outgoing TCP checksums are flagged as bad.
-l	Makes *stdout* line buffered. Useful if you want to see the data while capturing it. For example: `tcpdump -l | tee dat` `  or` `tcpdump -l  >  dat & tail -f dat`
-L	Lists the known data link types for the interface and exits.
-m *module*	Loads SMI MIB module definitions from the *module* file. This option can be used several times to load several MIB modules into **tcpdump**.
-M	Uses secret as a shared secret for validating the digests that are found in TCP segments by using the **TCP-MD5** option (Request for Comment (RFC) 2385).
-n	Blocks converting the host addresses, and the port numbers to names.
-N	Omits printing domain name qualification of host names. For example, **tcpdump** prints `nic` instead of `nic.ddn.mil`.
-O	Keeps **tcpdump** from running the packet-matching code optimizer. This is useful only if you suspect a bug in the optimizer.
-p	Stops putting the interface into promiscuous mode. Note that the interface might be in promiscuous mode for some other reason; hence, **-p** cannot be used as an abbreviation for `ether host {local-hw-addr}` or `ether broadcast`.
-q	Quick output. Prints less protocol information so output lines are shorter.
-Q	Enables filtered system tracing for the recorded packets. You must run the AIX trace daemon to record the selected system events that are related to the network communication subsystem.
-r *file*	Read packets from *file* (which was created with the **-w** option). Standard input is used if *file* is "-".

Item	Description
-R	Assumes ESP/AH packets are based on old specification. (RFC1825 to RFC1829). If specified, **tcpdump** does not print replay prevention field. Since there is no protocol version field in ESP/AH specification, **tcpdump** cannot deduce the version of ESP/AH protocol.
-S	Prints absolute rather than relative TCP sequence numbers.
-s *snaplen*	Snarf *snaplen* bytes of data from each packet rather than the default of 68. 68 bytes is adequate for IP, ICMP, TCP and UDP but may truncate protocol information from name server and NFS packets (see below). Packets truncated because of a limited snapshot are indicated in the output with [|*proto*], where *proto* is the name of the protocol level at which the truncation has occurred. Note that taking larger snapshots increases the amount of time it takes to process packets and effectively decreases the amount of packet buffering. This can cause packets to be lost. You should limit *snaplen* to the smallest number that captures the protocol information you are interested in. Setting *snaplen* to 0 means use the required length to catch whole packets.
-T	Forces packets selected by *expression* to be interpreted the specified type. Currently known types are cnfp (Cisco NetFlow protocol), rpc (Remote Procedure Call), rtp (Real-Time Applications protocol), rtcp (Real-Time Applications control protocol), snmp (Simple Network Management Protocol), tftp (Trivial File Transfer Protocol), vat (Visual Audio Tool), and wb (distributed White Board).
-t	Omits the printing of a timestamp on each dump line.
-tt	Prints an unformatted timestamp on each dump line.
-ttt	Prints a delta (in microseconds) between current and previous line on each dump line.
-tttt	Prints a timestamp in default format proceeded by date on each dump line.
-ttttt	Prints a delta (in microseconds) between the current and the first line on each dump line.
-u	Prints undecoded NFS handles.
-U	Make output saved via the **-w** option, for example, "packet- buffered." As each packet is saved, it is written to the output file, rather than being written only when the output buffer fills.

Item	Description
-v	Specifies slightly more verbose output. For example, the time to live, identification, total length and options in an IP packet are printed. Also enables additional packet integrity checks such as verifying the IP and ICMP header checksum.
-vv	Even more verbose output than **-v**. For example, additional fields are printed from NFS and reply packets are fully decoded.
-vvv	Even more verbose output than **-vv**. For example, telnet SB ... SE options are printed in full. With `-X` Telnet options are printed in hex as well.
-V	Sets the socket debug flag (the **SO_DEBUG** socket option) and the trace level on sockets. This flag must be used along with the **-Q** flag.
-w *file*	Writes the raw packets to *file* rather than parsing and printing them out. They can later be printed with the **-r** flag. Standard output is used if *File* is "-".
-x	Prints each packet (minus its link level header) in hexadecimal. The smaller of the entire packet or snaplen bytes is printed. Note that this is the entire link-layer packet, so for link layers that pad (e.g. Ethernet), the padding bytes is also printed when the higher layer packet is shorter than the required padding.
-xx	Prints each packet, including its link level header, in hexadecimal.
-X	Prints each packet (minus its link level header) in hexadecimal and ASCII. This is very handy for analyzing new protocols.
-y *datalinktype*	Sets the data link type to use while capturing packets to *datalinktype*.
-z *command*	When used in conjunction with the **-C** or **-G** option, causes the **tcpdump** command to run the specified command on the *savefile*. For example, specifying **-z gzip** or **-z bzip2** compresses each *savefile* by using the **gzip** or **bzip2** command. **Note:** The **tcpdump** command runs the **-z** command in parallel to the capture by using the lowest priority so that this does not disturb the capture process.
-Z *user*	Runs the **tcpdump** command with the system privileges of the specified user.

Parameters

expressions

Selects the packets that are to be dumped. If an expression is provided, only the packets for which the expressions is `true` are dumped; otherwise, all the packets on the net are dumped.

The expression consists of one or more primitives. Primitives usually consist of an id (name or number) preceded by one or more qualifiers. There are three different kinds of qualifier:

`type` qualifiers say what type of primitive the id name or number refers to. Possible types are host, net and port. For example, `host foo', `net 128.3', `port 20'. If there is no type qualifier, host is assumed.

`dir` qualifiers specify a particular transfer direction to and/or from id. Possible directions are src, dst, src or dst and src and dst. If there is no dir qualifier, src or dst is assumed. For some link layers, such as SLIP and for some other device types, the inbound and outbound qualifiers can be used to specify a desired direction.

`proto` qualifiers restrict the match to a particular protocol. Possible protos are fddi, tr, wlan, ip, ip6, arp, rarp, decnet, tcp and udp. If there is no proto qualifier, all protocols consistent with the type are assumed.

`fddi` is an alias for ether. The parser treats it as meaning "the data link level used on the specified network interface." FDDI headers contain Ethernet-like source and destination addresses, and often contain Ethernet-like packet types, so you can filter on these FDDI fields just as with the analogous Ethernet fields. FDDI headers also contain other fields, but they cannot be named in a filter expression.

Like `fddi`, `tr` and `wlan` are aliases for ether. The previous paragraph's statements about FDDI headers also apply to Token Ring and 802.11 wireless LAN headers. For 802.11 headers, the destination address is the DA field and the source address is the SA field; the BSSID, RA, and TA fields aren't tested.

In addition to the above, there are some special `primitive' keywords that don't follow the pattern: gateway, broadcast, less, greater and arithmetic expressions. All of these are described below.

More complex filter expressions are built by using the words `and`, `or`, and `not` to combine primitives.

Environment Variables

LIBPATH environmental variable must be set or `libcrypto` library should be in `/usr/lib` for the `-E` flag to work. For example:

```
ksh$ LIBPATH=/opt/freeware/lib tcpdump -E"algo:secret"
```

Exit Status

Item	Description
0	Success
non-zero	Error

Security

Reading packets from a network interface requires read access to `/dev/bpf*`, which is typically root-only. Reading packets from a file does not require any special privileges except file read permission.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To print all packets arriving at or departing from sundown, enter:

```
tcpdump host sundown
```

2. To print traffic between helios and either hot or ace, enter:

```
tcpdump host helios and \( hot or ace \)
```

3. To print all IP packets between ace and any host except helios, enter:

```
tcpdump ip host ace and not helios
```

4. To print all traffic between local hosts and hosts at Berkeley, enter:

```
tcpdump net ucb-ether
```

5. To print all ftp traffic through internet gateway snup, enter:

```
tcpdump 'gateway snup and (port ftp or ftp-data)'
```

   Note: The expression is quoted to prevent the shell from mis-interpreting the parentheses.

6. To print traffic neither sourced from nor destined for local hosts (if you gateway to one other net, this should never make it onto your local net), enter:

```
tcpdump ip and not net localnet
```

7. To print the start and end packets (the SYN and FIN packets) of each TCP conversation that involves a non-local host, enter:

```
tcpdump 'tcp[tcpflags] & (tcp-syn|tcp-fin) != 0 and not src and d dst net localnet'
```

8. To print IP packets longer than 576 bytes sent through gateway snup, enter:

```
tcpdump 'gateway snup and ip[2:2] > 576'
```

9. To print IP broadcast or multicast packets that were not sent via ethernet broadcast or multicast, enter:

```
tcpdump 'ether[0] & 1 = 0 and ip[16] >= 224'
```

10. To print all ICMP packets that are not echo requests/replies (for instance, not ping packets), enter:

```
tcpdump 'icmp[icmptype] != icmp-echo and icmp[icmptype] != icmp-e choreply'
```

Standard Error

All errors and warnings are sent to `stderr`.

Limitations

A packet trace that crosses a Daylight Saving Time change gives skewed time stamps (the time change is ignored).

Filter expressions on fields other than those in Token Ring headers handles the source-routed Token Ring packets incorrectly.

Filter expressions on fields other than those in 802.11 headers handles the 802.11 data packets with both To DS and From DS set incorrectly.

ip6 proto should chase header chain, but at this moment it does not. ip6 protochain is supplied for this behavior.

Arithmetic expression against transport layer headers, like tcp[0], does not work against IPv6 packets. It only looks at IPv4 packets.

Packet tracing does not work in WPAR environment because the underlying BPF driver is not WPAR aware.

Files

Item	Description
/usr/sbin/tcpdump	Location of the tcpdump command.
/usr/lib/libpcap.a	
/dev/bpf*	
/opt/freeware/lib/libcrypto.a(libcrypto.so)	Optional

tcptr Command

Purpose

Configures or displays TCP Traffic Regulation (TR) policy information to control the maximum incoming socket connections for ports.

Syntax

tcptr -add < *start port* > < *end port* > < *max connection* > [*divisor*]

tcptr -delete < *start port* > < *end port* >

tcptr -show

Description

The **tcptr** command assigns a maximum limit of incoming TCP connections to a given network port or a range of ports. You can run this command to add new pools of connection resources to be shared collectively by incoming socket requests remotely accessing the AIX TCP-layer.

The system automatically ensures that resources are shared across multiple remote IP addresses that are attempting to connect through TCP to a specific port. Root users can control system resources related to TCP Traffic Regulation (TR).

Notes:

- By default, the **tcptr** command is not enabled.
- The **tcptr** command does not limit the rate of connections from a particular IP address. The total pool of connections from any client for a specific port or port-range is controlled.
- When the limit is reached, the connection to the server is lost. Message is not logged and the connection is lost, because the server is regulating the traffic and the system is following the instructions from the server.
- The TCP TR policies that are added by using the **tcptr** command are not activated until the **tcptr_enable** network attribute is set to a value of 1 by using the **no** command. These policies automatically persist after a system restart, but they are not activated until the network flag is enabled by using the **-p** flag as specified in the following command:

```
no -p -o tcptr_enable=1
```

Flags

Item	Description
-add	Adds new TCP TR policies to the system. You should specify the maximum allowable connections for the current policy, the start port, and the end port with the **-add** flag. The start port and the end port can be the same port when a port range is not specified. Optionally, you can specify a divisor to allow a greater diversity of resource sharing on the pool of available TCP connections.
-delete	Deletes existing TCP TR policies that are defined for the system. This flag requires the user specify the maximum allowable connections for the current policy, the start port, and the end port (can be the same as start port if not specifying a port-range).
-show	Displays all existing TCP TR policies defined on the system. You might use the **-show** flag to see the active policies before you use the **-delete** flag.

Parameters

Item	Description
max connection	Specifies the maximum incoming TCP connections for the given TR policy.
start port	Specifies the beginning port for the current TR policy.
end port	Specifies the end port for the current TR policy. If the port is a range, the value specified must be larger than the start port. If the TR policy is for a single port, the value specified must be equal to the value specified for the start port.
divisor	Specifies a divisor to compare the number of available incoming TCP connections with the number of consumed incoming TCP connections for an IP, and corresponds to a division of the overall available connections by a power of two. The divisor is the power of two that is used in the division. This parameter is optional, and if it is not specified, the default value is one. In that case, half of the number of available connections are used.

Algorithm for `tcptr` traffic regulation

When a new connection request is received, the **`tcptr`** command uses the following algorithm to allow or deny the new socket connections:

```
If a new connection request is received and (N-X) = 0, the request is rejected.
If a new connection request is received and (N-X) > 0 and
the request is from a source that already has connections
with this port(range), then:
     if X+1 < [(N-X)/2^divisor] then
        Allow the new connection
     else
        Deny the new connection
```

N
Maximum allowed connections for a port (range).

X
Currently used connections for a particular IP address.

divisor
Optional, default value is 1 (one).

Examples

1. To add a TCP Traffic Regulation Policy that covers only TCP port 23, and to set a maximum incoming connection pool of 256 with an available connections divisor of 3, enter the following command:

```
# tcptr -add 23 23 256 3
```

2. To add a TCP Traffic Regulation Policy that covers a TCP port that ranges from 5000 to 6000, and to set a maximum incoming connection pool of 5000 with an available connections divisor of 2, enter the following command:

```
# tcptr -add 5000 6000 5000 2
```

3. To show TCP Traffic Regulation Policies set for the system, enter the following command:

```
# tcptr -show
```

4. To delete the TCP Traffic Regulation Policy that covers a TCP port that ranges from 5000 to 6000, enter the following command:

```
# tcptr -delete 5000 6000
```

5. To add a TCP Traffic Regulation Policy with the IP address 10.20.30.1 that makes 256/2^3=32 connections to port 80, enter the following command:

```
tcptr -add 80 80 256 3
```

   In this case, the next connection attempt from this IP address to port 80 is rejected and a TCP RST is received.

tcsd Daemon

Purpose

Manages trusted computing resources.

Syntax

tcsd [**-f**]

Description

TrouSerS is an open source Trusted Computing Group Software Stack (TSS) that is released under the Common Public License. TrouSerS aims to be compliant with 1.1b and 1.2 TSS specifications.

According to the TSS specification, the **tcsd** daemon is a user-space daemon that must be the only portal to the Trusted Platform Module (TPM) device driver. At boot time, the system must start the **tcsd** daemon, and then the **tcsd** daemon communicates with the TPM device driver. From that point onwards, all requests to the TPM are routed through the TSS. The **tcsd** daemon manages the TPM resources and handles both local and remote requests from the TCG Service Provider (TSP).

Flags

Item	Description
-f	Runs the **tcsd** daemon in the foreground.

Access Control

There are two types of access control for the **tcsd** daemon: access to the daemon socket and access to specific commands that are internal to the **tcsd** daemon.

Access to the **tcsd** daemon port is controlled by the system administrator by using firewall rules.

Access to individual commands that are internal to the **tcsd** daemon is configured by the **remote_ops** directive of the **tcsd** configuration file. Each function call in the TCG Core Services (TCS) API is reachable by a unique ordinal. Each labeled **remote_op** directive defines a set of ordinals (usually more than one) that are necessary to accomplish the operation. For example, the **random** operation enables the ordinals for opening and closing a context, calling the **TCS_StirRandom**, the **TCS_GetRandom**, and the **TCS_FreeMemory** functions. By default, connections from a local host allow any ordinals.

Data Files

TSS applications have access to the following types of persistent storage:

User persistent storage
User persistent storage has a lifetime similar to the lifetime of the application that uses it; therefore, it is destroyed when an application exits. User persistent storage is controlled by the TSP of the application. By default, user persistent storage files are stored as `/var/tss/lib/tpm/user.{pid}`.

System persistent storage
System persistent storage is controlled by the TCS and stays valid across application lifetimes, the **tcsd** daemon restarts, and system resets. The data registered in system persistent storage remains valid until an application requests its removal. By default, system persistent storage files are stored as `/var/tss/lib/tpm/system.data`. The system persistent storage file is initially created when ownership of the TPM is received.

Files

Item	Description
/etc/security/tss/tcsd.conf	Contains all the default options and configurations for the **tcsd** daemon.

Conforming To

The **tcsd** daemon conforms to the TSS specification Version 1.10 Golden.

tctl Command

Purpose

Gives subcommands to a streaming tape device.

Syntax

tctl [**-f** *Device*] [**eof** | **weof** | **fsf** | **bsf** | **fsr** | **bsr** | **rewind** | **offline** | **rewoffl** | **erase** | **retension** | **reset** | **status**] [*Count*]

tctl [**-b** *BlockSize*] [**-f** *Device*] [**-p** *BufferSize*] [**-v**] [**-n**] [**-B**] { **read** | **write** }

Description

The **tctl** command gives subcommands to a streaming tape device. If you do not specify the *Device* variable with the **-f** flag, the **TAPE** environment variable is used. If the environment variable does not exist, the **tctl** command uses the **/dev/rmt0.1** device. (When the **tctl** command gives the **status** subcommand, the default device is **/dev/rmt0**.) The *Device* variable must specify a raw (not block) tape device. The *Count* parameter specifies the number of end-of-file markers, number of file marks, or number of records. If the *Count* parameter is not specified, the default count is 1.

Subcommands

Item	Description
eof or **weof**	Writes the number of end-of-file markers specified by the *Count* parameter at the current position on the tape. On an 8 mm tape drive, an end-of-file marker can be written in three places: • Before blank tape • Before an extended file mark • At the beginning-of-tape mark On a 9-track tape drive, the end-of-tape marker can be written at any location on the tape. However, this subcommand does not support overwriting single blocks of data.
fsf	Moves the tape forward the number of file marks specified by the *Count* parameter and positions it on the end-of-tape (EOT) side of the file mark.
bsf	Moves the tape backward the number of file marks specified by the *Count* parameter and positions it on the beginning-of-tape (BOT) side of the file mark. If the **bsf** subcommand moves the tape past the beginning, the tape rewinds, and the **tctl** command returns **EIO**.
fsr	Moves the tape forward the number of records specified by the *Count* parameter.
bsr	Moves the tape backwards the number of records specified by the *Count* parameter.
rewind	Rewinds the tape. The *Count* parameter is ignored.
offline or **rewoffl**	Rewinds the tape and takes the tape drive offline. This will unload the tape when appropriate. The tape must be re-inserted before the device can be used again.
erase	Erases all contents on the tape and rewinds it.
read	Reads from the specified tape device (using the specified block size) until the internal buffer is full, and then writes the data to standard output, continuing to read and write this way until an end-of-file (EOF) mark is reached.
reset	Sends a bus device reset (BDR) to the tape device. The BDR will only be sent if the device cannot be opened and is not busy.

Item	Description
retension	Moves the tape to the beginning, then to the end, and then back to the beginning of the tape. If you have excessive read errors during a restore operation, you should run the **retension** subcommand. If the tape has been exposed to environmental extremes, you should run the **retension** subcommand before writing to tape. The 8 mm tape drive will not respond to this command.
status	Prints status information about the specified tape device.
write	Opens the tape device, reads from standard input, and writes the data to the tape device.

Tip: When you specify the **read** or **write** subcommand, the **tctl** command opens the tape device and sets up the tape block size as specified by the **-b** or **-n** flag. If neither flag is specified, the **tctl** command uses a default block size of 512 bytes.

Restrictions:

- The **-b**, **-n**, **-p**, and **-v** flags apply only when using the **read** and **write** subcommands.
- The **-B** flag applies only when using the **read** subcommand.

Flags

Item	Description
-b *BlockSize*	Specifies, in bytes, the size of buffer used to read and write to the tape device, and also specifies, in the absence of the **-n** flag, the tape block size. If the block size is 0, variable-length blocks are used and the size of the tape buffer is 32,768. If the **-b** flag is not specified, the default block size and the size of the tape buffer is 512 bytes.
-B	Writes the contents of the buffer each time the tape is read. Set this flag when reading variable-length records that are not of a regular and consistent size.
-f *Device*	Specifies the tape device.
-p *BufferSize*	Specifies the size of the buffer to be used on standard input and standard output. The default buffer size is 32,768 bytes. The *BufferSize* value must be a multiple of the tape block size.
-v	Verbose. Prints the sizes of each read and write to standard error.
-n	Specifies variable-length records when reading or writing to tape with the **read** or **write** subcommand.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To rewind the rmt1 tape device, enter:

```
tctl -f /dev/rmt1 rewind
```

2. To move forward two file marks on the default tape device, enter:

   ```
   tctl  fsf 2
   ```

3. To write two end-of-file markers on the tape in /dev/rmt0.6, enter:

   ```
   tctl  -f /dev/rmt0.6  weof 2
   ```

4. To read a tape device formatted in 80-byte blocks and put the result in a file, enter:

   ```
   tctl -b 80  read > file
   ```

5. To read variable-length records from a tape device formatted in 80-byte blocks and put the result in a file, enter:

   ```
   tctl -b 80  -n read > file
   ```

6. To write variable-length records to a tape device using a buffer size of 1024 byes, enter:

   ```
   cat file | tctl -b 1024 -n -f/dev/rmt1 write
   ```

7. To write to a tape device in 512-byte blocks and use a 5120-byte buffer for standard input, enter:

   ```
   cat file | tctl -v -f /dev/rmt1 -p 5120 -b 512 write
   ```

   Note: The only valid block sizes for quarter-inch (QIC) tape drives are 0 and 512.

8. To write over one of several backups on an 8 mm tape, position the tape at the start of the backup file and issue these commands:

   ```
   tctl bsf 1

   tctl eof 1
   ```

The first command moves the tape to the beginning-of-tape side of the file mark. The second command rewrites the file mark, because writing is allowed before extended file marks. The erase head of the drive erases data before the write head reaches it, so the **write** subroutines can write over data already in the tape. However, all old data following is lost because its file markers are meaningless.

> **Note:** The **write** subroutines cannot write over a short file mark unless blank tape follows the short file mark. To write over existing data, as in the case of this example, the tape must be written with extended file marks (as specified through the SMIT interface).

Files

Item	Description
/dev/rmt*n*	Specifies the raw streaming tape interface.
/usr/bin/tctl	Contains the **tctl** command.

tee Command

Purpose

Displays the output of a program and copies it into a file.

Syntax

tee [**-a**] [**-i**] [*File ...*]

Description

The **tee** command reads standard input, then writes the output of a program to standard output and simultaneously copies it into the specified file or files.

Flags

Item	Description
-a	Adds the output to the end of *File* instead of writing over it.
-i	Ignores interrupts.

Exit Status

This command returns the following exit values:

Item	Description
0	The standard input was successfully copied to all output files.
>0	An error occurred.

> **Note:** If a write to any successfully opened *File* operand is not successful, writes to other successfully opened *File* operands and standard output will continue, but the exit value will be **>0**.

Examples

1. To view and save the output from a command at the same time:

   ```
   lint program.c | tee program.lint
   ```

   This displays the standard output of the command **lint program.c** at the workstation, and at the same time saves a copy of it in the file `program.lint`. If a file named `program.lint` already exists, it is deleted and replaced.
2. To view and save the output from a command to an existing file:

   ```
   lint program.c | tee  -a program.lint
   ```

   This displays the standard output of the **lint program.c** command at the workstation and at the same time appends a copy of it to the end of the `program.lint` file. If the `program.lint` file does not exist, it is created.

Files

Item	Description
/usr/bin/tee	Contains the **tee** command.

telinit or init Command

Purpose

Initializes and controls processes.

Syntax

{ **telinit** | **init** } { **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **a** | **b** | **c** | **h** | **Q** | **q** | **S** | **s** | **M** | **m** | **N** }

Description

The **init** command initializes and controls processes. Its primary role is to start processes based on records read from the **/etc/inittab** file. The **/etc/inittab** file usually requests that the **init** command run the **getty** command for each line on which a user can log in. The **init** command controls autonomous processes required by the system.

The process that constitutes the majority of the **init** command's process dispatching activities is **/usr/sbin/getty**. The **/usr/sbin/getty** process initiates individual terminal lines. Other processes typically dispatched by the **init** command are daemons and the shell.

The **telinit** command, which is linked to the **init** command, directs the actions of the **init** command. The **telinit** command takes a one-character argument and signals the **init** command by way of the **kill** subroutine to perform the appropriate action.

The **telinit** command sets the system at a specific run level. A run level is a software configuration that allows only a selected group of processes to exist. The system can be at one of the following run levels:

Item	Description
0-9	Tells the **init** command to place the system in one of the run levels **0-9**. When the **init** command requests a change to run levels **0-9**, it kills all processes at the current run levels and then restarts any processes associated with the new run levels.
0-1	Reserved for the future use of the operating system.
2	Contains all of the terminal processes and daemons that are run in the multiuser environment. In the multiuser environment, the **/etc/inittab** file is set up so that the **init** command creates a process for each terminal on the system. The console device driver is also set to run at all run levels so the system can be operated with only the console active.
3-9	Can be defined according to the user's preferences.
S,s,M,m	Tells the **init** command to enter the maintenance mode. When the system enters maintenance mode from another run level, only the system console is used as the terminal.

The following arguments also serve as directives to the **init** command:

Item	Description
a,b,c,h	Tells the **init** command to process only those records in the **/etc/inittab** file with **a**, **b**, **c**, or **h** in the run level field. These four arguements, **a**, **b**, **c**, and **h**, are not true run levels. They differ from run levels in that the **init** command cannot request the entire system to enter run levels **a, b, c**, or **h**. When the **init** command finds a record in the **/etc/inittab** file with a value of **a**, **b**, **c**, or **h** in the run level field, it starts the process. However, it does not kill any processes at the current run level; processes with a value of **a**, **b**, **c**, or **h** in the run level field are started in addition to the processes already running at the current system run level. Another difference between true run levels and **a**, **b**, **c**, or **h** is that processes started with **a**, **b**, **c**, or **h** are not stopped when the **init** command changes run levels. Three ways stop **a**, **b**, **c**, or **h** processes: • Type **off** in the *Action* field. • Delete the objects entirely. • Use the **init** command to enter maintenance state.
Q,q	Tells the **init** command to re-examine the **/etc/inittab** file.
N	Sends a signal that stops processes from being respawned.

During system startup, after the root file system has been mounted in the pre-initialization process, the following sequence of events occurs:

1. The **init** command is run as the last step of the startup process.

2. The **init** command attempts to read the **/etc/inittab** file.
3. If the **/etc/inittab** file exists, the **init** command attempts to locate an `initdefault` entry in the **/etc/inittab** file.
   a. If the `initdefault` entry exists, the **init** command uses the specified run level as the initial system run level.
   b. If the `initdefault` entry does not exist, the **init** command requests that the user enter a run level from the system console (**/dev/console**).
   c. If the user enters an **S**, **s**, **M** or **m** run level, the **init** command enters maintenance run level. These are the only run levels that do not require a properly formatted **/etc/inittab** file.
4. If the **/etc/inittab** file does not exist, the **init** command places the system in the maintenance run level by default.
5. The **init** command rereads the **/etc/inittab** file every 60 seconds. If the **/etc/inittab** file has changed since the last time the **init** command read it, the new commands in the **/etc/inittab** file are executed during system startup.

When you request the **init** command to change the run level, the **init** command reads the **/etc/inittab** file to identify what processes should exist at the new run level. Then, the **init** command cancels all processes that should not be running at the new level and starts any processes that should be running at the new level.

The processes run by the **init** command for each of these run levels are defined in the **/etc/inittab** file. The run level is changed by having a root user run the **telinit** command, which is linked to the **init** command. This user-run **init** command sends appropriate signals to the original **init** command initiated by the system during startup. The default run level can be changed by modifying the run level for the `initdefault` entry in the **/etc/inittab** file.

In the maintenance run level, the **/dev/console** console terminal is opened for reading and writing. The password for root is prompted. When the root password is entered successfully, the **su** command is invoked. Two ways exist to exit from the maintenance run level:

- If the shell is terminated, the **init** command requests a new run level.

  OR
- The **init** (or **telinit**) command can signal the **init** command and force it to change the run level of the system.

During a system startup attempt, apparent failure of the **init** command to prompt for a new run level (when **initdefault** is maintenance) may be due to the fact that the terminal console device (**/dev/console**) has been switched to a device other than the physical console. If this occurs and you wish to work at the physical console rather than the **/dev/console**, you can force the **init** command to switch to the physical console by pressing the DEL (delete) key at the physical console device.

When the **init** command prompts for a new run level, enter one of the digits **0** through **9** or any of the letters **S**, **s**, **M**, or **m**. If you enter **S**, **s**, **M**, or **m**, the **init** command operates in maintenance mode with the additional result that if control had previously been forced to switch to the physical console, the **/dev/console** file is switched to this device as well. The **init** command generates a message to this effect on the device to which the **/dev/console** file was previously connected.

If you enter a **0** through **9** run level, the **init** command enters the corresponding run level. The **init** command rejects any other input and re-prompts you for the correct input. If this is the first time the **init** command enters any run level other than maintenance, it searches the **/etc/inittab** file for entries with the **boot** or **bootwait** keywords. If the **init** command finds these keywords, it performs the corresponding task, provided the run level entered matches that of the entry. For example, if the **init** command finds the **boot** keyword, it boots the machine. Any special initialization of the system, such as checking and mounting file systems, takes place before any users are allowed on the system. The **init** command then scans the **/etc/inittab** file to find all entries that are processes for that level. It then resumes normal processing of the **/etc/inittab** file.

Run level **2** is defined by default to contain all of the terminal processes and daemons that are run in the multiuser environment. In the multiuser environment, the **/etc/inittab** file is set up so that the **init** command creates a process for each terminal on the system.

For terminal processes, the shell terminates either as a result of an end of file character (EOF) typed explicitly or as the result of disconnection. When the **init** command receives a signal telling it that a process has terminated, it records the fact and the reason it stopped in **/etc/utmp** file and /**var**/**adm**/**wtmp** file. The /**var**/**adm**/**wtmp** file keeps a history of the processes started.

To start each process in the **/etc/inittab** file, the **init** command waits for one of its descendant processes to stop, for a power fail signal **SIGPWR**, or until the **init** command is signaled by the **init** or **telinit** commands to change the system's run level. When one of the above three conditions occurs, the **init** command re-examines the **/etc/inittab** file. Even if new entries have been added to the **/etc/inittab** file, the **init** command still waits for one of the three conditions to occur. To provide for instantaneous response, re-examine the **/etc/inittab** file by running the **telinit -q** command.

If the **init** command finds that it is continuously running an entry in the **/etc/inittab** file (more than five times in 225 seconds), it assumes that an error in the entry command string exists. It then prints an error message to the console and logs an error in the system error log. After the message is sent, the entry does not run for 60 seconds. If the error continues to occur, the command will respawn the entry only five times every 240 seconds. The **init** command continues to assume an error occurred until the command does not respond five times in the interval, or until it receives a signal from a user. The **init** command logs an error for only the first occurrence of the error.

When the **init** command is requested to change run levels by the **telinit** command, the **init** command sends a **SIGTERM** signal to all processes that are undefined in the current run level. The **init** command waits 20 seconds before stopping these processes with the **SIGKILL** signal.

If the **init** command receives a **SIGPWR** signal and is not in maintenance mode, it scans the **/etc/inittab** file for special power fail entries. The **init** command invokes the tasks associated with these entries (if the run levels permit) before any further processing takes place. In this way, the **init** command can perform cleanup and recording functions whenever the system experiences a power failure. It is important to note that these power fail entries should not use devices that need to be initialized first.

Environments

Because the **init** command is the ultimate ancestor of every process on the system, every other process on the system inherits the **init** command's environment variables. As part of its initialization sequence, the **init** command reads the **/etc/environment** file and copies any assignments found in that file into the environment passed to all of its subprocesses. Because **init** subprocesses do not run from within a login session, they do not inherit a umask setting from **init**. These processes may set the umask to whatever value they require. A command that is executed by **init** from the **/etc/inittab** file uses **init**'s ulimit values and not the default values as given in **/etc/security/limits**. The result is that a command that is successfully executed from the command line may not execute correctly when invoked by **init**. Any command that has specific **ulimit** requirements should include specific actions to set the **ulimit** values as required.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To request the **init** command to reexamine the **/etc/inittab** file, enter:

   ```
   telinit  q
   ```
2. To request the **init** command to enter maintenance mode, enter:

```
telinit  s
```

Files

Item	Description
/etc/inittab	Specifies the **init** command control file.
/etc/utmp	Specifies the record of logged-in users.
/var/adm/wtmp	Specifies the permanent login accounting file.
/sbin/rc.boot	Specifies the pre-initialization command file.
/etc/rc	Specifies the initialization command file.
/etc/environment	Specifies system environment variables.
/dev/console	Specifies the console device driver.

telnet, tn, or tn3270 Command

Purpose

Connects the local host with a remote host, using the Telnet interface.

Syntax

{ **telnet** | **tn** | **tn3270** } [**-d**] [**-p**] [**-n** *TraceFile*] [**-e** *TerminalType*] [**-f** | **-F**] [**-k** *realm*] [**-l** user] [*Host* [*Port*]]

Description

The **telnet** command, which is also referred to as the **tn** or **tn3270** command, operates in two different modes: command mode and input mode.

System

The user is assigned a default login Sensitivity Label (SL) and Integrity Label (TL), which is SL and TL of the user's process after successful login.

If the user does not want to login using the default login SL, the user can choose to supply a different SL at the login time using the **-e** option. The SL that is supplied by the user must be dominated by the user's clearance and contained in the system accreditation range. The TL cannot be specified by the user at login time. The default login SL and TL are defined in the **/etc/security/user** file along with the username and clearance for each user. To use the **-e** option, the server side's kernel trusted network bit must be turned off.

Restriction: Any user with an ID less than or equal to 128 cannot login to remote Trusted AIX system.

Command Mode

When the **telnet** command is issued without arguments, it enters command mode, as indicated by the `telnet>`, `tn>`, or the `tn3270>` prompt. A user can also enter command mode from input mode by pressing Ctrl-] for the **telnet** command, Ctrl-T for the **tn** command, or Ctrl-C for the **tn3270** command. In command mode, subcommands can be entered to manage the remote system. Some of these subcommands return you to the remote session upon completion. For those subcommands that do not, pressing the Enter key returns you to the remote session.

> **Note:** The default escape sequence for this command is Ctrl-] for the **telnet** command, Ctrl-T for the **tn** command, or Ctrl-C for the **tn3270** command. This default can be overridden by changing the **TNESC** environment variable.

To enter **telnet** command mode while connected to a remote host, type the Telnet escape key sequence. When in command mode, the standard operating system editing conventions, such as backspace, are available.

Input Mode

When the **telnet** command is issued with arguments, it performs an **open** subcommand with those arguments and then enters input mode. The type of input mode is either character-at-a-time or line-by-line, depending on what the remote system supports. In character-at-a-time mode, most text that is typed is immediately sent to the remote host for processing. In line-by-line mode, all text is echoed locally and completed lines are sent to the remote host.

In either input mode, if the **toggle localchars** subcommand has a value of True, the user's QUIT, INTR, and FLUSH characters are trapped locally and sent as Telnet Protocol sequences to the remote host. The **toggle autoflush** and **toggle autosynch** subcommands cause this action to flush subsequent output to the terminal until the remote host acknowledges the Telnet sequence and to flush previous terminal input (in the case of QUIT and INTR characters).

Arabic/Hebrew Support

The **telnet**, **tn**, and **tn3270** command supports the Arabic and Hebrew texts, allowing the user to type Arabic or Hebrew characters while in an emulation session. The **Ar_AA** locale displays the Arabic characters in their correct shapes. The following functions support the bidirectional Arabic and Hebrew texts:

Language Selection

This function allows you to toggle the language layer. Activate the Arabic/Hebrew language selection with the following key combinations:

Item	Description
Alt+N	From an AIX terminal
Esc+N	From an ASCII terminal
Alt+N or **Esc+N**	From a Latin AIX terminal

Activate the Latin language layer with the following key combinations:

Item	Description
Alt+L	From an Arabic or Hebrew AIX terminal
Esc+L	From an ASCII terminal
Alt+L or **Esc+L**	From an AIX terminal

Screen Reverse

This function reverses the screen image and invokes the default language of the new screen orientation. Thus, if the screen is reversed to right-to-left, the language is changed to Arabic/Hebrew. If the screen is reversed to left-to-right, the language is changed to Latin.

If symmetric character swapping is enabled, reversing the screen causes bidirectional characters to be replaced by their counterparts. For example, if numeric character swapping is enabled, reversing the screen causes Hindi numerals to be replaced by their Arabic counterparts and the Arabic numerals to be replaced by their Hindi counterparts.

Activate screen reverse with the following key combinations:

Item	Description
Alt+S	From an Arabic or Hebrew AIX terminal
Esc+S	From an ASCII terminal

Item	Description
Alt+S or **Esc+S**	From a Latin AIX terminal

Push/End Push

The Push function allows you to edit text whose direction is opposite the screen orientation. When you activate this function, the cursor orientation is reversed, the language layer is changed accordingly, and a Push segment is created.

The Push function has two secondary modes:

Item	Description
Boundary Mode	This mode is activated upon entering the Push mode. In this mode, the cursor remains in its position while you type additional characters. The text is pushed in the opposite direction of the screen orientation.
Edit Mode	This mode is activated when the cursor is moved from its boundary position into the Push segment area. In this mode, you can edit the text within the Push segment, while typing in the field's natural direction.

Activate this function with the following key combinations:

Item	Description
Alt+P	From an Arabic or Hebrew AIX terminal
Esc+P	From an ASCII terminal
Alt+P or **Esc+P**	From a Latin AIX terminal

The End Push function terminates the Push function. The cursor jumps to the end of the Push segment and its direction changes to the original direction. You can activate End Push by pressing any field exit keys such as cursor up, cursor down, or any attention identifier (AID) key such as the Enter key. You can also activate this function with the following key combinations:

Item	Description
Alt+E	From an Arabic or Hebrew AIX terminal
Esc+E	From an ASCII terminal
Alt+E or **Esc+E**	From a Latin AIX terminal

Field Reverse

This function toggles the field orientation to either the opposite of or the same as the screen orientation. This function does not invert the text in the field. The cursor orientation is set to the new field orientation and the language layer is selected accordingly.

For example, if the cursor is in the first logical position of a field or line when you activate the field reverse function, the cursor skips to the opposite side of that field or line. This position is now the first logical position. If the cursor is not in the first position of the field or line when you activate field reverse function, the cursor remains in its position and allows natural and correct editing of the existing text. Activate this function with the following key combinations:

Item	Description
Alt+R	From an Arabic or Hebrew AIX terminal
Esc+R	From an ASCII terminal
Alt+R or **Esc+R**	From a Latin AIX terminal

Autopush

This function assists you in typing mixed left-to-right and right-to-left text. When enabled, reversed segments are automatically initiated and terminated according to the typed characters or the selected language layer. Thus, this mode automatically invokes the Push mode and relieves you of invoking the Push function.

When you type a digit or Latin character in a right-to-left field, the Autopush function automatically initiates the Push function without changing the language. If you type additional digits or Latin character, the Push function continues; otherwise, the Push function automatically terminates. Thus, you can type Arabic/Hebrew text with embedded digits or Latin characters without invoking the Push/End Push functions.

When you type an Arabic/Hebrew character in a left-to-right field, the Autopush function automatically initiates the Push function without a language change. If you then type a digit or Latin character, the Autopush function automatically terminates. Thus, you can type Latin text with embedded Arabic/Hebrew text using the Language Selection function rather than the Push/End Push functions.

Activate this function with the following key combinations:

Item	Description
Alt+A	From an Arabic or Hebrew AIX terminal
Esc+A	From an ASCII terminal
Alt+A or **Esc+A**	From a Latin AIX terminal

Field Shape

This function shapes the Arabic characters in the current field or line. Activate this function with the following key combinations:

Item	Description
Alt+H	From an Arabic AIX terminal
Esc+H	From an ASCII terminal
Alt+H or **Esc+H**	From a Latin AIX terminal

Field Deshape

This function deshapes Arabic text in the current field or line. Activate this function with the following key combinations:

Item	Description
Alt+B	From an Arabic AIX terminal
Esc+B	From an ASCII terminal
Alt+B or **Esc+B**	From a Latin AIX terminal

Contextual Shape Determination

This function determines the shape of an Arabic character based on the surrounding text. Use the Contextual Shape Determination function only when typing or editing right-to-left text. This function is terminated when any of the specific shape selection keys is pressed. This is the default function. Activate this function with the following key combinations:

Item	Description
Alt+C	From an Arabic AIX terminal
Esc+C	From an ASCII terminal
Alt+C or **Esc+C**	From a Latin AIX terminal

Initial Shape Determination

This function shapes Arabic characters in their initial shapes. Activate this function with the following key combinations:

Item	Description
Alt+I	From an Arabic AIX terminal
Esc+I	From an ASCII terminal
Alt+I or **Esc+I**	From a Latin AIX terminal

Middle Shape Determination

This function shapes Arabic characters in their middle shapes. Activate this function with the following key combinations:

Item	Description
Alt+M	From an Arabic AIX terminal
Esc+M	From an ASCII terminal
Alt+M or **Esc+M**	From a Latin AIX terminal

Isolated Shape Determination

This function shapes Arabic characters in their isolated shapes. Activate this function with the following key combinations:

Item	Description
Alt+O	From an Arabic AIX terminal
Esc+O	From an ASCII terminal
Alt+O or **Esc+O**	From a Latin AIX terminal

Final Shape Determination

This function shapes Arabic characters in their final shapes. Activate this function with the following key combinations:

Item	Description
Alt+Y	From an Arabic AIX terminal
Esc+Y	From an ASCII terminal
Alt+Y or **Esc+Y**	From a AIX terminal

Miscellaneous Functions

To activate numeric swapping, type the following line at the command line:

```
export ARB_NUM_SWAP=1
```

To activate symmetric swapping, that is, to swap bidirectional characters such as braces, brackets, and so on, type the following line at the command line:

```
export ARB_SYM_SWAP=1
```

To specify the code page that the host uses, type the following line at the command line:

```
export RM_HOST_LANG=IBM-420
```

Terminal Type Negotiation

The **telnet** command negotiates the terminal type, using the Telnet protocol, and it sets the **TERM** environment variable according to what has been negotiated.

To override the terminal negotiation from the console, use the **EMULATE** environment variable or the **-e** flag; or invoke the tn3270 command if you require 3270 emulation. To determine whether terminal-type negotiation is performed, the following list describes the order of the **telnet** command processing:

1. The **-e** command-line flag. (No negotiation.)
2. The **EMULATE** environment variable. (No negotiation.)
3. The **tn3270** command. (No negotiation.)
4. If steps 1, 2, and 3 are not present, terminal-type negotiation occurs automatically.

If the client and the server negotiate to use a 3270 data stream, the keyboard mapping is determined by the following precedence:

Item	Description
$HOME/.3270keys	Specifies the user's 3270 keyboard mapping when the **tn** or **telnet** command is invoked. If you are using a color display, you can also change this file to customize the colors for 3270 displays.
/etc/map3270	Specifies the user's 3270 keyboard mapping when the **tn3270** command is invoked. The **/etc/map3270** file defines keyboard mapping and colors for the **tn3270** command.
/etc/3270.keys	Specifies the base 3270 keyboard mapping for use with limited function terminals.

Secure Attention Key (SAK) Option

In addition to terminal negotiation, the **telnet** command allows negotiation for the Secure Attention Key (SAK) option. This option, when supported, provides the local user with a secure communication path to the remote host for tasks such as changing user IDs or passwords. If the remote host supports the **SAK** function, a trusted shell is opened on the remote host when the **telnet send sak** subcommand is issued. The **SAK** function can also be assigned to a single key available in **telnet** input mode, using the **set sak** subcommand.

End-of-Line Convention

The Telnet protocol defines the carriage-return line-feed (CR-LF) sequence to mean "end-of-line." For terminal input, this corresponds to a command-completion or end-of-line key being pressed on a user terminal. On an ASCII terminal, this is the CR key, but it may also be labeled "Return" or "Enter."

When a Telnet server receives the Telnet end-of-line sequence, CR-LF, as input from a remote terminal, the effect is the same as if the user had pressed the end-of-line key on a local terminal.

On ASCII servers, receiving the Telnet sequence CR-LF causes the same effect as a local user pressing the CR key on a local terminal. CR-LF and CR-NUL have the same effect on an ASCII server when received as input over a Telnet connection.

> **Note:** A Telnet user must be able to send CR-LF, CR-NULL, or LF. An ASCII user must be able to send CR-LF or CR-NULL.

A Telnet user on an ASCII host should have a user-controllable mode to send either CR-LF or CR-NULL when the user presses the end-of-line key. The CR-LF should be the default. The Telnet end-of-line sequence, CR-LF, must be used to send Telnet data that is not terminal-to-computer. This occurs, for example, when a Telnet server sends output or when the Telnet protocol incorporates another application protocol.

The **telnet** command "execs" (using the **exec** command) the **/usr/sbin/login** command to validate a user. This 1) allows all user and device attributes to take effect on telnet connections and 2) causes telnet connections to count against the maximum number of login sessions allowable at a time (determined by the **maxlogins** attribute). Attributes are defined in the **/etc/security/user** and **/etc/security/login.cfg** files.

Restrictions

- Earlier versions of the **telnet** command are not compatible with AIX Version 4 and later of the **telnet** command in sending escapes that emulate a high function terminal (HFT). The present version of the **telnet** command sends only one escape when the escape key is hit, while prior versions send two escape characters.
- The **telnet** command must allow transmission of 8-bit characters that are not in binary mode to implement ISO 8859 Latin code page. This is necessary for internationalization of the TCP/IP commands.
- In order to support new character sets, the following was added to the `hft-m`, `ibm5081`, `hft`, `hft-nam`, `hft-c`, `aixterm-m`, and `aixterm` entries in the **terminfo** file:

```
box1=\154\161\153\170\152\155\167\165\166\164\156,     batt1=f1,
box2=\154\161\153\170\152\155\167\165\166\164\156,     batt2=f1md,
font0=\E(B,     font1=\E(0,
```

- The **rlogind** and **telnetd** daemons use POSIX line discipline to change the line discipline on the local TTY. If POSIX line discipline is not used on the local TTY, echoing other line disciplines may result in improper behavior. AIX TCP/IP must have POSIX line discipline to function properly.
- The mouse cannot be used as an input device with the **telnet** command.
- The **telnet** command does not support the APL data stream.

Environment Variables

The following environment variables can be used with the **telnet** command:

Item	Description
EMULATE	Overrides terminal-type negotiation in the same way as the **-e** flag. If the value of the **EMULATE** environment variable is defined as **vt100** or **3270**, the **telnet** command emulates a DEC VT100 terminal or 3270 terminal, respectively. If the **EMULATE** variable is not defined or has a value of **none**, the **telnet** command operates normally. If the **EMULATE** variable is set to **vt100** or **3270**, the **TERM** environment variable in the remote login connection should be set to the same value. You can check this by using the **env** command after the connection is open.
TNESC	Specifies an alternate TELNET escape character, other than the default, Ctrl-] for the **telnet** command, Ctrl-T for the **tn** command, or Ctrl-C for the **tn3270** command. To change the **telnet** escape sequence, set **TNESC** to the octal value of the character you want to use. Then export **TNESC**. For example, set **TNESC** to 35 to change the TELNET escape sequence to Ctrl-].
MAP3270	Specifies an alternate file that contains the user's 3270 keyboard mapping. The **MAP3270** variable must contain the full path name to the alternate file. Create the alternate file using the same format as the default **/etc/map3270** file.
RM_HOST_LANG	Specifies the EBCDIC code page being used on the remote 3270 host. Set the **RM_HOST_LANG** environment variable to the correct code page before you telnet (using the **telnet** command) to a non-English-speaking 3270 host. The default is English. Refer to the **Converters Overview for Programming** in *Globalization Guide and Reference* for possible code pages to use. Format the **RM_HOST_LANG** environment variable by specifying the desired code page. **Restriction:** The **tn3270** command does not support DBCS, because terminal types for DBCS are not supported. The **telnet** command converts characters by using the **iconv** command. Users can change the default conversion tables by using the **genxlt** command.

Flags

Item	Description
-d	Turns debugging mode on.
-e *TerminalType*	Overrides terminal-type negotiation. Possible values are **vt100**, **3270**, or **none**.
-n *TraceFile*	Records network trace information in the file specified by the *TraceFile* variable.
-p	Preserves current TTY attributes.
-f	Causes the credentials to be forwarded. This flag will be ignored if Kerberos 5 is not the current authentication method. Authentication will fail if the current DCE credentials are not marked forwardable.
-F	Causes the credentials to be forwarded. In addition, the credentials on the remote system will be marked forwardable (allowing them to be passed to another remote system). This flag will be ignored if Kerberos 5 is not the current authentication method. Authentication will fail if the current DCE credentials are not marked forwardable.
-k *realm*	Allows the user to specify the realm of the remote station if it is different from the local systems realm. For these purposes, a *realm* is synonymous with a DCE cell. This flag will be ignored if Kerberos 5 is not the current authentication method.
-l *user*	Specifies the remote user the telnet wants to login as. This option is ignored if Kerberos 5 is not the current authentication method.

Subcommands

Before entering each subcommand, press the escape key sequence. The escape sequence tells the program that non-text information follows. Otherwise, the program interprets subcommands as text.

For each of the subcommands in the following list, you only need to type enough letters to uniquely identify the subcommand. (For example, **q** is sufficient for the **quit** subcommand.) This is also true for the arguments to the **display**, **emulate**, **mode**, **set**, and **toggle** subcommands.

The **telnet** subcommands are:

Item	Description
? [*Subcommand*]	Requests help on **telnet** subcommands. Without arguments, the **?** subcommand prints a help summary. If a *Subcommand* variable is specified, help information is displayed for that subcommand.
close	Closes the TELNET connection and returns to **telnet** command mode when the **open** subcommand is used to establish the connection. When the **telnet** command is invoked and a host is specified, the **close** subcommand closes the TELNET connection and exits the **telnet** program (identical to the **quit** subcommand).
display [*Argument*]	Displays all of the **set** and **toggle** values if no *Argument* variable is specified; otherwise, lists only those values that match the *Argument* variable.

Item	Description
emulate *TerminalType*	Overrides terminal-type negotiation with the specified terminal type. Possible choices are: **?** Prints help information. **3270** Emulates a 3270 terminal. **none** Specifies no emulation.
vt100	Emulates a DEC VT100 terminal.

All output received from the remote host is processed by the specified emulator. The initial terminal type to emulate can be specified through the **EMULATE** environment variable or the **-e** flag to the **telnet** command.

Restriction: Only standard ASCII characters are allowed in emulation mode.

Item	Description
mode *Type*	Specifies the current input mode. When the *Type* variable has a value of **line**, the mode is line-by-line. When the *Type* variable has a value of **character**, the mode is character-at-a-time. Permission is requested from the remote host before entering the requested mode, and if the remote host supports it, the new mode is entered.
open *Host* [*Port*]	Opens a connection to the specified host. The *Host* specification can be either a host name or an Internet address in dotted-decimal form. If no *Port* variable is specified, the **telnet** subcommand attempts to contact a TELNET server at the default port.
quit	Closes a TELNET connection and exits the **telnet** program. A Ctrl-D in command mode also closes the connection and exits.

Item	Description
send *Arguments*	Sends one or more arguments (special character sequences) to the remote host. Multiple arguments are separated by spaces. The following arguments can be used: **?** Prints help information for the **send** subcommand. **ao** Sends the TELNET AO (Abort Output) sequence, which causes the remote host to flush all output from the remote system to the local terminal. **ayt** Sends the TELNET AYT (Are You There) sequence, to which the remote system can respond. **brk** Sends the TELNET BRK (Break) sequence, which causes the remote system to perform a kill operation. **ec** Sends the TELNET EC (Erase Character) sequence, which causes the remote host to erase the last character entered. **el** Sends the TELNET EL (Erase Line) sequence, which causes the remote system to erase the line currently being entered. **escape** Sends the current **telnet** escape character. The default escape sequence is Ctrl-] for the **telnet** command, Ctrl-T for the **tn** command, or Ctrl-C for the **tn3270** command. **ga** Sends the TELNET GA (Go Ahead) sequence, which provides the remote system with a mechanism to signal the local system to return control to the user. **ip** Sends the TELNET IP (Interrupt Process) sequence, which causes the remote system to cancel the currently running process. **nop** Sends the TELNET NOP (No Operation) sequence. **sak** Sends the TELNET SAK (Secure Attention Key) sequence, which causes the remote system to invoke the trusted shell. If the SAK is not supported, then an error message is displayed that reads: `Remote side does not support SAK.` **synch** Sends the TELNET SYNC sequence, which causes the remote system to discard all previously typed input that has not yet been read. This sequence is sent as TCP/IP urgent data.

Item	Description
set *VariableValue*	Sets the specified TELNET variable to the specified value. The special value **off** turns off the function associated with the variable entered. The **display** subcommand can be used to query the current setting of each variable. The variables that can be specified are: **echo** Toggles between local echo of entered characters and suppressing local echo. Local echo is used for normal processing, while suppressing the echo is convenient for entering text that should not be displayed on the screen, such as passwords. This variable can only be used in line-by-line mode. **eof** Defines the character for the **telnet** command. When the **telnet** command is in line-by-line mode, entering the eof character as the first character on a line sends the character to the remote host. The initial value for the eof character is the local terminal End-Of-File character. **erase** Defines the erase character for the **telnet** command. When the **telnet** command is in character-at-a-time mode and **localchars** has a value of **true**, typing the erase character sends the TELNET EC sequence to the remote host. The initial value for the erase character is the local terminal ERASE character. **escape** Specifies the **telnet escape** character, which puts the **telnet** command into command mode when connected to a remote host. This character can also be specified in octal in the **TNESC** environment variable. **flushoutput** Defines the flush character for the **telnet** command. When **localchars** has a value of **true**, typing the flushoutput character sends the TELNET AO sequence to the remote host. The initial value for the flush character is Ctrl-O. If the remote host is running AIX, the **flushoutput** variable, unlike the other special characters defined by the **set** subcommand, only works in **localchars** mode since it has no **termio** equivalent. **interrupt** Defines the interrupt character for the **telnet** command. When **localchars** has a value of **true**, typing the interrupt character sends the TELNET IP sequence to the remote host. The initial value for the interrupt character is the local terminal interrupt (INTR) character. **kill** Defines the kill character for the **telnet** command. When the **telnet** command is in character-at-a-time mode and **localchars** has a value of **true**, typing the kill character sends the TELNET EL sequence to the remote host. The initial value for the kill character is the local terminal KILL character. **quit** Defines the quit character for the **telnet** command. When **localchars** has a value of **true**, typing the quit character sends the TELNET BRK sequence to the remote host. The initial value for the quit character is the local terminal QUIT character. **sak** Defines the Secure Attention Key (SAK) for the **telnet** command. When the sak character is entered, the remote system is asked to create a trusted shell. If the remote host does not support the SAK, this sequence has no effect.
status	Shows the status of the **telnet** command, including the current mode and the currently connected remote host.

Item	Description
toggle *Arguments*	Toggles one or more arguments that control how the **telnet** command responds to events. Possible values are **true** and **false**. Multiple arguments are separated by spaces. The **display** subcommand can be used to query the current setting of each argument. The following arguments can be used: **?** Displays valid arguments to **toggle**. **autoflush** If **autoflush** and **localchars** both have a value of **true** and the AO, INTR, and QUIT characters are recognized and transformed into TELNET sequences, the **telnet** command does not display any data on the user's terminal until the remote system acknowledges (with a TELNET **timing mark** option) that it has processed those TELNET sequences. The initial value of **autoflush** is **true** if the terminal has not done an **stty noflsh**, and **false** if it has. **autosynch** If **autosynch** and **localchars** are both **true**, then typing the INTR or QUIT character sends that character's TELNET sequence, followed by the TELNET SYNC sequence. This procedure causes the remote host to discard all previously typed input until both of the TELNET sequences have been read and acted upon. The initial value of this toggle is **false**. **crmod** Toggles carriage return mode. When set to **true**, most carriage return characters received from the remote host are mapped into a carriage return followed by a line feed. This mode does not affect the characters typed by the user, only those received from the remote host. This mode is useful when the remote host sends only a carriage return and not a line feed. The initial value of this toggle is **false**. **debug** Toggles debugging at the socket level. The initial value of this toggle is **false**. **localchars** Determines the handling of TELNET special characters. When this value is **true**, the ERASE, FLUSH, INTERRUPT, KILL, and QUIT characters are recognized locally and transformed into the appropriate TELNET control sequences (EC, AO, IP, BRK, and EL, respectively). When this value is **false**, these special characters are sent to the remote host as literal characters. The initial value of **localchars** is **true** in line-by-line mode and **false** in character-at-a-time mode. **netdata** Toggles the display of all network data (in hexadecimal format). The data is written to standard output unless a *TraceFile* value is specified with the **-n** flag on the **telnet** command line. The initial value of this toggle is **false**. **options** Toggles the display of internal TELNET Protocol processing options, such as terminal negotiation and local or remote echo of characters. The initial value of this toggle is **false**, indicating that the current options should not be displayed. **lineterm** Toggles the default end-of-line terminator to CR-LF (ASCII carriage-return line-feed). A telnet client running on an ASCII host should have the user configurable option to send either the CR-NUL or CR-LF terminator when the user presses the end-of-line key. The initial value of this toggle is **false**.

Item	Description
z	Suspends the TELNET process. To return to the TELNET process, use the **fg** built-in command of the **csh** or **ksh** command. **Note:** The **z** subcommand has the same effect as a Ctrl-Z key sequence for any other process. It suspends Telnet execution and returns you to your original login shell.

Authentication

If the system is configured for Kerberos 5 authentication, the telnet client will attempt authentication negotiation. The authentication negotiation used by telnet and the definitions of the options and suboptions for this are defined in rfc 1416.

If the client and server agree on an authentication type, they will exchange authentication information including the account the client wants to access. This will be the local user unless the **-l** flag is set.

If they cannot agree on the authentication information or if it fails, the telnet connection will continue with the standard connection (provided Standard AIX is configured).

The remote host allows access only if all of the following conditions are satisfied:

- The local user has current DCE credentials.
- The remote system accepts the DCE credentials as sufficient for access to the remote account. See the **kvalid_user** function for additional information.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

In the following examples, if you enter the **tn** command instead of the **telnet** command, the command mode prompt is displayed as `tn>`.

1. To log in to the remote host `host1` and perform terminal negotiation, enter:

   ```
   telnet host1
   ```

2. To log in to `host1` as a **vt100** terminal (no terminal type negotiation), choose one of the following methods:

   a. Use the following commands to set the **EMULATE** environment variable for this login session, then enter the **telnet** command:

   ```
   EMULATE=vt100; export EMULATE
   telnet host1
   ```

   b. Use the **-e** flag to set the terminal type for this **telnet** session only:

   ```
   telnet -e vt100 host1
   ```

3. To log in to a remote host and then check the status of the **telnet** program, enter:

   ```
   telnet host3
   ```

   When the login prompt appears, enter your login ID and password. Press the Ctrl-T key sequence to receive the `telnet>` prompt. Enter the following at the `telnet>` prompt:

   ```
   status
   ```

Information similar to the following is displayed on your screen:

```
Connected to host3.
Operating in character-at-a-time mode.
Escape character is '^]'.
```

Upon completion of the **status** subcommand, press the Enter key to return to the remote prompt.

Once you have completed your login, you can issue commands. To log out of the system and close the connection, press the Ctrl-D key sequence, or exit.

4. To log in to a remote host using the **tn3270** command, enter:

```
tn3270 hostname
```

The host login screen should be displayed. You can now enter your login ID and password. Once you have completed your login, you can issue commands. To log out of the system and close the connection, press Ctrl-D or exit.

5. To connect to the **icehouse.austin.ibm.com** remote host with the **telnet** command with a user name `david` of specific SLs `sec a b`, enter the following commands:

   a. In the command line, enter `telnet icehouse.aoot.austin.ibm.com` to connect to the **icehouse.austin.ibm.com**

   b. In the `login` field, enter `david -e "sec a b"`

   c. In the `passwords` field, enter david's passwords.

   To disconnect from the remote server, use the **Ctrl-T** key sequence.

Files

Item	Description
/etc/3270.keys	Defines base 3270-keyboard mapping for use with limited function terminals.

telnetd Daemon

Purpose

Provides the server function for the TELNET protocol.

Syntax

/usr/sbin/telnetd [**-a**] [**-c**] [**-n**] [**-s**]

Description

> **Note:** The **telnetd** daemon is normally started by the **inetd** daemon. It can also be controlled from the command line, using SRC commands.

The **/usr/sbin/telnetd** daemon is a server that supports the Defense Advanced Research Product Agency (DARPA) standard Telnet Protocol (TELNET). Changes to the **telnetd** daemon should be made using the System Management Interface Tool (SMIT).

Changes to the **telnetd** daemon can be made using the System Management Interface Tool (SMIT) or System Resource Controller (SRC), by editing the **/etc/inetd.conf** or /**etc**/**services** file. Typing `telnetd` at the command line is not recommended. The **telnetd** daemon is started by default when it is uncommented in the **/etc/inetd.conf** file. By default, the **-a** flag is also turned on.

The **inetd** daemon get its information from the **/etc/inetd.conf** file and the **/etc/services** file.

After changing the **/etc/inetd.conf** or **/etc/services** file, run the **refresh -s inetd** or **kill -1** *InetdPID* command to inform the **inetd** daemon of the changes to its configuration file.

When a **telnet** session is started, the **telnetd** daemon sends TELNET options to the client (remote) host to indicate an ability to perform options.

Terminal Negotiation

The **telnetd** daemon requests the terminal type from the client host. On receipt, the **telnetd** daemon checks whether the indicated type is supported on the local system. If not, the daemon requests a terminal type again.

This terminal type negotiation continues until the remote client sends an acceptable terminal type or until the client sends the same type twice in a row, indicating that it has no other types available. When necessary, the **telnetd** daemon refers to the **/etc/telnet.conf** file to translate a client's terminal-type strings into **terminfo** file entries.

> **Note:** Because the **telnetd** daemon allows the sending and receiving of 8-bit ASCII, NLS is supported.

If the remote client sends the TELNET **SAK** command, the **telnetd** daemon passes the local SAK characters through the PTY to invoke the trusted shell.

The **telnetd** daemon supports the following telnet options:

- Binary
- Echo/no echo
- Support SAK
- Suppress go ahead
- Timing mark
- Negotiate About Window Size (NAWS)
- Authentication

The **telnetd** daemon also recognizes the following options for the remote client:

- Binary
- Suppress go ahead
- Echo/no echo
- Terminal type

The **telnetd** daemon should be controlled using the System Management Interface Tool (SMIT) or by changing the **/etc/inetd.conf** file. Typing `telnetd` at the command line is not recommended.

Authentication Negotiation

If the system has Kerberos 5 authentication configured, **telnetd** will accept authentication option negotiation. If both agree on Kerberos 5 authentication, the client will pass over the DCE principal and **telnetd** will use the **kvalid_user** routine to determine if the DCE principal should have access to the account. If it passes, no password will be requested.

Manipulating the telnetd Daemon with the System Resource Controller

The **telnetd** daemon is a subserver of the **inetd** daemon, which is a subsystem of the System Resource Controller (**SRC**). The **telnetd** daemon is a member of the **tcpip** SRC subsystem group. This daemon is enabled by default in the **/etc/inetd.conf** file and can be manipulated by the following SRC commands:

Item	Description
startsrc	Starts a subsystem, group of subsystems, or a subserver.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
lssrc	Gets the status or a subsystem, group or subsystems, or a subserver.

Flags

Item	Description
-a	Causes the PTY and socket to be linked directly in the kernel so that the data handling remains in the kernel to improve the performance.
-c	Suppresses the reverse host name lookup.
-n	Disables transport-level keep-alive messages. Messages are enabled by default.
-s	Turns on socket-level debugging.

Note: Unrecognized flags will be ignored by the daemon and logged to the `syslog` if syslog is enabled.

Security

The **telnetd** daemon is a PAM-enabled application with a service name of *telnet*. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the **usw** stanza of **/etc/security/login.cfg**, to PAM_AUTH as the root user.

The authentication mechanisms used when PAM is enabled depend on the configuration for the **telnet** service in **/etc/pam.conf**. The **telnetd** daemon requires **/etc/pam.conf** entries for the **auth**, **account**, **password**, and **session** module types. Listed below is a recommended configuration in **/etc/pam.conf** for the **telnet** service:

```
#
# AIX telnet configuration
#
telnet auth      required     /usr/lib/security/pam_aix

telnet account   required     /usr/lib/security/pam_aix

telnet password  required     /usr/lib/security/pam_aix

telnet session   required     /usr/lib/security/pam_aix
```

Examples

Note: The arguments for the **telnetd** daemon can be specified by using SMIT or by editing the **/etc/inetd.conf** file.

1. To start the **telnetd** daemon, type the following:

   ```
   startsrc  -t telnet
   ```

   This command starts the **telnetd** subserver.
2. To stop the **telnetd** daemon normally, type the following:

   ```
   stopsrc  -t telnet
   ```

   This command allows all pending connections to start and existing connections to complete but prevents new connections from starting.
3. To force stop the **telnetd** daemon and all **telnetd** connections, type the following:

   ```
   stopsrc -f -t telnet
   ```

   This command terminates all pending connections and existing connections immediately.
4. To display a short status report about the **telnetd** daemon, type the following:

   ```
   lssrc -t telnet
   ```

   This command returns the daemon's name, process ID, and state (active or inactive).

File

Item	Description
terminfo	Describes terminal by capability.

termdef Command

Purpose

Queries terminal characteristics.

Syntax

termdef [**-c** | **-l** | **-t**]

Description

The **termdef** command identifies the current display type, the active lines setting, or the current columns setting. This simplifies resetting the lines and columns when you switch fonts as well as resetting the **TERM** environment variable when you switch displays. The **terminfo** database defines the default number of lines and columns for each display, but the lines and columns can change depending upon which font is currently active. Also, the **TERM** environment variable does not automatically reflect the currently active display.

The flags for the **termdef** command are mutually exclusive. If you use more than one flag with the command, the **termdef** command recognizes and returns the current value for the first flag only. Any other flags are ignored. For example, the **termdef -lc** command returns only the active lines setting for the current display.

Flags

Item	Description
-c	Returns the current column value.
-l	Returns the current line value.
-t	Returns the name of the current display (the default action).

Example

To determine the current value of the **TERM** environment variable, enter:

```
termdef -c
```

File

Item	Description
/usr/bin/termdef	Contains the **termdef** command.

test Command

Purpose

Evaluates conditional expressions.

Syntax

test *Expression*

OR

[*Expression* **]**

Description

The **test** command evaluates the *Expression* parameter, and if the expression value is True, returns a zero (True) exit value. Otherwise, the **test** command returns a nonzero (False) exit value. The **test** command also returns a nonzero exit value if there are no parameters.

Requirements:

- In the second form of the command, the **[ ]** (brackets) must be surrounded by blank spaces.
- You must test explicitly for file names in the C shell. File-name substitution (globbing) causes the shell script to exit.

Functions and operators are treated as separate parameters by the **test** command. The *Expression* parameter refers to a statement that is checked for a true or false condition. The following functions are used to construct this parameter:

Item	Description
-b *FileName*	Returns a True exit value if the specified *FileName* exists and is a block special file.
-c *FileName*	Returns a True exit value if the specified *FileName* exists and is a character special file.
-d *FileName*	Returns a True exit value if the specified *FileName* exists and is a directory.
-e *FileName*	Returns a True exit value if the specified *FileName* exists.
-f *FileName*	Returns a True exit value if the specified *FileName* exists and is a regular file.
-g *FileName*	Returns a True exit value if the specified *FileName* exists and its Set Group ID bit is set.
-h *FileName*	Returns a True exit value if the specified *FileName* exists and is a symbolic link.
-k *FileName*	Returns a True exit value if the specified *FileName* exists and its sticky bit is set.
-L *FileName*	Returns a True exit value if the specified *FileName* exists and is a symbolic link.
-n *String1*	Returns a True exit value if the length of the *String1* variable is nonzero.
-p *FileName*	Returns a True exit value if the specified *FileName* exists and is a named pipe (FIFO).
-r *FileName*	Returns a True exit value if the specified *FileName* exists and is readable by the current process.
-s *FileName*	Returns a True exit value if the specified *FileName* exists and has a size greater than 0.
-t *FileDescriptor*	Returns a True exit value if the file with a file descriptor number of *FileDescriptor* is open and associated with a terminal.

Item	Description
-u *FileName*	Returns a True exit value if the specified *FileName* exists and its Set User ID bit is set.
-w *FileName*	Returns a True exit value if the specified *FileName* exists and the write flag is on. However, the *FileName*will not be writable on a read-only file system even if **test** indicates true.
-x *FileName*	Returns a True exit value if the specified *FileName* exists and the execute flag is on. If the specified file exists and is a directory, the True exit value indicates that the current process has permission to search in the directory.
-z *String1*	Returns a True exit value if the length of the *String1* variable is 0 (zero).
*String1***=** *String2*	Returns a True exit value if the *String1* and *String2* variables are identical.
*String1***!=***String2*	Returns a True exit value if the *String1* and *String2* variables are not identical.
String1	Returns a True exit value if the *String1* variable is not a null string.
Integer1 **-eq** *Integer2*	Returns a True exit value if the *Integer1* and *Integer2* variables are algebraically equal. Any of the comparisons **-ne**, **-gt**, **-ge**, **-lt**, and **-le** can be used in place of **-eq**.
file1 **-nt** *file2*	True if *file1* is newer than *file2*.
file1 **-ot** *file2*	True if *file1* is older than *file2*.
file1 **-ef** *file2*	True if *file1* is another name for *file2*.

These functions can be combined with the following operators:

Item	Description
!	Unary negation operator
-a	Binary AND operator
-o	Binary OR operator (that is, the **-a** operator has higher precedence than the **-o** operator)
\(*Expression***\)**	Parentheses for grouping

Exit Status

This command returns the following exit values:

Item	Description
0	The *Expression* parameter is true.
1	The *Expression* parameter is false or missing.
>1	An error occurred.

Examples

1. To test whether a file exists and is not empty, enter the following command:

```
if test ! -s "$1"
then
```

```
   echo $1 does not exist or is empty.
fi
```

If the file specified by the first positional parameter to the shell procedure, $1, does not exist, the **test** command displays an error message. If $1 exists and has a size greater than 0, the **test** command displays nothing.

Note: There must be a space between the **-s** function and the file name.

The quotation marks around $1 ensure that the test works properly even if the value of $1 is a null string. If the quotation marks are omitted and $1 is the empty string, the **test** command displays the error message `test: argument expected.`

2. To do a complex comparison, type:

```
if [ $# -lt 2 -o ! -e "$1" ]
then
   exit
fi
```

If the shell procedure is given fewer than two positional parameters or the file specified by $1 does not exist, then the shell procedure exits. The special shell variable $# represents the number of positional parameters entered on the command line that starts this shell procedure.

The **Shells** in *Operating system and device management* describes shells in general, defines terms that are helpful in understanding shells, and describes the more useful shell functions.

File

Item	Description
/usr/bin/test	Contains the **test** command.

tetoldif Command

Purpose

Prints certain Trusted Signature Database (TSD) and TE Policies that are defined locally to **stdout** in an ldif format.

Syntax

tetoldif -d < *baseDN* > [**-s** [*filename*]] [**-p** [*filename*]]

Description

The **tetoldif** command reads data from a locally defined TSD and TE policies database files and prints the result to **stdout** in ldif format. If the results are redirected to a file, they can be added to a LDAP server with the **ldapadd** command with the **-b** flag or the **ldif2db** command.

The **tetoldif** command reads the **/etc/security/ldap/sectoldif.cfg** file to determine what to name the trusted signature database and the TE policies database sub-trees where the data is exported to. The **tetoldif** command only exports data to the TSDDAT types and TEPOLICIES types defined in the **/etc/security/ldap/sectoldif.cfg** file. The names specified in the **/etc/security/ldap/sectoldif.cfg** file will be used to create sub-trees under the base distinguished name (DN) specified with the **–d** flag.

The **tetoldif** command reads the Trusted Execution LDAP database reference names from the **/etc/nscontrol.conf** file if it is present. If the specified names are unavailable in the **/etc/nscontrol.conf** file, then the default names will be used. The default names are *TSD* for the TSD and *TEPOL* for the TE Policy.

Flags

Item	Description
-d < *BaseDN* >	Specifies the base distinguished names (DN) under which to place the TSD and TE policies data. For example, *cn=aixdata*.
-s [*filename*]	Specifies the signature database. It will print only the TSD database to ldif format. If the filename is used, the default TSD **/etc/security/tsd/tsd.dat** data file can be changed to the filename.
-p [*filename*]	Specifies the TE policies database. It will print only the TE policies database to LDIF format. If the filename is used, the default TE Policies **//etc/security/tsd/tepolicies.dat** file is changed to the filename.

Exit Status

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: This command should grant execute (x) access only to the root user.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files:

Item	Description
/etc/security/tsd/tsd.dat	Contains the TSD attributes for the binaries which are configured.
/etc/security/tsd/ tepolicies.dat	Contains the TE policies configured.

Examples

1. To export the TSD and TE policies database content to a ldif format with the base DN of cn=aixdata, run the following command:

```
tetoldif –d cn=aixdata
```

2. To export only a TSD database to a ldif format with the base DN of cn=aixdata, run the following command:

```
 tetoldif –d cn=aixdata –s
```

3. To export only a TE policies database content to a ldif format with the base DN of cn=aixdata, run the following command:

```
 tetoldif –d cn=aixdata –p
```

4. To export only a TSD database from a different file than the default **/etc/security/tsd/tepolicies.dat** file to a ldif format with the base DN of cn=aixdata, run the following command:

```
 tetoldif –d cn=aixdata –s filename
```

5. To export TE policies from a different file than the default **/etc/security/tsd/tepolicies.dat** file to a ldif format with the base DN of cn=aixdata, run the following command:

```
tetoldif -d cn=aixdata -p filename
```

tftp or utftp Command

Purpose

Transfers files between hosts using the Trivial File Transfer Protocol (TFTP).

Syntax

{**tftp** | **utftp**}{ **-g** | **-o** | **-p** | **-r** | **-w** } *LocalName HostPort RemoteName* [**netascii** | **image**] [`blksize` *#*] [`timeout` *#*] [`tsize`]

Interactive Form Syntax

Command Line Form Syntax

Description

The **/usr/bin/tftp** and **utftp** commands transfer files between hosts using the Trivial File Transfer Protocol (TFTP). Since TFTP is a minimal file transfer protocol, the **tftp** and **utftp** commands do not provide all of the features of the **ftp** command. For example, the **tftp** and **utftp** commands do not provide the ability to list remote files or change directories at the remote host, and only limited file access privileges are given to the remote TFTP server. The **utftp** command is a form of the **tftp** command for use in a pipe.

The remote host must have a **tftpd** daemon started by its **inetd** daemon and have an account defined that limits the access of the **tftpd** daemon. Use the procedure defined by the **tftpd** command to setup the TFTP environment and the nobody account.

> **Note:** The **tftp** and **utftp** commands should not be available when your host is operating in secure mode.

The **tftp** command ignores duplicate acknowledgments for any block sent and sends an error packet and exit if a block with an inappropriate (future) block number arrives. It also ignores duplicate data blocks if they have already been received and sends an error packet and exits.

RFC2349 Option Negotiation

The `tftp` client is capable of negotiating the following TFTP options with the server: block size (`blksize`), transfer size (`tsize`), and timeout (`timeout`). Larger transfer block size can improve transfer performance, `tsize` reports the file size before the transfer to check for available space, and `timeout` negotiates the retransmit timeout. The TFTP server must support RFC2349 for option negotiation to take place.

Access Control

The **/etc/tftpaccess.ctl** file is searched for lines that start with `allow:` or `deny:`. Other lines are ignored. If the file doesn't exist, access is allowed. The allowed directories and files can be accessed and the denied directories cannot be accessed. For example, the **/usr** directory might be allowed and the **/usr/ucb** directory might be denied. This means that any directory or file in the **/usr** directory, except the **/usr/ucb** directory, can be accessed. The entries in the **/etc/tftpaccess.ctl** file must be absolute path names.

The **/etc/tftpaccess.ctl** file should be write-only by the root user and readable by all `groups` and `others` (that is, owned by `root` with permissions of 644). The user `nobody` must be able to read the **/etc/tftpaccess.ctl** file. Otherwise, the **tftpd** daemon is not able to recognize the existence of the file and allows access to the entire system. For more information, refer to the sample **tftpaccess.ctl** file, which resides in the **/usr/samples/tcpip** directory.

The search algorithm assumes that the local path name used in the **tftp** command is an absolute path name. It searches the **/etc/tftpaccess.ctl** file looking for `allow:/`. It repeatedly searches for allowed path names with each partial path name constructed by adding the next component from the file path name. The longest path name matched is the one allowed. It then does the same with denied names, starting with the longest allowed path name matched.

For example, if the file path name were **/a/b/c** and the **/etc/tftpaccess.ctl** file contained `allow:/a/b` and `deny:/a`, one allowed match would be made (`/a/b`) and no denied match starting with `/a/b` would be made, and access would be allowed.

I\f the **/etc/tftpaccess.ctl** file contained `allow:/a` and `deny:/a/b`, one allowed match would be made (`/a`) and one denied match starting with `/a` (`/a/b`) would be made, and access would be denied. If the **/etc/tftpaccess.ctl** file contained `allow:/a/b` and also contained `deny:/a/b`, access would be denied because allowed names are searched first.

> **Note:** Further information and example configurations for Xstations, Diskless clients, and restricted entry can be found in the **/usr/samples/tcpip/tftpaccess.ctl** file.

The **tftp** and **utftp** commands have two forms: interactive form and command-line form.

Interactive Form

In the interactive form, the **tftp** and **utftp** commands are issued alone or with a *Host* parameter that specifies the default host to use for file transfers during this session. If you choose, you can also specify with the *Port* parameter which port the **tftp** or **utftp** connection should use, such as the one specified for **mail** in the **/etc/services** file. When you enter the interactive form of either of these commands, the `tftp>` prompt is displayed.

When transferring data to a remote host, the transferred data is placed in the directory specified by the *RemoteName* parameter. The remote name must be a fully specified file name, and the remote file must both exist and have write permission set for others. The **tftp** command attempts to write the data to the specified file. However, if the remote TFTP server does not have the appropriate privileges to write the remote file or if the file does not already exist, the transfer is unsuccessful. This can be overridden using the **tftpd** daemon.

Command-Line Form

The command-line forms of the **tftp** and **utftp** commands are equivalent, except that the **utftp** command does not overwrite a local file. The **tftp** command can overwrite a file, but prompts the user before doing so. Because it is not interactive, the command line form of the **utftp** command can be more useful than the **tftp** command in a pipe. In the command line form, all of the arguments to either command are specified on the command line, and no prompt is displayed.

Subcommands

The **tftp** and **utftp** subcommands can be entered in either their interactive form or in their command-line form.

Subcommands Used in the Interactive Form

Once the `tftp>` prompt is displayed, the following subcommands can be issued:

Item	Description
? [*Subcommand*]	Displays help information. If a *Subcommand* parameter is specified, only information about that subcommand is displayed.
ascii	Synonym for the **mode ascii** subcommand.
binary	Synonym for the **mode binary** subcommand. This subcommand is used in the interactive mode. The **image** subcommand accomplishes the same thing as the **mode binary** subcommand, but is used on the command line.

Item	Description
blksize *Number of Bytes*	Enables the `blksize` option negotiation with the server. If successfully negotiated, this can substantially improve transfer rates. The transfer block size must be at least 8 octets and can be as high as 65464 octets. The default is 512 octets.
connect *Host* [*Port*]	Sets the remote host, and optionally the port, for file transfers. Since the TFTP protocol does not maintain connections between transfers, the **connect** subcommand does not create a connection to the specified host, but stores it for transfer operations. Because the remote host can be specified as part of the **get** or **put** subcommand, which overrides any host previously specified, the **connect** subcommand is not required.

get *RemoteFile* [*LocalFile*]

Item	Description
get *RemoteFile RemoteFile RemoteFile* [*RemoteFile* . . .]	Gets a file or set of files from the remote host to the local host. Each of the *RemoteFile* parameters can be specified in one of the following two ways: • As a file (*File*) that exists on the remote host if a default host has already been specified. • As a host file (*Host*:*File*), where *Host* is the remote host and *File* is the name of the file to copy to the local system. If this form of the parameter is used, the last host specified becomes the default host for later transfers in this **tftp** session.
mode *Type*	Sets the type (*Type*) of transfer mode to either **ascii** or **binary**. A transfer mode of **ascii** is the default.

put *LocalFile* [*RemoteFile*]

Item	Description
put *LocalFile LocalFile LocalFile* [*LocalFile* . . .] *RemoteDirectory*	Puts a file or set of files from the local host onto the remote host. The *RemoteDirectory* and *RemoteFile* parameters can be specified in one of the following two ways: • As a file or directory that exists on the remote host if a default host has already been specified. • With *Host*:*RemoteFile* parameter, where *Host* is the remote host and *RemoteFile* is the name of the file or directory on the remote system. If this form of the parameter is used, the last host specified becomes the default host for later transfers in this **tftp** session. In either case, the remote file or directory name must be a fully specified path name, even if the local and remote directories have the same name. If a remote directory is specified, the remote host is assumed to be a UNIX machine. The default value of the **put** subcommand is write-replace, but you can add an option in the **tftpd** daemon to allow write-create.

Item	Description
quit	Exits the **tftp** session. An End-Of-File key sequence also exits the program.
status	Shows the current status of the **tftp** program, including, for example, the current transfer mode (**ascii** or **binary**), connection status, and time-out value.
timeout *Value*	Sets the total transmission time out to the number of seconds specified by the *Value* parameter. The *Value* parameter must be 1 second or greater (the default is 5 seconds).
trace	Turns packet tracing on or off.
tsize	Enables the `tsize` option negotiation with the server. This allows the file size to be known before the transfer starts. If allocation is exceeded, an error is returned and the file transfer does not occur.
verbose	Turns verbose mode, which displays additional information during file transfer, on or off.

Subcommands Used in the Command Line Form

In this form, if the *Action* flag is:

Item	Description
-w or **-p**	Writes (or puts) local data, specified by the *LocalName* parameter, to the file specified by the *RemoteName* parameter on the remote host specified by the *Host* parameter. If the *LocalName* parameter is a file name, the **tftp** command transfers the specified local file. If the *LocalName* parameter is specified as a - (dash), the **tftp** command transfers data from local standard input to the remote host. When the *LocalName* parameter is standard input, the **tftp** command allows 25 seconds for all input to be entered before it times out.
-r or **-g** or **-o**	Reads (or gets) remote data from the file specified by the *RemoteName* parameter at the remote host specified by the *Host* parameter and writes it to the file specified by the *LocalName* parameter. If the *LocalName* parameter is a file name, the **tftp** command writes the data to the specified local file. For the **-r** and **-g** actions, the **tftp** command prompts for verification before overwriting an existing local file. For the **-o** action, the **tftp** command overwrites an existing local file without prompting. If the *LocalName* parameter is specified as a - (dash), the **tftp** command writes the data to local standard output. **Note:** Since the **tftp -g** and **tftp -r** commands prompt before overwriting an existing local file, it may be impractical to use the **tftp** command in a pipe. The **utftp** command performs the same **-r** and **-g** actions as the **tftp** command, but simply stops before overwriting a local file. Thus, the **utftp** command may be more appropriate for use in a pipe.

For both of the following modes of file transfer, the *RemoteName* parameter is the name of a file that has write permission set for others. Note that the *RemoteName* parameter must be in double quotes (" ") if it contains shell special characters.

The mode of transfer is one of the following:

Item	Description
netascii	Transfers the data as 7-bit ASCII characters in 8-bit transfer bytes. This is the default.
image	Transfers the data as 8-bit binary data bytes in 8-bit transfer bytes, with no conversion. **image** transfer can be more efficient than **netascii** transfer when transferring between two hosts. It is recommended that **netascii** be used when transferring ASCII files from a workstation to a different type of host.

Examples

The following examples distinguish the differences between the interactive form and the command line form of the **tftp** command:

Using the Interactive Form of the tftp Command

T\o enter the **tftp** command, check the current status, connect to a remote host, and transfer a file from a remote host to your local host, enter:

```
tftp
```

The tftp> prompt is displayed. Enter the **status** subcommand following this prompt:

```
status
```

A message similar to the following is displayed on your screen:

```
Not connected.
Mode: netascii   Verbose: off   Tracing: off
Max-timeout: 25 seconds
tftp> _
```

After the tftp> prompt, enter the **connect** subcommand and the name of the remote system to which you want to connect:

```
tftp>  connect host1
```

The tftp> prompt is displayed as an indication that you are connected to host1. Following the tftp> prompt, enter the **get** subcommand to transfer the file update from the remote host to your local host.

```
get /home/alice/update update
```

The /home/alice directory on the remote host must have read permission set for others. The /home/alice/update file from host1 was transferred to the update file on your local system. In this example, the user is connected to host1 and the update file is transferred from host1 to the local host.

Using the Command Line Form of the tftp Command

1. To copy a text file from a remote host and write it to a local file, enter:

   ```
   tftp  -g newsched host1 /home/john/schedule
   $ _
   ```

   In this example, the /home/john/schedule file was copied from the remote host host1 and written to the local file newsched.
2. To copy a file from a remote host and redirect the output to standard output of the local host, enter:

   ```
   tftp  -g - host3 /etc/hosts
   ```

   If the copy is successful, information similar to the following is displayed on your screen:

   ```
   192.100.13.3 nameserver
   192.100.13.3 host2
   192.100.13.5 host1
   192.100.13.7 host3
   192.100.13.3 timeserver
   ```

```
Received 128 bytes in 0.4 seconds
$ _
```

In this example, the `/etc/hosts` file from remote host `host3` was copied and the output redirected to standard output of the local host.

3. To copy a file from a remote host, pipe it to the **grep** command, and write it to a local file, enter:

```
utftp -g - host1 /home/john/schedule | grep Jones > jones.todo
$ _
```

In this example, the `/home/john/schedule` file was copied from the remote host `host1`. This file was then piped to the **grep** command and written into the local file `jones.todo`.

4. To copy a file to another system, enter:

```
tftp -p /home/jeanne/test host2 /tmp/test
```

If the copy is successful, information similar to the following is displayed on your screen:

```
Sent 94146 bytes in 6.7 seconds
```

In this example, the `/home/jeanne/test` file was sent to the `/tmp` directory on the remote host `host2`.

5. To copy a binary file to another system, enter:

```
tftp -p core host3 /tmp/core image
```

If the copy is successful, information similar to the following is displayed on your screen:

```
Sent 309295 bytes in 15 seconds
```

In this example, the binary file `core` from the current directory was sent to the `/tmp` directory on remote host `host3`.

Files

Item	Description
/etc/tftpaccess.ctl	Allows or denies access to files and directories.

tftpd Daemon

Purpose

Provides the server function for the Trivial File Transfer Protocol.

Syntax

/usr/sbin/tftpd [**-c**] [**-n**] [**-p**] [**-v**] [**-t**] [**-s**] [**-x**] [**-z**] [**-d** *Directory*] [**-r** *Option*]

Description

> **Note:** The **tftpd** daemon is normally started by the **inetd** daemon. It can also be controlled from the command line, using SRC commands.

The **/usr/sbin/tftpd** daemon runs the Trivial File Transfer Protocol (TFTP) server. Files sent using TFTP can be found in the directory specified by the full path name given on the **tftp** or **utftp** command line.

Note: The **tftp** command, **utftp** command, and **tftpd** server are not available when the auditing system is in use. For more information, see **TCP/IP Security,** the **Auditing overview**, and the **audit** command.

Changes to the **tftpd** daemon can be made using the System Management Interface Tool (SMIT) or System Resource Controller (SRC), by editing the **/etc/inetd.conf** or **/etc/services** file. The **tftpd** daemon is started by default when it is uncommented in the **/etc/inetd.conf** file.

The **inetd** daemon get its information from the **/etc/inetd.conf** file and the **/etc/services** file.

After changing the **/etc/inetd.conf** or **/etc/services** file, run the **refresh -s inetd** or **kill -1** *InetdPID* command to inform the **inetd** daemon of the changes to its configuration file.

The **tftpd** server should have a user ID with the least privileges possible. The **nobody** ID allows the least permissions, and is the default user ID.

The **tftpd** daemon should be controlled using the System Management Interface Tool (SMIT) or by changing the **/etc/inetd.conf** file. Entering `tftpd` at the command line is not recommended.

The **tftpd** server is a multithreaded application and is able to handle option negotiation (RFC2349). This capability allows a client to negotiate a file size to be transferred. It also allows for a timeout and a larger block size. Block size (`blksize`) is negotiated for the read requests (RRQ) only. As a result, the boot time performance of diskless nodes using TFTP can improve significantly.

The Transfer Size option (`tsize`) negotiation for both read and write requests allows the file size to be known before the transfer, resulting in an error message if allocation exceeded before the transfer started. The timeout option (`timeout`) allows for the client and the server to negotiate a retransmit timeout (between 1 and 255 seconds). The `tftp` client must also support RFC2349 for the option negotiation to take place.

tftpaccess.ctl File

The **/etc/tftpaccess.ctl** file is searched for lines that start with `allow:` or `deny:`. Other lines are ignored. If the file doesn't exist, access is allowed. The allowed directories and files minus the denied directories and files can be accessed. For example, the **/usr** directory might be allowed and the **/usr/ucb** directory might be denied. This means that any directory or file in the **/usr** directory, except the **/usr/ucb** directory, can be accessed. The entries in the **/etc/tftpaccess.ctl** file must be absolute path names.

The **/etc/tftpaccess.ctl** file should be write-only by the root user and readable by all `groups` and `others` (that is, owned by `root` with permissions of 644). The user `nobody` must be able to read the **/etc/tftpaccess.ctl** file. Otherwise, the **tftpd** daemon is not able to recognize the existence of the file and allows access to the entire system. For more information, refer to the sample **tftpaccess.ctl** file, which resides in the **/usr/samples/tcpip** directory.

The search algorithm assumes that the local path name used in the **tftp** command is an absolute path name. It searches the **/etc/tftpaccess.ctl** file looking for `allow:/`. It repeatedly searches for allowed path names with each partial path name constructed by adding the next component from the file path name. The longest path name matched is the one allowed. It then does the same with denied names, starting with the longest allowed path name matched.

For example, if the file path name were **/a/b/c** and the **/etc/tftpaccess.ctl** file contained `allow:/a/b` and `deny:/a`, one allowed match would be made (`/a/b`) and no denied match starting with `/a/b` would be made, and access would be allowed.

If the **/etc/tftpaccess.ctl** file contained `allow:/a` and `deny:/a/b`, one allowed match would be made (`/a`) and one denied match starting with `/a` (`/a/b`) would be made, and access would be denied. If the **/etc/tftpaccess.ctl** file contained `allow:/a/b` and also contained `deny:/a/b`, access would be denied because allowed names are searched first.

Manipulating the tftpd Daemon with the System Resource Controller

The **tftpd** daemon is a subserver of the **inetd** daemon, which is a subsystem of the System Resource Controller (**SRC**). The **tftpd** daemon is a member of the **tcpip** SRC subsystem group. This daemon is enabled when it is uncommented in the **/etc/inetd.conf** file and can be manipulated by the following SRC commands:

Item	Description
startsrc	Starts a subsystem, group of subsystems, or a subserver.

Item	Description
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
lssrc	Gets the status of a subsystem, group of subsystems, or a subserver.

Flags

Item	Description
-c	Specifies the maximum number of concurrent threads per process, excluding the initial thread.
-d *Directory*	Specifies default destination directory. The *Directory* specified will be used as the home directory for storing files only. This default directory will be used only if a full pathname is not specified. The default directory for retrieving files is still **/tftpboot**.
-i	Logs the IP address of the calling machine with error messages.
-n	Allows the remote user to create files on your machine. Remote users are only allowed to read files with read permission for other if this flag is not specified.
-p	Specifies the port number for the incoming request.
-r *Option*	Specifies a `tftp` option negotiation to disable. Multiple `-r` flags can be used. For example, the following line in the `/etc/inetd.conf` file disables option negotiation for `tsize` and `blksize`: `tftp dgram udp6 SRC nobody /usr/sbin/tftpd tftpd -n -r tsize -r blksize`
-s	Turns on socket-level debugging.
-t	Specifies the timeout value for datagrams.
-v	Logs information messages when any file is successfully transferred by the **tftpd** daemon. This logging keeps track of who is remotely transferring files to and from the system with the **tftpd** daemon.
-x	Specifies the maximum of timeouts waiting for a datagram.
-z	Specifies the maximum allowed segment size for transfers.

Examples

Note: The arguments for the **tftpd** daemon can be specified by using SMIT or by editing the **/etc/inetd.conf** file.

1. To start the **tftpd** daemon, enter the following:

   ```
   startsrc  -t tftp
   ```

   This command starts the **tftpd** subserver.
2. To stop the **tftpd** daemon normally, enter the following:

   ```
   stopsrc  -t tftp
   ```

   This command allows all pending connections to start and existing connections to complete but prevents new connections from starting.
3. To force stop the **tftpd** daemon and all **tftpd** connections, enter the following:

   ```
   stopsrc -f -t tftp
   ```

   This command terminates all pending connections and existing connections immediately.
4. To display a short status report about the **tftpd** daemon, enter the following:

```
lssrc -t tftp
```

This command returns the daemon's name, process ID, and state (active or inactive).

tic Command

Purpose

Translates the terminfo description files from source to compiled format.

Syntax

tic [**-v** [*Number*]] [**-c**] *FileName*

Description

The **tic** command translates the terminfo files from the source format into the compiled format. The **tic** command places the results in the **/usr/share/lib/terminfo** directory. If the **TERMINFO** environment variable is set, the results are placed there instead of in the **/usr/share/lib/terminfo** directory.

The **tic** command compiles all terminfo descriptions in *FileName*. When the **tic** command finds a use= entry-*name* field, it searches the current file first, If unable to find the entry -*name*, it obtains the entry from the binary file in **/usr/share/lib/terminfo**. If **TERMINFO** is set, the terminfo directory is searched before **/usr/share/lib/terminfo**.

The total compiled entries cannot exceed 4096 bytes, and the name field cannot exceed 128 bytes.

Flags

Item	Description
-v[*Number*]	Writes trace information on the progress of the **tic** command. *Number* is an integer from 1 to 10 inclusive that increases the level of the verbosity. If *Number* is omitted, the default level is 1. The amount of information output increases as *Number* increases.
-c	Only checks *FileName* for errors. Errors in *use=entry-name* are not detected.

Files

Item	Description
/usr/share/lib/terminfo/?/*	Contains the compiled terminal capability database.

time Command

Purpose

Prints the time of the execution of a command.

Syntax

time [**-p**] *Command* [*Argument* ...]

Description

The **time** command prints the elapsed time during the execution of a command, time in the system, and execution time of the **time** command in seconds to standard error.

Note: Sleep time is not charged to either system or user time.

The **time** command is also built into the C shell (**csh**) and Korn shell (**ksh**) with a different format. To run the **time** command while in the **csh** and **ksh** shells, enter:

```
/usr/bin/time
```

Flags

Item	Description
-p	Writes the timing output to standard error. Seconds are expressed as a floating-point number with at least one digit following the radix character. The standard format for this flag is as follows: `"real %f\nuser %f\nsys %f\n", <real seconds>, <user seconds>, <system seconds>`

Exit Status

If you use the *Command* parameter, the exit status of the **time** command is the exit status of the specified command. Otherwise, the **time** command exits with one of the following values:

Item	Description
1-125	Indicates an error occurred in the **time** command.
126	Indicates the command specified by the *Command* parameter was found but could not be invoked.
127	Indicates the command specified by the *Command* parameter could not be found.

Examples

1. To measure the time required to run a program, enter:

   ```
   /usr/bin/time -p a.out
   ```

   This command runs the program **a.out** and writes the amount of real, user, and system time to standard error, in the format specified by the **-p** flag; for example:

   ```
   real      10.5
   user      0.3
   sys       3.6
   ```

2. To save a record of the **time** command information in a file, enter:

   ```
   /usr/bin/time a.out 2> a.time
   ```

Files

Item	Description
/usr/bin/time	Specifies the path of the **time** command.

timed Daemon

Purpose

Invokes the time server daemon.

Syntax

/usr/sbin/timed [**-c**] [**-M**] [**-t**] [[**-n** *Network*] ... | [**-i** *Network*] ...]

> **Note:** Use the **rc.tcpip** file to start the daemon with each initial program load. You can specify the **timed** daemon at the command line. You can also use SRC commands to control the **timed** daemon from the command line.

Description

The **timed** daemon synchronizes one machine's clock with those of other machines on the local area network that are also running the **timed** daemon. The **timed** daemon slows the clocks of some machines and speeds up the clocks on other machines to create an average network time.

When the **timed** daemon is started without the **-M** flag, the machine locates the nearest master time server and asks for the network time. Then the machine uses the **date** command to set the machine's clock to the network time. The machine accepts synchronization messages periodically sent by the master time server and calls the **adjtime** subroutine to perform the needed corrections on the machine's clock.

When the **timed** daemon is started with the **-M** flag, the machine polls each of its local area networks to determine which networks have master time servers. The machine becomes a master time server on the networks that do not have a master time server. The machine becomes a submaster time server on the networks that already have a master time server. The **timed** daemon creates the **/var/adm/timed.masterlog** file when the **timed** daemon is started with the **-M** flag. The **/var/adm/timed.masterlog** file contains a log of the deltas between the local machine's clock and the clocks of the other machines on the networks for which the local machine is the master time server. The **/var/adm/timed.masterlog** file is updated approximately every 4 minutes and is never cleared. You may need to clear this file to conserve disk space. If the machine is only a submaster time server on its networks, the **/var/adm/timed.masterlog** file remains empty. To clear the **/var/adm/timed.masterlog** file, enter:

```
cat /dev/null > /var/adm/timed.masterlog
```

If the master time server ceases to function on a network, a new master time server is elected from the submaster time servers on that network. The **timedc** command enables you to select which submaster time server becomes the master time server.

The **timed** daemon can be controlled using the System Resource Controller (SRC), the System Management Interface Tool (SMIT), or the command line. The **timed** daemon is not started by default. Use the **rc.tcpip** file to start the **timed** daemon with each initial program load.

Manipulating the timed Daemon with the System Resource Controller

The **timed** daemon is a subsystem controlled by the **SRC**. The **timed** daemon is a member of the SRC **tcpip** system group. Use the following SRC commands to manipulate the **timed** daemon:

Item	Description
startsrc	Starts a subsystem, group of subsystems, or a subserver.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
lssrc	Gets the short status of a subsystem, group of subsystems, or a subserver. The long status option usually found in **lssrc** is not supported for the timed daemon.

Flags

Item	Description
-c	Specifies that the master-timed daemon should ignore the time values it gets from the other worker-timed daemons when for calculating the average network time. This flag changes the network time to be the same as the system clock on the master-timed daemon.
-i *Network*	Specifies a network to be excluded from clock synchronization. The *Network* variable can be either a network address or a network name. If a network name is specified for the *Network* variable, the network name must be defined in the **/etc/networks** file. Specify one network address or network name with each **-i** flag. Do not use this flag with the **-n** flag.
-M	Specifies the machine is a master or submaster time server on its local area networks. If a master time server is not currently available on a network, the machine becomes the master time server for that network. If a master time server already exists on a network, the machine becomes a submaster time server on that network. However, the machine can become the master time server if the current master time server becomes inoperative. The **timed** daemon creates the **/var/adm/timed.masterlog** file when the **timed** daemon is started with the **-M** flag.
-n *Network*	Specifies a network to include in clock synchronization. The *Network* variable can be either a network address or a network name. If a network name is specified for the *Network* variable, the network name must be defined in the **/etc/networks** file. Specify one network address or network name with each **-n** flag. Do not use this flag with the **-i** flag.
-t	Allows the **timed** daemon to trace the messages it receives and store them in the **/var/adm/timed.log** file. You can also use the **timedc** command to activate tracing.

Examples

1. To start the **timed** daemon with SRC control, enter:

   ```
   startsrc -s timed
   ```

   This command starts the daemon. You can use this command in the **rc.tcpip** file or on the command line. The **-s** flag specifies that the subsystem that follows is to be started.
2. To stop the **timed** daemon normally with SRC control, enter:

   ```
   stopsrc -s timed
   ```

   This command stops the daemon. The **-s** flag specifies that the subsystem that follows is to be stopped.
3. To get a short status report from the **timed** daemon, enter:

   ```
   lssrc -s timed
   ```

   This command returns the name of the daemon, the process ID of the daemon, and the state of the daemon (active or inactive).
4. To start the **timed** daemon with SRC control as the master or submaster time server and to exclude networks `net1` and `net2` from clock synchronization, enter:

   ```
   startsrc -s timed -a "-M -i net1 -i net2"
   ```

   This command starts the daemon. The machine becomes the master or submaster time server for its networks. Networks `net1` and `net2` are excluded from clock synchronization. The **-s** flag specifies that

the subsystem that follows is to be started. The **-a** flag specifies that the **timed** daemon should be started with the flags that follow. The flags must be enclosed in quotes.

5. To start the **timed** daemon, activate tracing, and include `net1` and `net2` in clock synchronization, enter:

```
timed -t -n net1 -n net2
```

This command starts the daemon. Tracing is activated and both `net1` and `net2` are included in clock synchronization.

Files

Item	Description
/var/adm/timed.log	Contains the messages traced for the **timed** daemon. This file is created when the **timed** daemon is started with the **-t** flag or when tracing is enabled with the **timedc** command.
/etc/rc.tcpip	Contains the SRC commands to be executed at system startup.
/var/adm/timed.masterlog	Contains a log of the deltas between the master time server clock and the clocks of the other machines on the networks. This file is created when the **timed** daemon is started with the **-M** flag. However, this file only contains information for those networks on which the machine is the master time server.

timedc Command

Purpose

Returns information about the **timed** daemon.

Syntax

timedc [*Subcommand* [*Parameter* ...]]

Description

The **timedc** command controls the operation of the **timed** daemon. The **timedc** command does the following:

- Measures the difference between clocks on various machines on a network.
- Finds the location of the master time server.
- Enables or disables tracing of messages received by the **timed** daemon.
- Debugs.

Without any variables, the **timedc** command assumes an interactive mode and prompts for subcommands from standard input. If variables are supplied, the **timedc** command interprets the first variable as a subcommand and the remaining variables as parameters to the subcommand. You can redirect standard input so the **timedc** command reads subcommands from a file.

Variables

The **timedc** command recognizes the following subcommands:

Item	Description
? [*Parameter* ...]	Displays a short description of each variable specified in the parameter list. The **?** subcommand only works in interactive mode. If you give no variables, the **?** subcommand shows a list of subcommands recognized by the **timedc** command.
clockdiff *Host* ...	Computes the differences between the clock of the host machine and the clocks of the machines given as variables.
election *Host* ...	Requests that the **timed** daemon on the specified host (s) reset its election timers and ensure that a **timed** master server is available. Up to 4 hosts can be specified. If a master **timed** server is no longer available, then the **timed** daemon on the specified host (s) will request to become the new **timed** master server. The specified host(s) must be running the **timed** daemon in submaster mode with the **-M** flag.
help [*Parameter* ...]	Displays a short description of each subcommand specified in the parameter list. If you give no variables, the **help** subcommand shows a list of subcommands recognized by the **timedc** command.
msite	Finds the location of the master site.
quit	Exits the **timedc** command.
trace { **on** | **off** }	Enables or disables tracing of incoming messages to the **timed** daemon. The messages are held in the **/var/adm/timed.log** file.

You can use other commands for testing and debugging the **timed** daemon. Use the **help** command to find these commands.

These error messages may occur with the **timedc** command:

Item	Description
`Ambiguous command`	Abbreviation matches more than one command.
`Invalid command`	No match found.
`Privileged command`	Command can be executed only by the root user.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Examples

1. To display the time difference between the local host `sahara` and the remote host `sandy`, enter:

   ```
   timedc clockdiff sandy
   ```

   The output would be:

   ```
   time on sandy.austin.century.com is 37904247 ms ahead of time on
   sahara.austin.century.com
   ```

2. To display the client location of the **timed** daemon, enter:

```
timedc msite
```

The output would be:

```
client timed daemon runs on bupu.austin.century.com
```

timex Command

Purpose

Reports, in seconds, the elapsed time, user time, and system execution time for a command.

Syntax

timex [**-o**] [**-p**] [**-s**] *Command*

Description

The **timex** command reports, in seconds, the elapsed time, user time, and system execution time for a command. With specified flags, the **timex** command lists or summarizes process accounting data for a command and all of its children. *Command* is the name of any executable file on the system. It also reports total system activity during the execution interval. Output is written to standard error. The system uses the **/var/adm/pacct** file to select process records associated with the command and includes background processes with the same user ID, workstation ID, and execution time window.

Flags

Item	Description
-o	Reports the total number of blocks read or written and total characters transferred by a command and all its children.
-p	Lists process accounting records for a command and all its children. The number of blocks read or written and the number of characters transferred are reported. The **-p** flag takes the **f**, **h**, **k**, **m**, **r**, and **t** arguments defined in the **acctcom** command to modify other data items. **-f** Print the fork/ exec flag and system exit status columns in the output. **-h** Instead of mean memory size, shows the fraction of total available CPU time consumed by the process (hogfactor). **-k** Instead of memory size, shows total kcore minutes (memory measurement in kilobyte segments used per minute of run time). **-m** Shows mean main-memory size. This is the default. The **-h** flag or **-k** flag turn off the **-m** flag. **-r** Shows CPU factor. **-t** Shows separate system and user CPU times.
-s	Reports total system activity during the execution of the command. All the data items listed in the **sar** command are reported.

Note: Accounting must be turned on to use the **-o** or **-p** flags.

Examples

1. To report the total number of blocks read and total characters transferred by the **ls** command, enter:

```
timex -o ls
```

2. To list the process accounting records for the **ps** command, enter:

```
timex -p ps -fe
```

3. To report total system activity for the execution of the **ls** command, enter:

```
timex -s ls
```

Files

Item	Description
/var/adm/pacct	Used to select record associated with the command.

tip Command

Purpose

Connects to a remote system.

Syntax

tip [**-v**] [**-** *BaudRate*] { *SystemName* | *PhoneNumber* }

Description

The **tip** command connects to a remote system and allows you to work on the remote system as if logged in directly.

Either the *SystemName* parameter or the *PhoneNumber* parameter is required. The *SystemName* parameter specifies the name of a remote system to be contacted. The remote system must be defined in the **/etc/remote** file, or in the file specified by the **REMOTE** environment variable. The *PhoneNumber* parameter specifies the number to dial over a modem connection.

When the **tip** command is invoked with the *SystemName* parameter, it searches the **remote** file for an entry beginning with that system name. When the command is invoked with the *PhoneNumber* parameter, it searches the **remote** file for an entry of the form **tip***BaudRate*, where *BaudRate* is the baud rate for the connection. If the **-***BaudRate* flag is not used, the **tip** command looks for a `tip1200` entry, because 1200 is the default baud rate.

The actions of the **tip** command can be controlled using flags, escape signals and variables. The **tip** command reads the **/etc/remote** file to find out how to contact a remote system and discover the escape-send sequence to use when communicating with that system. In addition, the command may check the **/etc/phones** file to find out a phone number for the remote system.

A **tip** user can create an individual remote file in the format of the **/usr/lib/remote-file** file, and then specify the file to use with the **REMOTE** environment variable. A user can also create an individual phones file in the format of the **/usr/lib/phones-file** file, and then specify the file to use with the **PHONES** environment variable. The **tip** command does not read the **/usr/lib/remote-file** file or **/usr/lib/phones-file** file by default, however. The default files that the **tip** command uses are the **/etc/remote** file and **/etc/phones** file.

A **tip** user can create a **$HOME/.tiprc** file to specify initial settings for the **tip** variables. In addition, settings made in the remote file, the phones file, and the **.tiprc** file can be overridden by using escape

signals while **tip** is running. Escape signals can also be used, for instance, to start and stop file transfers or interrupt a connection to remote system.

The **tip** command uses lock files in the **/etc/locks** directory to lock devices against multiple access and to prevent multiple users from logging in on the same system.

When the **tip** command prompts for a response, edit the line as you type using the standard keys. Entering ~. (tilde, period) in response to a prompt, or pressing the Interrupt key, will abort the **tip** dialog and return you to the remote system.

You can use the **tip** command to transfer files to and from the remote system. You can use **tip** command escape signals to start and stop the file transfers. Several **tip** command variables work together to control file transfers.

File transfers usually use tandem mode to control the flow of data. If the remote system does not support tandem mode, set the *echocheck* variable to on to cause the **tip** command to synchronize with the remote system after transmitting each character. When transferring files with the ~< and ~> escape signals, use the **eofread** and *eofwrite* variables to specify the end of a file when writing, and recognize the end of a file when reading.

If the *verbose* variable is set on, the **tip** command performs the following:

- Writes a running count of the number of lines transferred during a file transfer.
- Writes messages indicating its actions as it dials a phone number.

You can use scripting to record the conversations you have with the **tip** command. Use the *script* variable to start scripting.

Note:

1. Only a user with root user authority can change the *dialtimeout* variable.
2. Although any user can specify a host at the command line, only the root user can change the *host* variable setting after the **tip** command has been started. However, this does not change the system to which the **tip** command is currently connected.

Flags

Item	Description
-v	Displays the settings of variables as they are read from the **.tiprc** file.
-*BaudRate*	Overrides the default baud rate, which is 1200 baud.

Escape Signals

Using escape signals, you can instruct the **tip** command to terminate, log off from the remote system, and transfer files. The escape character at the beginning of a line indicates an escape signal. The default escape character is a ~ (tilde). The character can be changed using the *escape* variable. All other typed characters are transmitted directly to the remote system. The **tip** command recognizes the following escape signals:

Item	Description
~^D~	Terminates the connection and exits. You may still be logged in on the remote system; if so, you can issue another **tip** command to reconnect to that remote system.
~c [*Directory*]	Changes, on the local system, to the directory specified by the *Directory* variable. If you do not include the *Directory* variable, the **tip** command changes to your home directory.
~!	Escapes to a shell on the local system. When you exit from the shell, you return to the **tip** command.

Item	Description
~>	Copies a file from the local system to the remote system. The **tip** command prompts you for the name of the local file.
~<	Copies a file from the remote system to the local system. The **tip** command prompts you for the name of the remote file.

A **tip** file download will only download the file until one of the EOF characters listed in the **eofread** command cariable is encountered. If one of these characters is not encountered, then the file copy will not succeed.

When downloading a file with the ~< signal, the user will be prompted for a local file name. The user may respond with any valid writeable file name. When prompted for the remote command, the user should append the EOF character to the end of the file being read.

This signal can be used as shown in the following example:

```
List command for remote system? echo "\04" | cat /etc/passwd
```

This example assumes that the character 0x4 is present in the **tip** *eofread* variable. The best way of ensuring that this character exists in the variable is to assign it in the usr's **.tiprc** file, which should reside in the user's home directory.

To accomplish this, the following command can be issued:

```
echo"eofread=\04" >> ~/.tiprc
```

Item	Description
~p *Source* [*Dest*]	Sends (puts) the *Source* file to a remote UNIX host system, using the **cat** command to copy the *Source* file to the *Dest* file. If the *Dest* file name is not specified, the **cat** command uses the name of the *Source* file. If the *Dest* file exists on the remote host, it will be replaced by the *Source* file. This signal is a UNIX-specific version of the ~> signal.
~t *Source* [*Dest*]	Transfers (takes) the *Source* file from a remote UNIX host system to the local system, using the **cat** command to copy the *Source* file to the *Dest* file on the local system. If the *Dest* file name is not specified, the **cat** command uses the name of the *Source* file. If the *Dest* file exists on the local system, it will be replaced by the *Source* file. This signal is a UNIX-specific version of the ~< signal.
~|	Pipes the output of a remote command to a local process. The command string sent to the local system is processed by the shell.

A remote pipe will only succeed if the data from the remote pipe is terminated by one of the eof characters listed in the *eofread* **tip** command variable. If one of these characters is not encountered, then the output pipe will not succeed.

When piping remote output with the ~| signal, the user will be prompted for a local command name. The user may respond with any valid command name. When prompted for the remote command, the user should append the EOF character to the end of the file being read.

This signal can be used as shown in the following example:

```
Local command? cat
List command for remote system? echo
"asdfasdfasdfasdf\04"
```

This example assumes that the character 0x4 is present in the **tip** *eofread* variable. The best way of ensuring that this character exists in the variable is to assign it in the usr's **.tiprc** file, which should reside in the user's home directory.

To accomplish this, the following command can be issued:

```
echo"eofread=\04" >> ~/.tiprc
```

Item	Description
~$	Pipes the output of a local process to the remote system. The command string sent to the remote system is processed by the shell.
~#	Sends a **BREAK** signal to the remote system.
~s { *Variable*=*Value* | [**!**]*BoolVariable* | **all** | *Variable***?** }	Sets or queries the **tip** command variables. To change the value of a non-Boolean variable, enter the variable name or abbreviation, followed by an **=** (equal sign), followed by the new value. For example, type `~s rc=^U` to change the character used to turn uppercase conversion on or off (the *raisechar* variable). To change the value of a Boolean variable, enter the variable name or abbreviation. To reset the variable to its default value, type an **!** (exclamation point) in front of the name. For example, type `~s !ec` to reset the *echocheck* variable to its default value. To display all variables readable by the user, specify **all** as an argument to the **~s** signal. You may also request the display of a specific variable by attaching a **?** (question mark) to the variable name. For example, type the command `~s eol?` to display the current end-of-line string (the *eol* variable).
~^Z	Stops the **tip** command. The **~^Z** signal is only available with job control.
~^Y	Stops the local portion of the **tip** command. The remote portion, which displays the output from the remote system, continues to run. The **~^Y** signal is only available with job control.
~?	Displays a list of the escape signals.

Variables

The **tip** command uses variables that control its operation. These variables may be numeric, string, character, or Boolean values. Some of these variables can be changed by any user who can run the **tip** command. However, the following variables can be changed only by a user with root user authority: the *baudrate* variable and the *dialtimeout* variable.

Variables may be initialized at run time in the **$HOME/.tiprc** file. Additionally, you can display and set the variables while already running the **tip** command by using the **~s** escape signal.

Variables may be numeric, string, character, or Boolean values. To set a non-Boolean variable, enter the variable name or abbreviation followed by an **=** (equal sign) and the value. For example, type either `~s host=zeus` or `~s ho=zeus` to change the **host** name to `zeus`. In the **.tiprc** file, type `host=zeus` or `ho=zeus`.

To change the value of a Boolean variable, enter the variable name or abbreviation as an argument to the **~s** signal or on a line of the **.tiprc** file. To reset the variable to its default value, type an `!` (exclamation point) in front of the name. For example, type `~s !echocheck` to reset the *echocheck* variable to its default value while running the **tip** command.

Following are the common variables, their types, abbreviations, and default values.

Variable (Abbreviation)	Type	Description
beautify (be)	Boolean	Instructs the **tip** command to discard unprintable characters when a session is being scripted. Does not discard characters specified with the *exceptions* variable. The default setting is on.
baudrate (ba)	Numeric	Reflects the baud rate of the connection. Changing the value of this variable will *not* change the current baud setting of the connected tty device.
dialtimeout (dial)	Numeric	Specifies the time in seconds that the **tip** command waits for a connection when dialing a phone number. The default is 60 seconds. The **dialtimout** setting can be changed only by someone with root user authority.
echocheck (ec)	Boolean	Instructs the **tip** command to synchronize with the remote system during a file transfer by awaiting the echo of the last character transmitted before transmitting the next character. The default setting is `off`.
eofread (eofr)	String	Specifies the set of characters that signifies end-of-transmission during a remote-to-local (~< or ~t) file transfer.
eofwrite (eofw)	String	Specifies the string that is sent to indicate the end of a transmission during a local-to-remote (~> or~p) file transfer.
eol (none)	String	Specifies the string that indicates the end of a line. The **tip** command recognizes escape signals only when they follow an end-of-line string.
escape (es)	Character	Specifies the character prefix for escape signals. The default is ~ (tilde).
etimeout (et)	Numeric	Specifies the time to wait for a response when the *echocheck* variable is set on. If the echo is not received within the designated time, the file transfer is discontinued. The default time is 28 seconds.
exceptions (ex)	String	Specifies the set of characters that should not be discarded even when the **beautify** switch is set to on. The `\t\n\f\b` string is the default.
force (fo)	Character	Specifies the character that is used to force literal data transmissions during binary transfers. The ^P character is the default. Literal data transmissions are off until the user types the character specified by the *force* variable.
framesize (fr)	Numeric	Specifies the number of bytes to buffer between files system writes when receiving files from the remote system.
host (ho)	String	Specifies the name of the remote system to which you were connected when the **tip** command was invoked. This variable cannot be changed.
halfduplex (hdx)	Boolean	Toggles Half-duplex mode. The default setting is `off`.
localecho (le)	Boolean	Toggles the Local-echo mode. The default setting is `off`.

Variable (Abbreviation)	Type	Description
log (none)	String	Defines the file used to log dial-outs with the **tip** command. The default file is the **/var/spool/uucp/.Admin/aculog** file. The log file can be changed only by someone with root authority.

Variable (Abbreviation)	Type	Description
parity (par)	String	Defines the parity for file transfers. Defaults to the following string: `no parity, 8 data bits`
phones (none)	String	Specifies the name of the user's phone file. The file can have any valid file name and must be set up in the format of the **/usr/lib/phones-file** file. The default is the **/etc/phones** file. If a file is specified with the **PHONES** environment variable, it is used in place of (not in addition to) the **/etc/phones** file.
prompt (pr)	Character	Specifies the character that indicates the end of the line on the remote host. This character is used to synchronize during data transfers. The **tip** command counts lines transferred during a file transfer, based on the number of times it receives the **prompt** character. The `\n` character is the default.
raise (ra)	Boolean	When set to on, instructs the **tip** command to convert all lowercase letters to uppercase before transmitting them to the remote system. The default setting is `off`.
raisechar (rc)	Character	Specifies a character that is used to toggle uppercase conversion. The ^A character is the default.
rawftp (raw)	Boolean	If the *rawftp* variable is set to on, data is transmitted over the connection during a file transfer with no additional processing carried out. That is, when sending files, line-feeds are not mapped to line-feed/carriage carried out.
record (rec)	String	Specifies the name of the file in which the **tip** command records the session script. The **tip.record** file is the default. The **tip** command places the file in the user's current directory on the local system.
remote (none)	String	Specifies the name of the user's remote system definition file. The file can have any valid file name and must be set up in the format of the **/usr/lib/remote-file** file. The default is the **/etc/remote** file. If a file is specified with the **REMOTE** environment variable, it is used in place of (not in addition to) the **/etc/remote** file.
script (sc)	Boolean	When the **script** switch is set on, the **tip** command records everything transmitted by the remote system in a file on the local system. The file name is specified by the *record* variable. If the **beautify** switch is set to on, only printable ASCII characters (those between 040 and 0177) will be recorded in the script file. The *exceptions* variable specifies unprintable characters that will be recorded even if the **beautify** switch is set to on. The default setting for the **script** switch is `off`.

Variable (Abbreviation)	Type	Description
tabexpand (tab)	Boolean	Causes the **tip** command to expand tab characters to eight spaces during file transfers. The default setting is `off`.
verbose (verb)	Boolean	When the **verbose** switch is set on, the **tip** command prints messages while dialing, shows the current number of lines transferred during a file transfer, and displays other status information about the connection. The default setting is on.
SHELL (none)	String	Specifies the type of shell to use for the **~!** signal. The default value is **/usr/bin/sh** or is taken from the environment.
HOME (none)	String	Specifies the home directory to use for the **~c** signal. The default value is taken from the environment.

Examples

1. To specify a baud rate when making a direct connection, type:

   ```
   tip -300 hera
   ```

   This instructs the **tip** command to use baud rate of `300` when contacting remote system `hera`.
2. To use a modem to connect to a remote system, type:

   ```
   tip 9,343-2132
   ```

   The **tip** command connects the local system to the remote system reached by the telephone number `343-2132`, after dialing a `9` to reach an outside line.
3. To connect directly to a remote system and display the variables, type:

   ```
   tip  -v hera
   ```

   The **-v** flag causes the **tip** command to display the values of the variables as it reads them from the **$HOME/.tiprc** file. If the **.tiprc** file contains the following settings:

   ```
   sc
   be
   rec=/home/jimk/callout
   ```

   then output from the **-v** flag is as follows:

   ```
   set script
   set beautify
   set record=/home/jimk/callout
   ```

Files

Item	Description
/usr/bin/tip	Contains the **tip** command.
/etc/locks/*	Contains lock files that prevent multiple uses of devices and multiple calls to systems.
/etc/remote	Contains system descriptions for the **tip** command. If the *remote* variable or the **REMOTE** environment variable is set, that file is used instead.

Item	Description
/usr/lib/remote-file	Contains sample **remote** file. If the*remote* variable or the **RECORD** environment variable is set, that file is used instead.
/etc/phones	Contains the telephone number database for the **tip** command. If the *phones* variable or the **PHONES** environment variable is set, that file is used instead.
/usr/lib/phones-file	Contains the telephone number database for the **tip** command. If the *phones* variable or the **PHONES** environment variable is set, that file is used instead.
$HOME/.tiprc	Defines initial settings for the **tip** command variables.
tip.record	Contains the **tip** command scripts. By default, the file is stored in the current directory. The user can change the file name and directory using the *record* variable.

tncconsole Command

Purpose

Reports and manages the trusted network connect (TNC) server, the TNC client, the TNC IP Referrer (IPRef), and Service Update Management Assistant (SUMA). It manages fileset and patch management policies regarding endpoint (server and client) integrity at or after network connection to protect the network from threats and attacks.

Note: This command is used to demonstrate **TNC** options and has limited functionality. To use the full function of this command, install PowerSC Standard Edition. In PowerSC Standard Edition, the name of the **tncconsole** command was changed to the **psconf** command.

Syntax

TNC server operations:

tncconsole mkserver [**tncport=**<*port*>] **pmserver=**<*host*:*port*> [tsserver=<host>]
[**recheck_interval=**<*time_in_minutes*> | **d** (days) : **h** (hours) : **m** (minutes)] [**dbpath** = *<user-defined directory>*]

tncconsole { **rmserver** | **status** }

tncconsole { **start** | **stop** | **restart** } **server**

tncconsole chserver attribute = *value*

tncconsole add -F <*FSPolicyname*> -r <*buildinfo*> [**apargrp=** [**±**]<*apargrp1, apargrp2..* >]
[*ifixgrp*=[+|-]<*ifixgrp1,ifixgrp2...*>]

tncconsole add { **-G** <*ipgroupname*> **ip=**[**±**]<*host1, host2...*> | {**-A**<*apargrp*> [**aparlist**=[**±**]*apar1, apar2...* |
{**-V** <*ifixgrp*> [*ifixlist*=[+|-]*ifix1,ifix2...*]}

tncconsole add -P <*policyname*> { **fspolicy=**[**±**]<*f1,f2...*> | **ipgroup=**[**±**]<*g1,g2...*> }

tncconsole add -e *emailid* [-E FAIL | COMPLIANT | ALL] [**ipgroup=** [**±**]<*g1,g2...*>]

tncconsole add -I ip= [**±**]<*host1, host2...*>

tncconsole delete { **-F** <*FSPolicyname*> | **-G** <*ipgroupname*> | **-P** <*policyname*> | **-A** <*apargrp*> | **-V** <*ifixgrp*>}

tncconsole delete -H -i <*host* | **ALL**> **-D** <*yyyy-mm-dd*>

tncconsole certadd -i <*host*> **-t** <**TRUSTED** | **UNTRUSTED**>

tncconsole certdel -i <*host*>

tncconsole verify -i <*host*> | **-G** <*ipgroup*>

tncconsole update [-p] {-i< *host* >| **-G** <*ipgroup*> [**-r** <*buildinfo*> | **-a** <*apar1, apar2...*> | [**-u**] **-v** <*ifix1, ifix2,...*>}

tncconsole log loglevel=<info | **error** | **none**>

tncconsole import -C -i <*host*> **-f** <*filename*> | **–d** <*import database filename*>

tncconsole { **import -k** <*key_filename*> | **export**} **-S -f** <*filename*>

tncconsole list { **-S** | **-G** < *ipgroupname* | **ALL** > | **-F** < *FSPolicyname* | **ALL** > | **-P** < *policyname* | **ALL** > | **-r** < *buildinfo* | **ALL** > | **-I -i** < *ip* | **ALL** > | **-A** < *apargrp* | **ALL** > | **-V** <*ifixgrp*>} [**-c**] [**-q**]

tncconsole list { **-H** | **-s** <**COMPLIANT** | **IGNORE** | **FAILED** | **ALL**> } **-i** <*host* | **ALL**> [**-c**] [**-q**]

tncconsole export -d <*path to export directory*>

tncconsole report -v <CVEid|ALL> -o <TEXT|CSV>

tncconsole report -A <advisoryname>

tncconsole report -P <policyname|ALL> -o <TEXT|CSV>

tncconsole report -i <ip|ALL> -o <TEXT|CSV>

tncconsole report -B <buildinfo|ALL> -o <TEXT|CSV>

TNC client operations:

tncconsole mkclient [**tncport=**<*port*>] **tncserver=**<*host:port*>

tncconsole mkclient tncport=<*port*> -T

tncconsole { **rmclient** | **status** }

tncconsole {**start** | **stop** | **restart** } **client**

tncconsole chclient attribute = *value*

tncconsole list { **-C** | **-S** }

tncconsole export { **-C** | **-S** } **-f** <*filename*>

tncconsole import { **-S** | **-C -k** <*key_filename*> } **-f** <*filename*>

TNC IPRef operations:

tncconsole mkipref [**tncport=**<*port*>] **tncserver=**<host:port>

tncconsole { **rmipref** | **status**}

tncconsole { **start** | **stop** | **restart**} ipref

tncconsole chipref attribute = *value*

tncconsole { **import -k** <*key_filename*> | **export** } **-R -f** <*filename*>

tncconsole list -R

Description

The TNC technology is an open standard-based architecture for endpoint authentication, platform integrity measurement, and integrating security systems. The TNC architecture inspects endpoints (network clients and servers) for compliance with security policies before allowing them on the protected network. The TNC IPRef notifies the TNC server about any new IPs that are detected on the virtual I/O server (VIOS).

SUMA helps move system administrators away from the task of manually retrieving maintenance updates from the web. It offers flexible options that enable the system administrator to set up an automated interface to download fixes from a fix distribution website to their systems.

The **tncconsole** command manages the network server and clients by adding or deleting security policies, validating clients as trusted or untrusted, generating reports, and updating the server and the client.

The following operations can be performed by using the **tncconsole** command:

Item	Description
add	Adds a policy, a client, or the email information on the TNC server.
apargrp	Specifies the APAR group names as part of the fileset policy that are used for verification of TNC clients.
aparlist	Specifies the list of APARs that are part of the APAR group.
certadd	Marks the certificate as trusted or untrusted.
certdel	Deletes the client information.
chclient	Changes the attributes in the `tnccs.conf` file. An explicit **`start`** command is required for the changes to take effect in the TNC client. The syntax of attribute=value will be same as that of **mkclient**.
chipref	Changes the attributes in the `tnccs.conf` file. An explicit **`start`** command is required for the changes to take effect in IPRef. The syntax of attribute=value is the same as that of the **mkipref**.
chserver	Changes the attributes in the `tnccs.conf` file. An explicit **`start`** command is required for the changes to take effect in the TNC server. The syntax of attribute=value is same as that of **mkserver**. **Note:** The **dbpath** attribute cannot be changed by using the **chserver** command. It can be set only while running the **mkserver**.
dbpath	Specifies the TNC database location. The default value is `/var/tnc`.
delete	Deletes a policy or the client information.
export	Exports the client or server certificate , or database on TNC server.
fspolicy	Specifies the fileset policy of the release, technology level and service pack that are used for verification of TNC Clients.
import	Imports a certificate on client or server, or database on TNC server.
ipgroup	Specifies the Internet Protocol (IP) group that contains multiple client IP addresses or host names.
list	Displays information about the TNC server, the TNC client, or the SUMA.
log	Sets the log level for the TNC components.
mkclient	Configures the TNC client.
mkipref	Configures the TNC IPRef.
mkserver	Configures the TNC server.
pmport	Specifies the port number on which the **pmserver** listens to. The default value is 38240.
pmserver	Specifies the host name or IP address of the **suma** command that downloads the latest service packs and security fixes available in the IBM® ECC website and the IBM Fix Central website.

Item	Description
recheck_interval	Specifies the interval in minutes or d (days) : h (hours) : m (minutes) format for the TNC server to verify the TNC clients. **Note:** A value of **recheck_interval=0** means that the scheduler does not initiate verification of the clients at regular intervals and the registered clients are automatically verified during the startup. In such cases, the client can be manually verified.
report	Generates a report that has .txt or .csv file extension.
restart	Restarts the TNC client, the TNC server, or the TNC IPRef.
rmclient	Unconfigures the TNC client.
rmipref	Unconfigures the TNC IPRef.
rmserver	Unconfigures the TNC srever.
start	Starts the TNC client, the TNC server, or the TNC IPRef.
status	Shows the status of the TNC configuration.
stop	Stops the TNC client, the TNC server, or the TNC IPRef.
tncport	Specifies the port number on which the TNC server listens to. The default value is 42830.
tncserver	Specifies the TNC server that verifies or updates the TNC clients.
tssserver	Specifies the IP or host name of the TS server.
update	Installs patches on the client.
verify	Initiates a manual verification of the client.

Flags

Item	Description
-A <advisoryName>	Specifies the advisory name for the report.
-B <buildinfo>	Specifies the build information to prepare a patch report.
-i *host*	Specifies the IP address or host name.
-f *filename*	Specifies the file from which the certificate must be read in case of an import operation, or specifies the location to which the certificate must be written in case of an export operation.
-F *fspolicy buildinfo*	Specifies the file system policy name, followed by the build information. The build information can be provided in the following format: 6100-04-01, where 6100 represents version 6.1, 04 is the maintenance level, and 01 is the service pack.
-G *ipgroupname* **ip=**[**±**]*ip1, ip2...*	Specifies the IP group name followed by a comma-separated IP list.
-P *policyname* **fspolicy=**[**±**]*fspolicy1, fspolicy2...* **ipgroup=**[**±**]*g1, g2...*	Specifies the policy name followed by a comma-separated file system policy name list and an IP group name list. File system policies and IP groups can be added or removed from the file system policy name list and IP group name list by using + or - symbols, respectively.

Item	Description
-I ip=[**±**]*ip1, ip2...* | [±] host1,host2...	Specifies the IP/host name that must be ignored during verification.
-e *emailid* **ipgroup=**[**±**]*g1, g2...*	Specifies the email ID followed by a comma separated IP group name list.
-E | **FAIL** | **COMPLIANT** | **ALL** |	Specifies the event for which the emails need to be sent to the configured email id. FAIL- Mails are sent when the verification status of the client is FAILED. COMPLIANT- Mails are sent when the verification status of the client is COMPLAINT. ALL - Mails are sent for all the statuses of the client verification.
-d *database file location/dir path of database*	Specifies the file path location for import of the database/specifies the directory path location for export of the database.
-t TRUSTED | **UNTRUSTED**	Marks the specified client as trusted or untrusted. **Note:** Only system administrators can verify the server or client as trusted or untrusted.
-c	Displays the user attributes in colon-separated records as follows: `# name: attribute1: attribute2: ...` `policy: value1: value2: ...`
-p	Previews the TNC client update.
-q	Suppresses the header information.
-s COMPLIANT | **IGNORE** | **FAILED** | **ALL**	Displays the client by status as follows: **COMPLIANT** Displays the active clients. **IGNORE** Displays the clients that are excluded from any verification. **FAILED** Displays the clients that have failed verification as per the configured policy. **ALL** Displays all the clients irrespective of their statuses.
-u	Uninstalls an interim fix that is installed on a TNC client.
-r *buildinfo*	Generates the report based on the build information. The build information can be provided in the following format: 6100-04-01, where 6100 represents version 6.1, 04 is the maintenance level, and 01 is the service pack.
-H	Lists the history log.
-C	Specifies that the operation is for client component.
-S	Specifies that the operation is for server component.
-T	Specifies that the client can accept request from any TS server that has a valid certificate.
-v	Specifies a comma-separated interim fix list.

Item	Description
-V	Specifies the interim fix group name.
-R	Specifies that the operation is for IPRef component.
-k *filename*	Specifies the file from which the certificate key must be read in case of an import operation.
-D *yyyy-mm-dd*	Specifies the date for a particular client entry in the log history, where *yyyy* is the year, *mm* in the month, and *dd* is the day.
-P <policyName>	Specifies the policy name to prepare a client policy report.
-S <host>	Specifies the host name to prepare a client security fix report.

Exit Status

This command returns the following exit values:

Item	Description
0	The command ran successfully, and all the requested changes are made.
>0	An error occurred. The printed error message includes more details about the type of failure.

Examples

1. To start the TNC server, enter the following command:

   ```
   tncconsole start server
   ```

2. To add a file system policy named `71D_latest` for the build `7100-04-02`, enter the following command:

   ```
   tncconsole add -F 71D_latest 7100-04-02
   ```

3. To delete a file system policy named `71D_old`, enter the following command:

   ```
   tncconsole delete -F 71D_old
   ```

4. To validate that the client that has an IP address of `11.11.11.11` is **trusted**, enter the following command:

   ```
   tncconsole certadd -i 11.11.11.11 -t TRUSTED
   ```

5. To delete the client that has an IP address of `11.11.11.11` from the server, enter the following command:

   ```
   tncconsole certdel -i 11.11.11.11
   ```

6. To verify the client information that has an IP address of `11.11.11.11`, enter the following command:

   ```
   tncconsole verify -i 11.11.11.11
   ```

7. To display the client information that has an IP address of `11.11.11.11`, enter the following command:

   ```
   tncconsole list -i 11.11.11.11
   ```

8. To generate the report for clients that are in **COMPLAINT** status, enter the following command:

   ```
   tncconsole list -s CPMPLIANT -i ALL
   ```

9. To generate the report for the build 7100-04-02, enter the following command:

```
tncconsole list -r 7100-04-02
```

10. To display the connection history of a client that has an IP address of 11.11.11.11, enter the following command:

```
tncconsole list -H -i 11.11.11.11
```

11. To delete the entry of a client that has an IP address of 11.11.11.11 from the log history older or equal to 1 February, 2009, enter the following command:

```
tncconsole delete -H -i 11.11.11.11 -D 2009-02-01
```

12. To import the client certificate of a client that has an IP address of 11.11.11.11 from the server, enter the following command:

```
tncconsole import -C -i 11.11.11.11 -f /tmp/client.txt
```

13. To export the server certificate from a client, enter the following command:

```
tncconsole export -S -f /tmp/server.txt
```

14. To update the client that has an IP address of 11.11.11.11 to an appropriate level from the server, enter the following command:

```
tncconsole update -i 11.11.11.11
```

15. To display the client statuses, enter the following command:

```
tncconsole status
```

16. To display the client certificate, enter the following command:

```
tncconsole list -C
```

17. To start the client, enter the following command:

```
tncconsole start client
```

Security

Attention RBAC users and Trusted AIX users:

This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in Security. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand

tninit Command

Purpose

Initializes the Trusted Network subsystem and maintains the Trusted Network rules database.

Syntax

tninit [**-v m**] **init** [*filename*]

tninit [**-v m**] **load***filename*

tninit [**-v m**] **save***filename*

tninit [**-v m**] **disp***filename*

Description

The **tninit** command initializes the Trusted Network subsystem and maintains the Trusted Network rules database, including the **/etc/security/rules.host** and the **/etc/security/rules.int** files that are loaded upon system startup.

Flags

Item	Description
-v	Specifies verbose mode.
-m	Maintains the existing host rules when loading a new database.
init [*filename*]	Initializes the Trusted Network subsystem. This parameter loads tables into the kernel that are responsible for making the translation between a local representation of an Sensitivity Label (SL) and what is transmitted over the network. Optionally, you can specify the name of a file containing the mappings with the *filename* parameter. If you do not specify a file, a set of hard coded mappings is used. You can see an example of the mapping in the **/usr/samples/tn/rfc1108.example** file.
load *filename*	Loads a rules database into the kernel. Use the *filename* parameter to specify the file name. The command appends the `.host` and `.int` extensions to get the two files that comprise the database.
save *filename*	Saves the rules that are active in the kernel into the two files of the database. Uses the *filename* parameter to specify the file name. The `.host` and `.int` extensions are appended to the file name to get the two files that comprise the database.
disp *filename*	Displays the database that is specified for standard output (STDOUT). Use the *filename* parameter to specify the file name. The command appends the `.host` and `.int` extensions to get the two files that comprise the database.

Parameters

Item	Description
filename	Specifies the file name. Do not use `init`, `load`, `save`, or `disp` as file name.

Authorization

A user must have the **aix.mls.network.init** authorization to run the **tninit** command.

Examples

To initialize the Trusted Network subsystem, enter the following command:

```
tninint init
```

To load a rules database into the kernel, enter the following command:

```
tninit load /etc/security/rules
```

To save the rules active in the kernel into the two files of the database, enter the following command:

```
tninit save /etc/security/rules
```

To display the rules database specified into STDOUT, enter the following command:

```
tninit disp /etc/security/rules
```

tokstat Command

Purpose

Shows token-ring device driver and device statistics.

Syntax

tokstat [**-d** **-r** **-t**] *Device_Name*

Description

The **tokstat** command displays the statistics gathered by the specified Token-Ring device driver. The user can optionally specify that the device-specific statistics be displayed in addition to the device driver statistics. If no flags are specified, only the device driver statistics are displayed.

This command is also invoked when the **netstat** command is run with the **-v** flag. The **netstat** command does not issue any **tokstat** command flags.

If an invalid *Device_Name* is specified, the **tokstat** command produces an error message stating that it could not connect to the device.

Flags

Item	Description
-d	Displays all the device driver statistics, including the device-specific statistics.
-r	Resets all the statistics back to their initial values. This flag can only be issued by privileged users.
-t	Toggles debug trace in some device drivers.

Parameters

Item	Description
Device_Name	The name of the Token-Ring device, for example, t**ok0**.

Statistic Fields

> **Note:** Some adapters may not support a specific statistic. The value of non-supported statistic fields is always 0.

The statistic fields displayed in the output of the **toktstat** command and their descriptions are:

Title Fields

Item	Description
`Device Type`	Displays the description of the adapter type.
`Hardware Address`	Displays the Token-Ring network address currently used by the device.

Item	Description
Elapsed Time	Displays the real time period which has elapsed since the last time the statistics were reset. Part of the statistics may be reset by the device driver during error recovery when a hardware error is detected. There will be another Elapsed Time displayed in the middle of the output when this situation has occurred in order to reflect the time differences between the statistics.

Transmit Statistics Fields

Item	Description
Packets	The number of packets transmitted successfully by the device.
Bytes	The number of bytes transmitted successfully by the device.
Interrupts	The number of transmit interrupts received by the driver from the adapter.
Transmit Errors	The number of output errors encountered on this device. This is a counter for unsuccessful transmissions due to hardware/network errors.
Packets Dropped	The number of packets accepted by the device driver for transmission which were not (for any reason) given to the device.
Max Packets on S/W Transmit Queue	The maximum number of outgoing packets ever queued to the software transmit queue.
S/W Transmit Queue Overflow	The number of outgoing packets which have overflowed the software transmit queue.
Current S/W+H/W Transmit Queue Length	The number of pending outgoing packets on either the software transmit queue or the hardware transmit queue.
Broadcast Packets	The number of broadcast packets transmitted without any error.
Multicast Packets	The number of multicast packets transmitted without any error.
Timeout Errors	The number of unsuccessful transmissions due to adapter reported timeout errors.
Current SW Transmit Queue Length	The number of outgoing packets currently on the software transmit queue.
Current HW Transmit Queue Length	The number of outgoing packets currently on the hardware transmit queue.

Receive Statistics Fields

Item	Description
Packets	The number of packets received successfully by the device.
Bytes	The number of bytes received successfully by the device.
Interrupts	The number of receive interrupts received by the driver from the adapter.
Receive Errors	The number of input errors encountered on this device. This is a counter for unsuccessful reception due to hardware/ network errors.
Packets Dropped	The number of packets received by the device driver from this device which were not (for any reason) given to a network demuxer.
Bad Packets	The number of bad packets received (saved) by the device driver.
Broadcast Packets	The number of broadcast packets received without error.
Multicast Packets	The number of multicast packets received without error.
Receive Congestion Errors	The number of incoming packets dropped by the hardware due to a no resource error.

General Statistics Fields

Item	Description
No mbuf Errors	The number of times mbufs were not available to the device driver. This usually occurs during receive operations when the driver must obtain mbuf buffers to process inbound packets. If the mbuf pool for the requested size is empty, the packet will be discarded. The **netstat -m** command can be used to confirm this.
Lobe Wire Faults	The number of times the adapter detected an open or short circuit in the lobe data path (for example, the cable is unplugged).
Abort Errors	The number of times the adapter had problems transmitting.
AC Errors	The number of times the adapter received more than one AMP (Active Monitor Present) or SMP (Standby Monitor Present) frame which had the address recognized and frame copied bits set to zero. This indicates a problem with neighbor notification. Every station learns and remembers who its Nearest Active Upstream Neighbor (NAUN) is from AMP and SMP frames. When a station reports a problem, it also reports who its NAUN is. This helps to define the *fault domain*.
Burst Errors	The number of times the adapter detected that the polarity of the signal did not switch when necessary.
Frame Copy Errors	The number of times the adapter detected that a frame with its specific address has been copied by another adapter.
Frequency Errors	The number of times the adapter detected that the frequency of the incoming signal differs from the expected frequency by more than that allowed by the IEEE 802.5 standard. Check the active monitor responsible for master clocking of the ring and compensating for frequency jitter.

Item	Description
Hard Errors	The number of times the adapter either transmitted or received a beacon MAC frame.
Internal Errors	The number of times the adapter had an internal error.
Line Errors	The number of times the adapter detected an invalid character in a frame or token.
Lost Frame Errors	The number of times the adapter transmitted a frame and failed to receive it back.
Only Station	The number of times the adapter sensed that it is the only adapter on the ring.
Token Errors	The number of times the adapter, acting as an active monitor, detected that the token got lost. This may be due to ring reconfiguration. If this occurs often, check to see if other soft errors indicate a specific problem.
Remove Received	The number of times the adapter received a Remove Ring Station MAC frame request.
Ring Recovered	The number of times the ring is purged and recovered back into a normal operating state.
Signal Loss Errors	The number of times the adapter detected the absence of a receive signal.
Soft Errors	The number of times the adapter detected a soft error (recoverable by the MAC layer protocols).
Transmit Beacon Errors	The number of times the adapter transmitted a beacon frame.
Driver Flags	The device driver internal status flags currently turned on.

Device Specific Statistics Fields

This part of the display may be different for each type of adapter. It may contain adapter-specific information and some extended statistics that were not included in the generic statistics. Some adapters may not have any device-specific statistics. Some fields that may be listed in this section are:

Item	Description
ARI/FCI Errors	ARI/FCI mismatch is also referred to as receiver congestion. If an adapter gets an address match on a frame going by on the ring, Address Recognized Indication(ARI), and has no place into which to copy the frame, Frame Copied Indication(FCI), an ARI/FCI mismatch has occurred. The adapter will turn on the ARI bits but will not turn on the FCI bits in the FS byte at the end of the frame as it goes by. In other words, the adapter saw a frame that was to be received but, could not receive it because the receive buffers have been depleted. Two seconds later the adapter will send a Report Soft Error MAC frame indicating a receiver congestion error.
DMA Bus Errors	The number of times the adapter completed a DMA transfer and detected a bus error.

Item	Description
DMA Parity Errors	The number of times the adapter completed a DMA transfer and detected a parity error.
Receive Overruns	The number of times the adapter receive FIFO was full when the adapter tried to receive a frame.
Receive Underruns	The number of times the adapter transmit FIFO was empty before the end of frame symbol was detected.
Number of read log commands issued	The number of times an adapter error counter overruns (reached 255) and the device driver issues a read log command to read (and reset) the error counters.

Examples

1. To display the device driver statistics for **tok0**, enter:

```
tokstat tok0
```

This produces the following output:

```
TOKEN-RING STATISTICS (tok0) :
Device Type: Token-Ring High-Performance Adapter (8fc8)
Hardware Address: 10:00:5a:4f:26:c1
Elapsed Time: 0 days 0 hours 8 minutes 33 seconds
Transmit Statistics:                  Receive Statistics:
--------------------                  -------------------
Packets: 191                          Packets: 8342
Bytes: 17081                          Bytes: 763227
Interrupts: 156                       Interrupts: 8159
Transmit Errors: 0                    Receive Errors: 0
Packets Dropped: 0                    Packets Dropped: 0
Max Packets on S/W Transmit Queue: 17 Bad Packets: 0
S/W Transmit Queue Overflow: 0
Current S/W+H/W Transmit Queue Length: 0
```

```
Broadcast Packets: 1                Broadcast Packets: 8023
Multicast Packets: 0                Multicast Packets: 0
Timeout Errors: 0                   Receive Congestion Errors: 0
Current SW Transmit Queue Length: 0
Current HW Transmit Queue Length: 0
```

```
General Statistics:
-------------------
No mbuf Errors: 0                   Lobe Wire Faults: 0
Abort Errors: 0                     AC Errors: 0
Burst Errors: 0                     Frame Copy Errors: 0
Frequency Errors: 0                 Hard Errors: 0
Internal Errors: 0                  Line Errors: 0
Lost Frame Errors: 0                Only Station: 0
Token Errors: 0                     Remove Received: 0
Ring Recovered: 0                   Signal Loss Errors: 0
Soft Errors: 0                      Transmit Beacon Errors: 0
Driver Flags: Up Broadcast Running
   AlternateAddress ReceiveFunctionalAddr
```

2. To display the token-ring device driver statistics and the Token-Ring device-specific statistics for **tok0**, enter:

```
tokstat -d tok0
```

This produces the following output:

```
TOKEN-RING STATISTICS (tok0) :
Device Type: Token-Ring High-Performance Adapter (8fc8)
Hardware Address: 10:00:5a:4f:26:c1
Elapsed Time: 0 days 2 hours 48 minutes 38 seconds
```

```
Transmit Statistics:                 Receive Statistics:
--------------------                 -------------------
Packets: 389                         Packets: 153216
Bytes: 42270                         Bytes: 14583150
Interrupts: 354                      Interrupts: 151025
Transmit Errors: 0                   Receive Errors: 0
Packets Dropped: 0                   Packets Dropped: 0
Max Packets on S/W Transmit Queue:17 Bad Packets: 0
S/W Transmit Queue Overflow: 0
Current S/W+H/W Transmit Queue Length: 0
```

```
Broadcast Packets: 1                 Broadcast Packets: 152642
Multicast Packets: 0                 Multicast Packets: 0
Timeout Errors: 0                    Receive Congestion Errors: 0
Current SW Transmit Queue Length: 0
Current HW Transmit Queue Length: 0
```

```
General Statistics:
-------------------
No mbuf Errors: 0                    Lobe Wire Faults: 0
Abort Errors: 0                      AC Errors: 0
Burst Errors: 0                      Frame Copy Errors: 0
Frequency Errors: 0                  Hard Errors: 0
Internal Errors: 0                   Line Errors: 0
Lost Frame Errors: 0                 Only Station: 0
Token Errors: 0                      Remove Received: 0
Ring Recovered: 0                    Signal Loss Errors: 0
Soft Errors: 0                       Transmit Beacon Errors: 0
Driver Flags: Up Broadcast Running
   AlternateAddress ReceiveFunctionalAddr
```

```
Token-Ring High-Performance Adapter (8fc8) Specific Statistics:
------------------------------------------------------------
DMA Bus Errors: 0                       DMA Parity Errors: 0
ARI/FCI Errors: 0
```

topas Command

Purpose

Reports selected local and remote system statistics.

Syntax

topas [**-d** *hotdisk*][**-f** *hotfs*] [**-h**] [**-i** *interval*] [**-n** *hotni*] [**-p** *hotprocess*] [**-w** *hotwlmclass*] [**-c** *hotprocessor*][`-I` *remotepollinterval*][`-@` [*wparname*]] [**-U** *username*] | [`-C` `-D` | `-G` | `-F` | **-L** | **-P** | **-V** | **-T** | **-M** | **-t** | **-E** | **-W**] [**-m**]

Restriction: You cannot use the **–C**, **-L**, **-E**, **-V**, **-T**, **-t**, **-w**, **-W**, **-I**, **-@** options when you issue the command from a workload partition.

Description

The **topas** command reports selected statistics about the activity on the local system. The command uses the curses library to display its output in a format suitable for viewing on an 80x25 character-based display or in a window of at least the same size on a graphical display. The **topas** command requires the `bos.perf.tools` and **perfagent.tools** file sets to be installed on the system.

The `topas` command can also report a limited set of performance metrics from remote AIX partitions that belong to the same hardware platform. This support is described in the Cross-Partition View and Cluster Utilization View sections.

Note: For any dynamic configuration changes to the system, the tool must be restarted to reflect the new changes.

The **topas -D** command reports the disk details. This report is described in the Disk Panel section. You can run the subcommands from the Disk panel to display the following views:

Adapter Panel
Specified by pressing the **d** key. This panel provides details on the adapters and the disks that belong to the selected adapters.

Virtual Adapter Panel
Specified by pressing the **d** key and then the **v** key. This panel provides details of the virtual adapters that are related to the disks.

MPIO Panel
Specified by pressing the **m** key. This panel provides the details of the disks and the paths.

Panel Freezing
Specified by pressing the **space bar** key on the keyboard. The **space bar** key acts as a toggle for freezing the `topas` panel.

Scrolling
The `Page Up` and `Page Down` keys are used to scroll through the data.

Restriction: Adapter panel, Virtual Adapter panel, and MPIO panel are restricted inside WPAR.

If the **topas** command is invoked without flags, it runs as if invoked with the following command:

```
topas -d20 -i2 -n20 -p20 -w20 -c20 -f0
```

Note: The Central Electronic Complex (CEC) or cluster panel re-spawns when the migration or hibernation of the partition is complete. All other behavior for the CEC and any other panel remains the same in the event of migration or hibernation.

The program extracts statistics from the system with an interval specified by the *monitoring_interval_in_seconds* argument. The default output, as shown below, consists of two fixed parts and a variable section. The top two lines at the left of the display show the name of the system the **topas** command runs on, the date and time of the last observation, and the monitoring interval.

The second fixed part fills the rightmost 25 positions of the display. It contains the following subsections of statistics:

Item	Description
EVENTS/QUEUES	Displays the per-second frequency of selected system-global events and the average size of the thread run and wait queues: **Cswitch** The number of context switches per second over the monitoring interval. **Syscalls** The total number of system calls per second that are run over the monitoring interval. **Reads** The number of read system calls per second that are run over the monitoring interval. **Writes** The number of write system calls per second that are run over the monitoring interval. **Forks** The number of fork system calls per second that are run over the monitoring interval. **Execs** The number of exec system calls per second that are run over the monitoring interval. **Runqueue** The average number of threads that were ready to run but were waiting for a processor to become available. **Waitqueue** The average number of threads that were waiting for paging to complete.
FILE/TTY	Displays the per-second frequency of selected file and the **TTY** statistics. The following data is reported: **Readch** The amount of bytes read per second through the **read** system call over the monitoring interval. **Writech** The amount of bytes written per second through the **write** system call over the monitoring interval. **Rawin** The amount of raw bytes read per second from TTYs over the monitoring interval. **Ttyout** The amount of bytes written to TTYs per second over the monitoring interval. **Igets** The number of calls per second to the inode lookup routines over the monitoring interval. **Namei** The number of calls per second to the path name lookup routines over the monitoring interval. **Dirblk** The number of directory blocks scanned per second by the directory search routine over the monitoring interval.

Item	Description
PAGING	Displays the per-second frequency of paging statistics. The following data is reported: **Faults** The total number of page faults taken per second over the monitoring interval. This includes page faults that do not cause paging activity. **Steals** The physical memory 4 K frames stolen per second by the virtual memory manager over the monitoring interval. **PgspIn** The number of 4 K pages read from paging space per second over the monitoring interval. **PgspOut** The number of 4 K pages written to paging space per second over the monitoring interval. **PageIn** The number of 4 K pages read per second over the monitoring interval. This includes paging activity associated with reading from file systems. Subtract **PgspIn** from this value to get the number of 4K pages read from file systems per second over the monitoring interval. **PageOut** The number of 4 K pages written per second over the monitoring interval. This includes paging activity associated with writing to file systems. Subtract **PgspOut** from this value to get the number of 4K pages written to file systems per second over the monitoring interval. **Sios** The number of I/O requests per second issued by the virtual memory manager over the monitoring interval.
MEMORY	Displays the real memory size and the distribution of memory in use. The following data is reported: **Real,MB** The size of real memory in megabytes. **% Comp** The percentage of real memory currently allocated to computational page frames. Computational page frames are generally those that are backed by paging space. **% Noncomp** The percentage of real memory currently allocated to non-computational frames. Non-computational page frames are generally those that are backed by file space, either data files, executable files, or shared library files. **% Client** The percentage of real memory currently allocated to cache remotely mounted files.
PAGING SPACE	Displays the size and use of paging space. The following data is reported: **Size,MB** The sum of all paging spaces on the system, in megabytes. **% Used** The percentage of total paging space currently in use. **% Free** The percentage of total paging space currently free.

Item	Description
NFS	Displays the NFS statistics in calls per second. The following data is reported: • Server V2 calls/sec • Client V2 calls/sec • Server V3 calls/sec • Client V3 calls/sec
`Total WPAR`	Displays the total number of workload partitions that are defined in the system. The total amount of workload partitions can be in the following states: **Defined**, **Active**, **Broken** or **Transition**.
`Active WPAR`	Displays the total number of resource active workload partitions.
AME	Displays memory compression statistics in an Active Memory Expansion enabled system. The following data is reported: **TMEM,MB** True Memory Size, in megabytes. **CMEM,MB** Compressed Pool Size, in megabytes. **EF[T/A]** Expansion Factors: Target & Actual. **CI** Compressed Pool Page-Ins. **CO** Compressed Pool Page-Outs.

The variable part of the **topas** display can have one, two, three, four, or five subsections. If more than one subsection displays, they are always shown in the following order:

- Processor utilization
- Network interfaces
- Physical disks
- File system
- Workload Manager classes
- workload partitions
- Processes

When the **topas** command is started, it displays all subsections for which hot entities are monitored. The Workload Manager (WLM) Classes subsection is displayed only when WLM is active.

The WLM should be started to view the WLM and WPAR statistics.

Tip: When there is no WPAR specific information for a metric, the system-wide value is displayed for that metric in inverted background (that is, white text and black context).

The following table provides the details for the subsections that the **topas** command displays:

Item	Description
Processor utilization	This subsection displays one-line report summary of all the processor usage. Pressing the **c** key only once turns this subsection off. If more than one processor exists, a list of processors is displayed by pressing the **c** key twice. Pressing the **c** key thrice displays a bar chart showing cumulative processor usage. The following fields are displayed by both formats: **User%** The percentage of processor used by programs running in user mode. (Default sorted by User%) **Kern%** The percentage of processor used by programs running in kernel mode. **Wait%** The percentage of time spent in waiting for I/O. **Idle%** The percentage of time that the processors are idle. **Physc** The number of physical processors that are consumed. Displayed only if the partition is running with shared processor. **%Entc** The percentage of entitled capacity that is consumed. Displayed only if the partition is running with shared processor. When this subsection displays the list of hot processors, the list is sorted by the **User%** field. However, the list can be sorted by the other fields by moving the cursor to the top of the desired column.
Network interfaces	This subsection shows a one-line report summary of the activity for all network interfaces. Pressing the **n** key once turns off this subsection. Pressing the **n** key twice displays a list of active network interfaces. The maximum number of interfaces displayed is the number of active interfaces being monitored, as specified by using the **-n** flag. A smaller number of interfaces are displayed if other subsections are also being displayed. Both reports display the following fields: **BPS** The total throughput in kilobytes per second over the monitoring interval. This field is the sum of kilobytes received and kilobytes sent per second. **Interf** The name of the network interface. **I-Pack** The amount of data packets received per second over the monitoring interval. **KB-In** The number of kilobytes received per second over the monitoring interval. **KB-Out** The number of kilobytes sent per second over the monitoring interval. **O-Pack** The amount of data packets sent per second over the monitoring interval. When this subsection displays the list of hot network interfaces, the list is sorted by the **BPS** field. However, the list can be sorted by the other fields by moving the cursor to the top of the desired column. Sorting is only valid for up to 16 network adapters.

Item	Description
Physical disks	This subsection shows a one-line report summary of the activity for all physical disks. Pressing the **d** key turns once off this subsection. Pressing the **d** key again displays a list of active physical disks. The maximum number of physical disks displayed is the number of active physical disks being monitored, as specified by using the **-d** flag. A smaller number of physical disks is displayed if other subsections are also being displayed. Both reports display the following fields: **Busy%** The percentage of time the physical disk is active (bandwidth use of the drive). **BPS** The amount of data transferred (read and written) in kilobytes per second over the monitoring interval. This field is the sum of the values of the **KB-Read** and **KB-Writ**. **Disk** The name of the physical disk. **KB-Read** The number of kilobytes read per second from the physical disk. **KB-Writ** The number of kilobytes written per second to the physical disk. **TPS** The number of transfers per second that were issued to the physical disk. A transfer is an I/O request to the physical disk. Multiple logical requests can be combined into a single I/O request to the disk. A transfer is of indeterminate size. When this subsection displays the list of hot physical disks, the list is sorted by the **BPS** field. However, the list can be sorted by the other fields by moving the cursor to the top of the desired column. Sorting is only valid for up to 128 physical disks.

Item	Description
File system	This subsection shows a one-line report summary of the activity for all of the file systems. Pressing the **f** key once turns off this section. Pressing the **f** key twice displays a list of active file systems. The maximum number of file systems that are displayed is the number of active file systems that are monitored when they are specified by using the **-f** flag. A smaller number of file systems are displayed if other subsections are also being displayed. Both reports display the following fields: **BPS** The amount of data transferred (read and written) in kilobytes per second over the monitoring interval. This field is the sum of the values of the **KB-Read** and **KB-Writ** fields. **File System** The name of the file system. **KB-Read** The number of kilobytes read per second from the file system. **KB-Writ** The number of kilobytes written per second to the file system. **TPS** The number of transfers per second that are issued to the file system. A transfer is an I/O request to the file system. Multiple logical requests can be combined into a single I/O request to the file system. The size of a transfer is not determinate. When this subsection displays the list of the file systems, the list is sorted by the **BPS** field. However, the list can be sorted by the other fields by moving the cursor to the top of the target column. **Tip:** If the file system name exceeds the field width in the display, then the file system name is displayed in a truncated format. The truncation contains the first and last few characters of the file system, the middle part of the name is replaced by periods (..). For example, if the file system name is `filesystem001234`, then the name is displayed as `files..01234`.

Item	Description
WLM classes	This subsection displays a list of hot Workload Manager (WLM) Classes. The maximum number of WLM classes displayed is the number of hot WLM classes being monitored as specified with the **-w** flag. A smaller number of classes will be displayed if other subsections are also being displayed. Pressing the **w** key turns off this subsection. The following fields are displayed for each class: **% processor Utilization** The average processor use of the WLM class over the monitoring interval. **% Mem Utilization** The average memory use of the WLM class over the monitoring interval. **% Blk I/O** The average percent of block I/O of the WLM class over the monitoring interval. When this subsection first displays the list of hot WLM classes, the list will be sorted by the **CPU%** field. However, the list can be sorted by the other fields by moving the cursor to the top of the desired column. **Tip:** If the WLM class name exceeds the field width in the display, the WLM class name is truncated. The truncation contains the first and last few characters of the WLM class, and the middle part of the name is replaced by periods (..). For example, if the WLM class name is `unclassified00123`, then the WLM class name is displayed as `uncla..00123`.
`Workload partitions`	The workload partitions subsection replaces WLM subsection if invoked with the **-@** flag. This subsection displays a list of hot workload partitions. The maximum number of workload partitions that are displayed is the number of hot WPAR that are monitored (when they are specified with the **-w -@** flag). A smaller number of WPAR is displayed if other subsections are also being displayed. To turn off the workload partitions subsection, press the **@** key. The following fields are displayed for each WPAR: **WPAR** The name of the workload partition (WPAR). **% processor Utilization** The average processor use of the WPAR over the monitoring interval. **% Mem Utilization** The average memory use of the WPAR over the monitoring interval. **% Blk I/O** The average percent of block I/O of the WPAR over the monitoring interval. When this subsection displays the list of hot WPAR, the list is sorted by the **CPU%** field. However, the list can be sorted by the other fields by moving the cursor to the top of the target column that you want to use to sort the list. **Tip:** If the WPAR name exceeds the field width in the display, the WPAR name is truncated. The truncation contains the first and last few characters of the WPAR, and the middle part of the name is replaced by periods (..). For example, if the WPAR name is `neptune00123`, then the WPAR is displayed as `neptu..00123`.

Item	Description
Processes	This subsection displays a list of hot processes. The maximum number of processes displayed is the number of hot processes being monitored as specified with the **-p** flag. A smaller number of processes will be displayed if other subsections are also being displayed. Pressing the **p** key turns off this subsection. The processes are sorted by their processor usage over the monitoring interval. The following fields are displayed for each process: **Name** The name of the executable program executing in the process. The name is stripped of any pathname and argument information and truncated to 9 characters in length. **Process ID** The process ID of the process. **% CPU Utilization** The average processor use of the process over the monitoring interval. The first time a process is shown, this value is the average processor use over the lifetime of the process. **Paging Space Used** The size of the paging space allocated to this process. This can be considered an expression of the footprint of the process but does not include the memory used to keep the executable program and any shared libraries it may depend on. **Process Owner (if the WLM section is off)** The user name of the user who owns the process. **Workload Manager (WLM) Class (if the WLM section is on)** The WLM class to which the process belongs. **WPAR (if the WPAR section is on)** The WPAR name that the process belongs to. **Tip:** If the WLM Class/WPAR name exceeds the field width in the display, the WLM Class/WPAR name is truncated. The truncation contains the first and last few characters of the WLM Class/WPAR, and the middle part of the name is replaced by periods (..). For example, if the WLM Class/WPAR name is `unclassified00123`, then the WLM Class/WPAR name is displayed as `uncla..00123`.

Adapter Panel View

When you use the **topas -D** command, you can press the **d** key to display the Adapter panel view. In this panel, the following metrics are displayed:

Item	Description
Adapter	The name of the adapter.
KBPS	The amount of data transferred (read or written) in the adapter in kilobytes per second.
TPS	Indicates the average number of transfers per second that the adapter issues.
KB-R	The total number of kilobytes that are read from the adapter.
KB-W	The total number of kilobytes that are written to the adapter.

If you press the **f** key, the following details of the disks that belong to the adapter are displayed on the Adapter panel:

Item	Description
AQD	The average number of requests that are waiting to be sent to the virtual target device or disk.
AQW	The average time spent by a transfer request in the waiting queue. The suffix indicates the unit of time. The default unit of time is millisecond.
ART	The average time to receive a response from the hosting for the read request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
AWT	The average time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
Busy%	The percentage of time the virtual target device or disk is active (bandwidth use of the virtual target device or disk).
KBPS	The amount of data that is read and written in kilobytes per second over the monitoring interval. This field is the sum of the values of the **KB-R** and **KB-W** metrics.
KB-R	The number of kilobytes per second that are read from the virtual target device or disk.
KB-W	The number of kilobytes per second that are written to the virtual target device or disk.
MRT	The maximum time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
MWT	The maximum time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
TPS	The number of transfers per second that are issued to the virtual target device or disk. A transfer is an I/O request to the virtual target device or disk. Multiple logical requests can be combined into a single I/O request to the disk. A transfer is of medium size.
Vtargets/Disk	The name of the virtual target device or disk.

Virtual Adapter Panel View

When you run the **topas -D** command, you can press the **v** key to display the Virtual Adapter panel view. In this panel, the following metrics are displayed:

Item	Description
AQD	The average number of requests waiting to be sent to the adapter.
AQW	The average time spent by a transfer request in the waiting queue. The suffix indicates the unit of time. The default unit of time is millisecond.
ART	The average time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
AWT	The average time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
KBPS	The amount of data transferred (read or written) in kilobytes per second in the adapter.
KB-R	The number of blocks received per second from the hosting server to the adapter.
KB-W	The number of blocks sent per second from this adapter to the hosting server.

Item	Description
MRT	The maximum time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
MWT	The maximum time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
TPS	The number of transfers per second that are issued to the adapter.
vAdapter	The name of the virtual adapter.

If you press the **f** key, the following details of the disks that belong to the adapter are displayed on the Virtual Adapter panel:

Item	Description
AQD	The average number of requests that are waiting to be sent to the virtual target device or disk.
AQW	The average time spent by a transfer request in the waiting queue. The suffix indicates the unit of time. The default unit of time is millisecond.
ART	The average time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
AWT	The average time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
Busy%	The percentage of time the virtual target device or disk is active (bandwidth use of the virtual target device or disk).
KBPS	The amount of data that is read and written in kilobytes per second over the monitoring interval. This field is the sum of the values of the **KB-R** and **KB-W** metrics.
KB-R	The number of kilobytes that are read per second from the virtual target device or disk.
KB-W	The number of kilobytes that are written per second to the virtual target device or disk.
MRT	The maximum time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
MWT	The maximum time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
TPS	The number of transfers per second that are issued to the virtual target device or disk. A transfer is an I/O request to the virtual target device or disk. Multiple logical requests can be combined into a single I/O request to the disk. A transfer is of medium size.
Vtargets/Disk	The name of the virtual target device or disk.

MPIO Panel View

When you use the **topas -D** command, you can press the **m** key to display the MPIO panel view. In this panel, the top section contains the same metrics that the Disks panel displays.

The bottom section of the panel contains the following fields:

Item	Description
Busy%	The percentage of time the path is active (bandwidth use of the path).
KBPS	The amount of data that is read and written in kilobytes per second over the monitoring interval. This field is the sum of the values of the **KB-R** and **KB-W** metrics.
KB-R	The number of kilobytes that is read per second in that path.
KB-W	The number of kilobytes that is written per second in that path.
Path	The name of the path.
TPS	The number of transfers per second that are issued in that path.

Panel Freezing

The `space bar` key on the keyboard acts as a toggle for freezing the `topas` panel. If frozen, `topas` stops data collection and continues to display the data from the previous iteration. You can move around the panel and sort the data based on the selected column. In frozen state, if you move between panels, some panels may not display the data. In this case, press the `space bar` key to unfreeze the `topas` panel.

Scrolling

If the amount of data is more than the `topas` window size, then `Page Up` and `Page Down` keys are used to scroll though the data. The data is sorted based on the selected column.

Note: The above functionality is available with selected panels in topas.

I/O Memory Entitlement Pools Panel

When a Logical Partition panel (**topas -L**) is enabled in shared-memory mode, you can press the **e** key to display the I/O Memory Entitlement Pools panel.

The following metrics are displayed in the lower section of this panel:

Item	Description
iompn	The name of the I/O memory pool.
iomin	The minimum I/O memory entitlement of the pool.
iodes	The desired I/O memory entitlement of the pool.
ioinu	The current I/O memory entitlement of the pool.
iores	The reserved I/O memory entitlement of the pool.
iohwm	The maximum I/O memory entitlement in use for the pool (high water mark).
ioafl	The total number of times the allocation requests have failed for this pool.

Cross-Partition View and Recording

This panel displays metrics similar to the `lparstat` command for all the AIX partitions it can identify as belonging to the same hardware platform. Dedicated and shared partitions are displayed in separate sections with appropriate metrics. The top section represents aggregated data from the partition set to show overall partition, memory, and processor activity.

Remote enablement for this panel to collect from other partitions requires to use the latest updates to the **perfagent.tools** and **bos.perf.tools** to support this function. For earlier versions of AIX, the **topas** command also collects remote data from partitions that have the Performance Aide product (**perfagent.server**) installed. The `topas` **-C** command may not be able to locate partitions residing on

other sub-nets. To avoid this, create a **$HOME/Rsi.hosts** file containing the fully qualified host names for each partition (including domains), one host per line.

Note: The **topas -C** command sends broadcast packet to all the Logical Partitions (LPARs) in the same subnet, but only processes response from the LPARs within the same CEC.

The following metrics display in the initial cross-partition panel. Additional metrics with full descriptive labels can be displayed by using the key toggles identified in the Additional cross-partition panel subcommands section:

Partition totals:

Item	Description
Shr	The number of shared partitions based on the system processor.
Ded	The number of dedicated partitions based on the system processor.

Memory (in GB):

Item	Description
Mon	The total memory of monitored partitions.
Avl	The memory available to partition set.
InUse	The memory in use on monitored partitions.

Processor:

Item	Description
Shr	The number of shared processors.
Ded	The number of dedicated processors.
PSz	The number of shared physical CPUs in the system.
APP	Indicates the available physical processors in the system (default shared processor pool). **Note:** Default shared processor pool contains the physical processors that are available on the managed system. The **topas** command retrieves the APP value from the data that is provided by the LPARs that are in the same managed system. If these LPARs do not belong to the default shared processor pool, the **topas** command cannot determine the APP value for the managed system. The APP value is indicated by the - (hyphen) character in this case.
Don	The total number of processors donated to the pool
Shr_PhysB	The total number of physical processors that are consumed by all shared partitions
Ded_PhysB	The total number of physical processors that are consumed by all dedicated partitions

Individual partition data:

Item	Description
Host	The host name
OS	The operating system level
Mod	The mode of the individual partitions. The mode is displayed in a set of 3 characters.

Item	Description
Character	The first character indicates the CPU in the partition. The second character indicates the memory mode of the partition. The third character indicates the energy state of the partition.
Mem	The total memory measured in gigabytes.
InU	The memory in use measured in gigabytes.
Lp	The number of logical processors.
Us	The percentage of processor used by programs executing in user mode.
Sy	The percentage of processor used by programs executing in kernel mode.
Wa	The percentage of time spent waiting for I/O.
Id	The percentage of time the processors are idle.
PhysB	The number of physical processors that are consumed by each partition.
Ent	The entitlement granted (shared-only).
%Entc	The percent entitlement consumed (shared-only).
Vcsw	The average of virtual context switches per second (shared-only).
PhI	The average of phantom interrupts per second (shared-only).
Pmem	The physical memory that is backing the partitions logical memory (if in shared-memory mode).
%idon	The percentage of physical processor that is used while explicitly donating idle cycles. This metric is applicable only for donating dedicated partitions.
%bdon	The percentage of physical processor that is used while busy cycles are being donated. This metric is applicable only for donating dedicated partitions.
%istl	The percentage of physical processor that is used while idle cycles are being stolen by the hypervisor. This metric is applicable only for dedicated partitions.
%bstl	The percentage of physical processor that is used while busy cycles are being stolen by the hypervisor. This metric is applicable only for dedicated partitions.

For shared partitions:

First Character	Description
C	SMT enabled and capped
c	SMT disabled and capped
U	SMT enabled and uncapped
u	SMT disabled and uncapped

For dedicated partitions:

First Character	Description
S	SMT enabled and not donating

First Character	Description
d	SMT disabled and donating
D	SMT enabled and donating
-	SMT disabled and not donating

Second Character	Description
M	AMS enabled and AME disabled
-	AME and AMS disabled
E	AME enabled and AMS enabled
e	AME enabled and AMS disabled

Third Character	Description
S	Static power save mode is enabled
d	Power save mode is disabled
D	Dynamic power save mode is enabled
-	Unknown / Undefined
E	Power save mode has been enabled
d	Power save mode has been disabled

The **%idon** and **%bdon** metrics are not displayed when there is no donating dedicated partition.

Requirement: At least one partition to be monitored must have Pool Utilization Authority (PUA) configured for pool information metrics to be collected.

For cross-partition monitoring/recording, some global data is not available from any partition. The **-o** option allows you to specify these fields in the command line. Optionally, you can configure a system to allow the `topas` command to query the HMC directly for this information. This requires the following steps:

1. Install OpenSSH at the partition.
2. Enable remote command support on the HMC for user **hscroot** to allow **ssh** connections to be opened from the partition.
3. Configure **ssh** on the HMC to not require a password for the HMC user **hscroot** when queried from the selected partition. This requires the **.ssh/authorized_keys2** on the HMC for user login **hscroot**.
4. Run **ssh -l hscroot** *hmc_address date* from the partition to confirm whether the date is displayed without requiring that a password be entered.
5. Utilize the **topas -o** options described in the usage table to specify the managed system and HMC names when running the `topas` command.

Restriction: This functionality is currently available only for HMC version 5 and above, and should only be enabled after careful consideration of any security implications.

The following displays when press the **g** key in the initial screen, which brings the cross partition view with detailed headers:

```
Topas CEC Monitor            Interval:  10             Mon Jan 22 00:08:00 2007
Partition Info    Memory (GB)        Processor         Virtual Pools :   2
Monitored  :  2   Monitored  : 6.2   Monitored  :2.0   Avail Pool Proc: 5
UnMonitored:  -   UnMonitored:   -   UnMonitored:  -   Shr Physical Busy:  0.00
Shared     :  0   Available  :   -   Available  :  -   Ded Physical Busy:  0.05
Uncapped   :  0   UnAllocated:   -   UnAllocated:  -   Donated Phys. processors: 0.00
Capped     :  2   Consumed   : 1.9   Shared     :  0   Stolen Phys. processors : 0.01
Dedicated  :  2                     Dedicated  :  2   Hypervisor
```

```
Donating   :  0                      Donated    :  0   Virt. Context Switch: 347
                                     Pool Size  :  0   Phantom Interrupts  :   0

Host       OS  M Mem InU Lp  Us Sy Wa Id  PhysB  Vcsw Ent  %EntC PhI
-----------------------------------shared-------------------------------------
ptoolsl1   A53 U 3.1 1.9  4   1  2  0 96   0.01  398  0.20   5.3   0k
Host       OS  M Mem InU Lp  Us Sy Wa Id  PhysB  Vcsw  %istl %bstl %bdon %idon
---------------------------------dedicated----------------------------------
ptools1    A54 S 3.1 0.9  2   0  0  0 99   0.00  177    0.1   0.0   0.0   0.0
ptoolsl3   A54 S 3.1 0.9  2   0  0  0 99   0.00  170    0.2   0.0   0.0   0.0
```

The following headers are in the previous screen:

Partition Info:

Item	Description
Monitored	The number of partitions that are monitored
Unmonitored	The number of partitions that are not monitored
Shared	The number of shared partitions
Uncapped	The number of uncapped shared partitions
Capped	The number of capped partitions
Dedicated	The number of dedicated partitions
Donating	The number of partitions that are currently donating

Memory:

Item	Description
Monitored	The total memory that is monitored
UnMonitored	The total memory that is not monitored
Available	The total memory that is available
UnAllocated	The total memory that is not allocated to any partition
Consumed	The total memory that is consumed by the partitions

Processor:

Item	Description
Monitored	The number of physical processors that are monitored
UnMonitored	The number of physical processors that are not monitored
Available	The number of physical processors that are available in CEC system
UnAllocated	The number of physical processors that are not allocated to any partition
Shared	The number of processors that are in shared partitions
Dedicated	The number of processors that are in dedicated partitions
Donated	The sum of the number of processors in all the partitions donating
Pool Size	The number of shared physical CPUs in the system.

Item	Description
Avail Proc Pool	Indicates the available physical processors in the system (default shared processor pool). **Note:** Default shared processor pool contains the physical processors that are available on the managed system. The **topas** command retrieves the APP value from the data that is provided by the LPARs that are in the same managed system. If these LPARs do not belong to the default shared processor pool, the **topas** command cannot determine the APP value for the managed system. The APP value is indicated by the - (hyphen) character in this case.
Shr Physical Busy	The sum of physical busy of all of the shared partitions
Ded Physical Busy	The sum of dedicated busy of all of the dedicated partitions
Donated Phys. processors	The sum of the donated processor cycles from all of the partitions reported as a number of processors
Stolen Phys. processors	The sum of stolen processor cycles from all of the partitions reported as a number of processors
Virtual Pools	The number of virtual pools
Virt. Context Switch	The total number of virtual context switches per second in the monitoring interval
Phantom Interrupts	The total number of phantom interrupts per second in the monitoring interval

When the **topas** command is running inside any cross partition view, press the **p** key to bring up the pool panel. The following is an example that displays:

```
pool  psize entc maxc physb app   mem  muse
0     3.0    2.0  4.0  0.1  2.0   1.0  1.5
1     4.0    3.0  5.0  0.5  1.5   1.0  0.5
2     3.0    2.5  4.0  0.2  2.0   1.0  0.5
```

You can scroll up or down in the pool ID column and press the **f** key to list only the shared partitions that belong to the **poolid** where cursor is positioned. The following headers might be displayed in the screen:

Item	Description
psize	The effective maximum capacity of the pool
entc	The entitled capacity of the pool
maxc	The maximum capacity of the pool
physb	The sum of physical busy of processors in shared partitions of a pool
app	The available physical processor in the pool
mem	The sum of monitored memory for all shared partitions in the pool
muse	The sum of memory consumed for all shared partitions in the pool

When the **topas** command is running inside any cross-partition view, press the **v** key to display the **Virtual I/O Server/Client Throughput panel**. The following metrics are displayed:

Item	Description
AQD	The average number of requests that are waiting to be sent.
AQW	The average time spent by a transfer request in the waiting queue. The suffix indicates the unit of time. The default unit of time is millisecond.

Item	Description
ART	The average time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
AWT	The average time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
Client	The name of the VIO Client.
KBPS	The amount of data that is read and written in kilobytes per second over the monitoring interval. This field is the sum of the values of the **KB-R** and **KB-W** metrics.
KB-R	The number of kilobytes that are read per second.
KB-W	The number of kilobytes that are written per second.
MRT	The maximum time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
MWT	The maximum time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default unit of time is millisecond.
Server	The name of the VIO Server.
TPS	The number of transfers that are issued per second.

When the **topas** command is running inside the Virtual I/O Server/Client Throughput panel, press the **d** key after selecting a server from the Virtual I/O Server/Client Throughput panel to toggle to **VIO Server/ Client Disk Details** panel. This panel displays the server adapter details in the top section and displays the target device and client disk details in the bottom of the section. To list the target devices and client disks belong to that adapter, select the adapter and press the **f** key.

The following metrics are displayed in a Virtual I/O Server/Client Disk Details panel:

Item	Description
Adapter	The name of the server adapter.
Vtargets	The name of the virtual target device that belongs to the server adapter.
Client_disk	The name of the client disk that is mapped to the virtual target device of the server adapter.

The following details of the adapters are displayed on the top section of the panel:

Item	Description
KBPS	The amount of data transferred (read or written) in the adapter in kilobytes per second.
TPS	The number of transfers per second that are issued to the adapter.
KB-R	The total number of kilobytes read from the adapter.
KB-W	The total number of kilobytes written to the adapter.
AQD	The number of requests waiting to be sent to the adapter.
AQW	The average time spent by a transfer request in the waiting queue. The suffix indicates the unit of time. The default time unit is millisecond.
ART	The time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond.
AWT	The time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond.

Item	Description
MRT	The maximum time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond.
MWT	The maximum time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond.

The following details for the virtual target device and client disk are displayed on the panel:

Item	Description
Busy%	The percentage of time the that the virtual target device or disk is active (bandwidth use of the virtual target device or disk).
KBPS	The number of kilobytes read and written per second over the monitoring interval. This field is the sum of the values of the **KB-R** and **KB-W** metrics.
TPS	The number of transfers per second that are issued to the virtual target device or disk. A transfer is an I/O request to the virtual target device or disk. Multiple logical requests can be combined into a single I/O request to the virtual target device or disk. A transfer is of medium size.
KB-R	The number of kilobytes read per second from the virtual target device or disk.
KB-W	The number of kilobytes written per second to the virtual target device or disk.
AQD	The average number of requests waiting to be sent to virtual target device or disk.
AQW	The average time spent by a transfer request in the waiting queue. The suffix indicates the unit of time. The default time unit is millisecond.
ART	The average time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond.
AWT	The average time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond.
MRT	The maximum time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond.
MWT	The maximum time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond.

To display the **Memory Pool panel** from the CEC panel, press the **m** key. This panel displays the statistics of all of the memory pools in the system. To display the partitions corresponding to that pool in the lower section of the panel, select a particular memory pool and press the **f** key.

The following values are displayed in the header section of the panel:

Item	Description
Mshr	The number of logical partitions (LPAR) running in the shared-memory mode.
Mded	The number of LPARrunning in dedicated-memory mode.
Pools	The total number of memory pools in the system.
Mpsz	The total size of physical memory of all the memory pools in gigabytes.
MPuse	The total memory used by LPAR associated with all of the pools in gigabytes.
Entl	The total I/O memory entitlement of all of the LPAR in all the pools in gigabytes.

Item	Description
Use	The total I/O memory entitlement in use of all of the LPAR in all the pools in gigabytes.
Mon	The total monitored memory of the system (sum of the values of the **Mpsz** metric and the **Total memory of dedicated memory partitions** metric).
InUse	The total memory in use of the system (sum of the **MPuse** metric and **Total memory inuse for dedicated memory partitions** metric).
Avl	The total memory available for the system (the value of the **Mon** metric minus the value of the **InUse** metric).

The following values of the pools are displayed:

Item	Description
mpid	The ID of the memory pool.
mpsz	The size of the total physical memory of the memory pool in gigabytes.
mpus	The total memory of the memory pool in use (this is the sum of the physical memory allocated to all of the LPAR in the pool).
mem	The size of the aggregate logical memory of all the partitions in the pool in gigabytes.
memu	The aggregate logical memory that is used for all the partitions in the pool in gigabytes.
iome	The aggregate of I/O memory entitlement that is configured for all the LPAR in the pool in gigabytes.
iomu	The aggregate of the I/O memory entitlement that is used for all the LPAR in the pool in gigabytes.
hpi	The aggregate number of hypervisor page faults that have occurred for all of the LPAR in the pool.
hpit	The aggregate of time spent in waiting for hypervisor page-ins by all of the LPAR in the pool in milliseconds.

The following values of the partitions in the pools are displayed:

Item	Description
mem	The size of logical memory of the partition in gigabytes.
memu	The logical memory that is used for the partition in gigabytes.
meml	The logical memory loaned to hypervisor by the LPAR.
pmem	The physical memory that is allocated to the partition from the memory pool in gigabytes.
iom	The amount of I/O memory entitlement that is configured for the LPAR in gigabytes.
iomu	The amount of I/O memory entitlement that is used for the LPAR in gigabytes.
hpi	The number of hypervisor page faults.
hpit	The time spent in waiting for hypervisor page-ins in milliseconds.
vcsw	The virtual context switches average per second.
physb	The physical processor that is busy.
%entc	The percentage of the consumed processor entitlement.

Cluster Utilization View

A cluster is a group of related partitions or nodes. The Cluster Utilization view can either show utilization of an HA cluster or a user-defined cluster. This panel displays metrics similar to the `lparstat` command for all the AIX partitions it can identify as belonging to the same hardware platform. The dedicated and shared partitions are displayed in separate sections with appropriate metrics. The top section represents aggregated data from the partition set to show overall partition, memory, and processor activity.

The following metrics are displayed in an initial cluster utilization panel. Additional metrics with full descriptive labels can be displayed using the key toggles identified in the Additional Cluster Utilization Panel Subcommands topic.

Partition totals:

Item	Description
Shr	The number of shared partitions based on the system processor.
Ded	The number of dedicated partitions based on the system processor.

Memory (in GB):

Item	Description
Mon	The total memory of monitored partitions.
InUse	The memory in use on monitored partitions.

Processor:

Item	Description
Shr	The number of shared processors.
Ded	The number of dedicated processors.
Shr_PhysB	The total number of physical processors that are busy for all shared partitions.
Ded_PhysB	The total number of physical processors that are busy for all dedicated partitions.

Individual partition data:

Item	Description
Host	The host name.
CEC	The CEC identifier.
OS	The operating system level
Mem	The total memory measured in gigabytes.
M	The mode of the individual partitions.
InU	The memory in use measured in gigabytes.
Lp	The number of logical processors.
Us	The percentage of the processor used by programs executing in user mode.
Sy	The percentage of the processor used by programs executing in kernel mode.
Wa	The percentage of time spent waiting for I/O.
Id	The percentage of time the processors are idle.

Item	Description
PhysB	The number of physical processors that are busy.
Ent	The entitlement granted (shared-only).
%Entc	The percentage entitlement consumed (shared-only).
Vcsw	The average of virtual context switches per second (shared-only).

For shared partitions

Character	Description
C	SMT enabled and capped
c	SMT disabled and capped
U	SMT enabled and uncapped
u	SMT disabled and uncapped

For dedicated partitions

Character	Description
S	SMT enabled and not donating
d	SMT disabled and donating
D	SMT enabled and donating
-	SMT disabled and not donating

The following data is displayed when you press the **g** key on the initial screen, which generates the cluster utilization view with detailed headers:

```
    Topas CEC Cluster Monitor ID:        Interval:  10             Thu Apr 2 16:13:18 2009
Partitions             Memory (GB)        Processor
Shr  :2                  Mon : 6.0          Shr  :1.5         Shr_PhyB : 0.01
Ded  :2                  InU : 3.0          Ded  :2           Ded_PhyB : 0.00

Host        CEC    OS   M  Mem  InU Lp Us Sy Wa Id  PhysB  Vcsw Ent  %EntC
-------------------------------------shared-------------------------------
clock16  19318230  A61  U  2.0  1.1 2  0  0  0  99  0.00   423  0.75  0.6
clock15  19318230  A61  U  2.0  1.6 2  0  0  0  99  0.01   985  0.75  0.9

Host        CEC    OS   M  Mem  InU Lp Us Sy Wa Id  PhysB  Vcsw
-------------------------------------dedicated------------------
ses10    19318230  A61  D  2.0  1.1  2 0  0  0  99  0.00   0
clock10  19318230  A61  D  0.0  0.0  2 0  0  0  99  0.00   742

The following display when press g key from the above panel,
which brings the cluster utilization view with detailed headers:

Topas Cluster Monitor ID:         Interval:  10      Thu Apr  2 16:13:44 2009
Partition Info    Memory (GB)      Processor         Supplier: ses10.in.ibm.com
Monitored :4      Monitored:6.0    Monitored :3.5    Shr Physical Busy :0.01
Shared    :2      Consumed :3.0    Shared    :1.5    Ded Physical Busy :0.00
Uncapped  :2                       Dedicated :2
Capped    :2
Dedicated :2

Host        CEC    OS   M  Mem  InU Lp Us Sy Wa Id  PhysB  Vcsw Ent  %EntC
-------------------------------------shared-------------------------------
clock16  19318230  A61  U  2.0  1.1 2  0  0  0  99  0.00   423  0.75  0.6
clock15  19318230  A61  U  2.0  1.6 2  0  0  0  99  0.01   985  0.75  0.9

Host        CEC    OS   M  Mem  InU Lp Us Sy Wa Id  PhysB  Vcsw
-------------------------------------dedicated------------------
ses10    19318230  A61  D  2.0  1.1  2 0  0  0  99  0.00   0
clock10  19318230  A61  D  0.0  0.0  2 0  0  0  99  0.00   742
```

Implementation Specifics

Disks and network adapters added after starting **topas** or any other SPMI consumer will not be reflected in **topas**. You must stop **topas** and all clients that use SPMI and then restart after the changes to disks and network adapters are made.

Flags

Item	Description
-@*wparname*	Shows the WPAR-specific metrics. If you specify a WPAR name with the *wparname* parameter, the **topas** monitors that WPAR.
-c*hotprocessor*	Specifies with the*hotprocessor* parameter the number of hot processors to be monitored. This is also the maximum number of processors displayed when enough room is available on the screen. If this number exceeds the number of processors available, only the installed processors will be monitored and displayed. If this argument is omitted, a default of 2 is assumed. If a value of `0` (zero) is specified, no processor information is monitored.
`-C`	Displays the Cross-partition panel. The `topas` command collects a set of metrics from AIX partitions running on the same hardware platform. The metrics are similar to those collected by the `lparstat` command. Dedicated and shared partitions are displayed, and a set of aggregated values provide an overview of the entire hardware systems partition set. Certain values only available from the HMC platform can be set through the `line` command if an HMC connection is not available.
-G	Displays the Cluster Utilization panel. The `topas` command collects a set of metrics from AIX partitions that are running on the same hardware platform. The metrics are similar to those collected by the `lparstat` command. Dedicated and shared partitions are displayed.

Item	Description
-D	Displays the Disk Metrics display (Disk panel view). The display reports disk service times, disk queuing metrics, and disk throughput. The following metrics are reported: **Disk** The name of the physical disk. **Busy%** The percentage of time that the physical disk is active (bandwidth use for the disk). **KBPS** The number of kilobytes that are read and written per second over the monitoring interval. This field is the sum of the values of the **KB-R** and **KB-W** metrics. **TPS** The number of transfers per second that are issued to the physical disk. A transfer is an I/O request to the physical disk. Multiple logical requests can be combined into a single I/O request to the disk. A transfer is of medium size. **KB-R** The number of kilobytes read per second from the physical disk. **ART** The average time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond. **MRT** The maximum time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond. **KB-W** The number of kilobytes written per second to the physical disk. **AWT** The average time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond. **MWT** The maximum time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond. **AQW** The average time spent by a transfer request in the waiting queue. The suffix indicates the unit of time. The default time unit is millisecond. **AQD** The average number of requests that are waiting to be sent to disk. With the **-D** flag specified, you can run the following subcommands: • To view the Adapter Panel, press the **d** key. • To display all of the virtual adapters present in the partition (Virtual Adapter Panel), press the **v** key. • To display the disks that belong to the adapter or the virtual adapter, press the **f** key. • To display the MPIO Panel, press the **m** key. This panel displays the disks details and the path details. To list the paths of the disks, press the **f** key. **Limitation:** The **-D** option provides Disk panel view where it reports disk service times, disk queuing metrics, and disk throughput. Whenever **-D** option is started, it resets the disk minimum and maximum service time metrics during the first interval. Because the service time metrics are reset during first interval of **-D** option, the existing instance of **-D** option or some other consumer's use of the disk service time metrics is affected.
-d *hotdisk*	Specifies the number of disks to be monitored. The *hotdisk* parameter specifies the number of the hot disks to be monitored. This is also the maximum number of disks displayed when enough room is available on the screen. When this number exceeds the number of disks installed, only the installed disks will be monitored and displayed. If this argument is omitted, a default of 2 is assumed. If a value of 0 (zero) is specified, no disk information is monitored.

Item	Description
-E	Displays the statistics of the shared Ethernet adapter on a Virtual I/O Server. The following metrics are displayed: **KBPS** The total throughput in kilobytes per second over the monitoring interval. This field is the sum of the kilobytes received and kilobytes sent per second. **I-Pack** The number of data packets received per second over the monitoring interval. **O-Pack** The number of data packets sent per second over the monitoring interval. **KB-In** The number of kilobytes received per second over the monitoring interval. **KB-Out** The number of kilobytes sent per second over the monitoring interval.
-F	Displays the file system display. When you specify the flag with the **-@** flag or the **@** subcommand, file system is shown in two windows. The top part of the display shows a list of active WPAR. This list can be sorted on any column. The display reports file system service times, file system queuing metrics, and file system throughput. The following metrics are reported: **File System** The name of file system. **KBPS** The amount of data transferred (read and written) per second over the monitoring interval. This field is the sum of the values of **KB-Read** and **KB-Writ**. **TPS** The number of transfers per second that are issued to the file system. A transfer is an I/O request to the file system. Multiple logical requests can be combined into a single I/O request to the file system. The size of a transfer is not determinate. **KB-Read** The amount of kilobytes read per second from the file system. **KB-Writ** The amount of kilobytes written per second from the file system. **Open** The logical number of files open. **Create** The logical number of files creates. **Lock** The number of files lock file system. **Tip:** If the file system name exceeds the field width in the display, then the file system name is displayed is truncated. The truncation contains the first and last few characters of the file system, and the middle part of the name is replaced by periods (..). For example, if the file system name is `filesystem001234`, then the file system name is displayed as `files..01234`.
-f *HotFS*	Specifies with the *HotFS* parameter the number of file system to be monitored. This is also the maximum number of file system displayed when enough room is available. When this number exceeds the number of file system mounted, only the mounted file system is monitored and displayed. If you do not specify the **-f** flag, the default value is two. If you specify a value of zero, the file system information is monitored.
-h	Displays help information in the following format: `usage: topas [-d number-of-monitored-hot-disks]` `[-h]` `[-i monitoring-interval_in_seconds]` `[-n number-of-monitored-hot-network-interfaces]` `[-p number-of-monitored-hot-processes]` `[-w number-of-monitored-hot-WLM classes]` `[-c number-of-monitored-hot-processors]` `[-U username_owned_processes]` `[-D|-P|-W|-L]` `[-m]`
-i *interval*	Sets the monitoring interval or the recording interval in seconds. If you specify the **-i** flag with the *interval* parameter, the *interval* parameter sets the monitoring intervals. The default value for the*interval* parameter is two seconds. If you specify the **-i** flag with the **-R** mode, the *interval* parameter becomes the recording interval for partition metrics. The default value for the*interval* parameter is 300 seconds. Valid values are 10, 15, 30, 60, 120, and 300 seconds.

Item	Description
`-I` *remotepollinterval*	For cross-partition display, sets with the *remotepollinterval* parameter the sampling interval to collect data from remote partitions. The default value for the *remotepollinterval* parameter is 10 seconds. Values of 10, 15, 30, 60 and 120 seconds are allowed.
-L	Displays the logical partition display. This display reports similar data to what is provided to **mpstat** and **lparstat**. In shared-memory mode, this panel displays information about I/O memory entitlement of the partition. The existing **%lbusy**, **%hypv** and **hcalls** metrics are replaced by the following metrics: **IOME** The I/O memory entitlement of the partition in gigabytes. **iomu** The I/O memory entitlement of the partition in use in gigabytes. **pmem** The physical memory that is backing logical memory of the partition in gigabytes. **hpi** The number of hypervisor page-ins. **hpit** The time in milliseconds waiting for hypervisor page-ins. With the **-L** flag specified, you can press the **e** key to display the I/O Memory Entitlement Pools panel. For more information about this panel, see **I/O Memory Entitlement Pools Panel**.

Item	Description
-M	Displays the Memory topology panel. The display reports similar data to what is provided by the **lssrad** command. There are two sections in this panel: • The first section gives us the memory topology from an SRAD point of view. Under every **REF1** system detail level, it provides the individual SRAD IDs and the resources (memory, processors) associated with each of them. • The second section, the CPU RAD display, gives the relevant data at a processor level. The following metrics are displayed as part of this panel. **REF1** The first hardware provided reference point, that identifies sets of resources that are near to each other. **SRAD** Scheduler Resource Allocation Domain ID. **TOTALMEM** The total memory in MB under the SRAD. **INUSE** The memory in use under the SRAD. **FREE** Free memory under the SRAD. **FILECACHE** The number of file cache bytes that are taken by the LRU daemon. **HOMETHRDS** The number of threads for which the SRAD is home. Threads typically run on the CPUs contained in the home SRAD, but it is not guaranteed. The system chooses a home SRAD for a thread when it is created. A thread's home SRAD may change during a thread's lifetime. **CPUS** The processors which are associated with this SRAD. **0** would indicate that cpu0 is associated with the corresponding SRAD id. **0 - 28** would indicate that all **cpus** from **cpu0** to **cpu28** are associated with the corresponding SRAD. If the cpu ids are not contiguous, then the values will be separated by commas. **TOTALDISP** Total number of threads dispatched from the corresponding processor during that interval. **LOCALDISP%** Percentage of threads that were dispatched locally within this SRAD, usually at the chip level. **NEARDISP%** Percentage of threads that were dispatched to a CPU that is not local, and that is not far. Typically, these may be resources that share the same hardware node. **FARDISP%** Percentage of threads that were dispatched to a processor typically outside the hardware node. **Note:** The hardware meanings for local, near and far vary with varying architectures.
-m	Displays in monochrome mode (no colors).
-n *hotni*	Specifies with the *hotni* parameter the number of hot network interfaces to be monitored. This is also the maximum number of network interfaces displayed when enough room is available on the screen. When this number exceeds the number of network interfaces installed, only the installed network interfaces will be monitored and displayed. If this argument is omitted, a default of value of 2 is assumed. If a value of 0 (zero) is specified, no network information is monitored.

Item	Description
-P	Similar to the **ps** command, the **-P** flag displays the full-screen process display. This display shows a list of the busiest processes, similar to the process subsection on the default display, only with more columns showing more metrics per process. This list can be sorted by any column. Following are the metrics displayed. **USER** The login name of the process owner. Truncates the user name to 8 characters. **PID** The process ID of the process. **PPID** The process ID of the parent process. **PRI** The priority of the process or kernel thread; higher numbers mean lower priority. **NI** The priority of a process specified with the **nice** command ; used in calculating priority for the sched other policy. **DATA RES** The real-memory data (resident set) size of the process (4 KB pages). **TEXT RES** The real-memory text (resident set) size of the process (4 KB pages). **PAGE SPACE** The virtual working set size used by process (4 KB pages). **Note:** The true paging space allocations per process are not available using the **topas** command. For more detailed reports, see the **svmon** command. **TIME** The total execution time for the process. **CPU%** The percentage of processor usage. **PGFAULTS** The number of I/O and other page faults. **COMMAND** The command name. Truncates the command name to 9 characters. When specified with **-@** (**topas –P -@**), a new field WPAR is displayed and the **PPID** field is removed. All other metrics remains the same. **WPAR** The WPAR name that the process belongs to. **Tip:** If the WPAR class name exceeds 12 characters and it need to be displayed in a 12 character format, the first five characters will be followed by two periods (.), and then follows the last five characters. For example, if the WPAR class name is `neptune001234`, then the WPAR name is displayed as `neptu..01234`.
-p*hotprocess*	Specifies with the *hotprocess* parameter the number of hot processes to be monitored. This is also the maximum number of processes shown when enough room is available on the screen. If this argument is omitted, a default of 20 is assumed. If a value of 0 is specified, no process information will be monitored. Retrieval of process information constitutes the majority of the **topas** overhead. If process information is not required, always use this option to specify that you do not want process information.
-t	Toggles the tape display section on or off in the main **topas** display.

Item	Description
-T	Displays the full screen tape display panel. **Note:** Only the **Atape** device utilization is reported. The following metrics are displayed in this panel: **Tape** The name of the tape device. **Busy%** The bandwidth use of the tape. **KBPS** The amount of data transferred (read or written) to the tape in kilobytes per second. **TPS** The average number of transfers per second issued to the tape. **KB-R** The total number of kilobytes read from the tape. **ART** The average time to receive a response for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond. **MRT** The maximum time to receive a response for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond. **KB-W** The total number of kilobytes written to the adapter. **AWT** The average time to receive a response for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond. **MWT** The maximum time to receive a response for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond.
-U *username*	With the **-P** flag, this flag shows the processes owned by the user specified with the *username* parameter. Only processes owned by the user that is specified will be shown in the `All Process Display`.
-V	Displays the Volume Group panel. The panel reports the following metrics of the volume groups in the top section of the panel, and the same metrics of the logical volumes in the bottom section of the panel. **LogicalVolume/VolumeGroup** The name of the logical volume or the volume group. **TPS** The total number of I/O requests over the interval that the metrics are displayed. **KB-R** The total number of kilobytes read over the interval. **KB-W** The total number of kilobytes written over the interval. **KBPS** The amount of data transferred (read or written) in kilobytes per second in the enquiring logical volume or volume group.
-W	Displays the full-screen WLM class display, which is a split display. The top part of the display shows a list of hot WLM classes, similar to the WLM classes subsection on the default display, but with enough space available to display the full class names. This list can be sorted on any column. If you specify the **-@** flag, or if you press the **@** subcommand, the WPAR section is displayed and the WLM section is not displayed. The WPAR section shows the list of hot WPAR. This list can be sorted on any column. The bottom part of the display shows a list of busiest processes, similar to the full screen process display, but only displays processes that belong to one WLM class or WPAR that are selected with the **f** key. **Note:** If the WLM class is not active then the default system processes will be displayed in the bottom part of the display.
-w *[number of monitored hot WLM classes]*	Specifies with the *hotwlmclass* parameter the number of hot Workload Manager (WLM) classes to be monitored. This is also the maximum number of WLM classes displayed when enough room is available on the screen. If this number exceeds the number of WLM classes installed, only the installed WLM classes will be monitored and displayed. If this argument is omitted, a default of 2 is assumed. If a value of 0 (zero) is specified, no WLM class information is monitored.

General Subcommands

While **topas** is running, it accepts 1-character subcommands. Each time the monitoring interval elapses, the program checks for one of the following subcommands and responds to the action requested.

Item	Description
a	Shows all of the variable subsections being monitored (processor, network, disk, WLM, and process). Pressing the **a** key always returns the **topas** command to the initial main display.
c	Replaces the current display of the cumulative report with the processor subsection. When you press the **c** key again, it displays the cumulative report. The number of busiest processors displayed will depend upon the space available on the screen.
C	Activates the Cross-Partition panel. If the panel is currently active, the **C** key resets the panel to display the global summary, dedicated, and shared sections. See the Additional Cross-partition Panel Subcommands section below for options specific to this panel.
d	Replaces the current display of the total disk activity with a list of the busiest disks. When you press the **d** key again, it displays the total disk activity. The number of busiest disks displayed will depends on the space available on the screen.
D	Replaces the current display with the Disk Metric display. This display offers additional information about disk access times and disk queuing. If the D key is pressed again, the display toggles back to the default main screen.
E	Shows the shared Ethernet adapter panel in VIO Server.
f	Press the **f** key while moving the cursor over a WLM class to display the list of top processes in the class at the bottom of the WLM screen. In the file system subsection of the **topas** command main panel, press the **f** key to replace the default report of total file system activity of the system with a list of busiest file system. When you press the **f** key again, it returns to the default display of the total file system activity. The number of busiest file system depends upon the space available on the screen. In the Volume Group panel (**topas -V**), you can select a volume group name and press the **f** key to display the list of top logical volumes that belong to the volume group at the bottom of the LVM panel.
F	Replaces the default display with the full-screen file system display. This display provides more detailed information about file systems on the system than the file system section of the main display. When the you press the **F** key again, it returns to the default main display.
G	Activates the Cluster Utilization panel. If the panel is currently active, the **G**key resets the panel to display the global summary, dedicated, and shared sections. See the Additional Cluster Utilization Panel Subcommands topic for options specific to this panel.
h	Shows the help screen.
H	Shows the help screen for the local panel, if available.
L	Replaces the current display with the logical partition display; LPAR, Micro-Partitioning, and simultaneous multithreading metrics similar to what **lparstat** and **mpstat** provide are displayed.
n	Replaces the report on the total network activity of the system with the list of the busiest interfaces. Press the **n** key in the network interfaces subsection. The number of busiest interfaces displayed will depend upon the space available on the screen.

Item	Description
p	Toggles the hot processes subsection on and off. The number of busiest processes displayed will depend upon the space available on the screen.
P	Replaces the default display with the full-screen process display. This display provides more detailed information about processes running on the system than the process section of the main display. When the **P** key is pressed again, it toggles to the default main display.
q	Quits the program.
r	Refreshes the display.
t	Toggles the tape display on or off in the main panel.
T	Shows the full-screen tape display.
V	Shows the Volume Group panel.
w	Toggles the Workload Manager (WLM) classes subsection on and off. The number of busiest WLM classes displayed will depend upon the space available on the screen.
W	Replaces the default display with the full-screen WLM class display. This display provides more detailed information about WLM classes, WPAR classes, and processes assigned to classes. When you press the @ key, the WLM class subsection is replaced by WPAR subsection. When you press the **W** key again, it toggles back to the default main display.
@	Toggles between the WLM class metric and WPAR metrics, that is, WPAR is monitored instead of WLM. This is the at (@) key. This key is valid for the Main panel, Process panel, File System panel, and WLM panel. If you press the **@** key from any other panel, it is ignored. The **@** key is restricted inside a WPAR, that is, it is ignore inside a WPAR. The **@** key is valid in the following panels: **Main Panel** The WLM and Process subsections are replaced by the WPAR metric. **Process Panel** The default mode of the process panel is replaced by the WPAR mode. **File System Panel** The file system panel contains WPAR names if you press the **f** key. The per WPAR file-system metrics are displayed on the lower section of this panel. **WLM Panel** The WLM subsection is replaced by the WPAR subsection.
Arrow and Tab keys	Subsections from the main display such as the processor, Network, Disk, WLM Classes, and the full-screen WLM and Process displays can be sorted by different criteria. Positioning the cursor over a column activates sorting on that column. The entries are always sorted from highest to lowest value. The cursor can be moved by using the **Tab** key or the arrow keys. Sorting is only valid for 128 disks and 16 network adapters.
~	Shows the **nmon** screen. This is the tilde (**~**) key.

Additional Cross-Partition Panel Subcommands

When the `topas` Cross-partition panel is active, it accepts the following additional 1-character subcommands. Each time the monitoring interval elapses, the program checks for one of the following subcommands and responds to any requested action.

Item	Description
d	Toggles the dedicated partition section on and off.
g	Toggles the top global section of the panel between brief listing, detailed listing, and off.
r	Forces `topas` to search the for HMC configuration changes if a connection is available. This includes the discovery of new partitions, processors, or memory allocations.
s	Toggles the shared partition section on and off.
p	Toggles the pool panel section on or off. Inside the pool panel, user can select one pool ID and press the **f** key to list the shared partitions that belong to the pool.
v	Toggles the Virtual I/O Server/Client Throughput details on or off. You can select one virtual I/O server and press the **f** key to list the VIO clients that belong to that server.
m	Toggles the memory pool panel on or off. You can select a memory pool and press the **f** key to view the partitions in that pool.

Additional Cluster Utilization Panel Subcommands

When the `topas` Cluster Utilization panel is active, it accepts the following additional 1-character subcommands. Each time the monitoring interval elapses, the program checks for one of the following subcommands and responds to any requested action:

Item	Description
d	Toggles the dedicated partition section on and off.
g	Toggles the top global section of the panel between brief listing, detailed listing, and off.
s	Toggles the shared partition section on and off.

Additional Disk Panel (topas -D) Subcommands

When the `topas` Disk panel is active, it accepts the following additional 1-character subcommands. Each time the monitoring interval elapses, the program checks for one of the following subcommands and responds to any requested action:

Item	Description
d	Toggles the Adapter panel on or off.
m	Toggles the MPIO panel on or off.

Additional Adapter Panel Subcommands

When the `topas` Adapter panel is active, it accepts the following additional 1-character subcommand. Each time the monitoring interval elapses, the program checks for one of the following subcommands and responds to any requested action:

Item	Description
v	Toggles the Virtual Adapter panel on or off. Press this key from the Adapter panel.

Additional Logical Partition Panel (topas –L) Subcommands

When the topas Logical panel is active, it accepts the following additional 1-character subcommand. Each time the monitoring interval elapses, the program checks for one of the following subcommands and responds to any requested action:

Item	Description
e	Toggles the I/O Memory Entitlement Pools panel.

Additional Virtual I/O Server/Client Throughput Panel Subcommands

When the topas Virtual I/O Server/Client Throughput panel is active, it accepts the following additional 1-character subcommand. Each time the monitoring interval elapses, the program checks for one of the following subcommands and responds to any requested action:

Item	Description
d	Turns the Virtual I/O Server/Client Disk panel on or off for the Virtual I/O Server that is selected in the Virtual I/O Server/Client Throughput panel. You can select the server adapters and press the **f** key to list the disks and the clients that belong to that adapter.

Sample Full-Screen Workload Manager Classes Output

The following is an example of the display generated by the **topas -W** command:

```
Topas Monitor for host:    ptoolsl3    Interval:   2    Mon Feb 12 06:25:11 2007
WLM-Class (Active)              CPU%      Mem%     Blk-I/O%
System                           0        57           0
Shared                           0         4           0
Default                          0         0           0
Unmanaged                        0        14           0
Unclassified                     0        38           0

================================================================================
                              DATA  TEXT  PAGE              PGFAULTS
USER       PID  PPID PRI NI   RES   RES SPACE    TIME CPU%  I/O  OTH COMMAND
root         1     0 108 20   197     9   180    0:24  0.0    0    0 init
root      1032     0  16 41     3  3374     3    0:00  0.0    0    0 lrud
root      1290     0  60 41     4  3374     4    0:02  0.0    0    0 xmgc
root      1548     0  36 41     4  3374     4    0:26  0.0    0    0 netm
root      1806     0  37 41    16  3374    16   13:25  0.0    0    0 gil
root      2064     0  16 41     4  3374     4    0:04  0.0    0    0 wlmsched
root      2698     1 108 20    14     2    14    0:00  0.0    0    0 shlap
root      3144     1 108 20    40     1    36    5:19  0.0    0    0 syncd
root      3362     0 108 20     4  3374     4    0:00  0.0    0    0 lvmbb
root      3666     1 108 20   135    23   123    0:00  0.0    0    0 errdemon
root      3982     0 108 20     4  3374     4    0:01  0.0    0    0 rtcmd
```

The following is an example of the display generated by **topas –W -@** command:

```
Topas Monitor for host:    ptoolsl3    Interval:   2    Mon Feb 12 06:25:11 2007
WPAR                            CPU%       Mem%     Blk-I/O%
neptune001234                     0          1           0









================================================================================
                                DATA  TEXT  PAGE              PGFAULTS
USER         PID  PPID   PRI NI   RES   RES SPACE    TIME CPU%  I/O  OTH COMMAND
root      356372  491650  58 41   370    67   370    0:00  0.1    0    0 topas
root      262246  188508  24 41   256    21   256    6:27  0.1    0    0 xmtopas
root      192626       1  60 20   113    17   113   11:17  0.1    0    0 getty
root       61470       0  16 41    17     0    17    0:31  0.0    0    0 wlmsched
root      290818       1  58 41   284    67   284    1:54  0.0    0    1 topas
```

```
root      57372       0  37 41    30     0    30    3:39  0.0    0    0 gil
root      86248       1  60 20    47     0    47    1:04  0.0    0    0 rpc.lock
root     385224  237728  60 20   254   197   254    0:00  0.0    0    0 sendmail
root     131174  176242  60 20   175    79   175    0:03  0.0    0    0 aixmibd
root      53274       0  36 41    13     0    13    0:05  0.0    0    0 netm
root      90244       1  60 20   126     2   126    2:35  0.0    0    0 syncd
root      45078       0  60 41    14     0    14    0:58  0.0    0    0 xmgc
root     266384  176242  60 20   644   160   644    0:27  0.0    0    0 IBM.CSMA
root     250004  176242  60 20   617   157   617    0:26  0.0    0    0 rmcd
root     184410  176242  60 20   254   197   254    0:14  0.0    0    0 sendmail
root     151640       0  60 20    13     0    13    0:02  0.0    0    0 rgsr
root      40980       0  59 41    71     0    71    0:02  0.0    0    0 pilegc
root     110738       0  60 20    13     0    13    0:01  0.0    0    0 n4bg
root     180368       1  60 20    98    14    98    0:01  0.0    0    0 cron
root          1       0  60 20   158    10   158    0:01  0.0    0    0 init
```

Examples

1. To display up to twenty "hot" disks every five seconds and omit network interface, WLM classes, file system information and process information, enter the following command:

   ```
   topas -i5 -n0 -p0 -w0 -f0
   ```

2. To display the five most active processes and up to twenty most active WLM classes (which is the default when omitting the **-w** flag) but no network , disk, or file system information, enter the following command:

   ```
   topas -p5 -n0 -d0 -f0
   ```

3. To run the program with default options, enter the following command:

   ```
   topas
   ```

4. To go directly to the process display, enter the following command:

   ```
   topas -P
   ```

5. To go directly to the WLM classes display, enter the following command:

   ```
   topas -W
   ```

6. To go directly to the logical partition display, enter the following command:

   ```
   topas -L
   ```

7. To go directly to the disk metric display, enter the following command:

   ```
   topas -D
   ```

8. To go directly to the file system display, enter the following command:

   ```
   topas -F
   ```

9. To go directly to WPAR monitoring mode *abc*, enter the following command:

   ```
   topas -@ abc
   ```

10. To go directly to the **topas** WPAR mode, enter the following command:

    ```
    topas -@
    ```

11. To go directly to the LVM display, enter the following command:

    ```
    topas –V
    ```

12. To go directly to the tape display, enter the following command:

```
topas –T
```

13. To go to the shared Ethernet adapter on the VIO Server panel, enter the following command:

```
topas -E
```

14. To go directly to the cluster utilization display, enter the following command:

```
topas -G
```

15. To go directly to the Memory topology panel and view SRAD statistics, enter the following command:

```
topas -M
```

16. To display the process utilization specific to the user **guest**, enter the following command:

```
topas -P -U guest
```

17. To display top two processors with high processor utilization, enter the following command:

```
topas -c2
```

Files

Item	Description
/usr/bin/topas	Contains the **topas** command.

topasout Command

Purpose

Generates reports by processing **xmwlm**, **nmon**, and **topas** recordings.

Syntax

Local reports

topasout -R *type* [**-i** *interval*] [**-b** *time*] [**-e** *time*] *topas_recording_file*

Comma-separated report

topasout -c [**-m** *type*] *topas_recording_file*

Spread-sheet report

topasout [**-s**] [**-m** *type*] *topas_recording_file*

Nmon analyzer report

topasout -a *topas_recording_file*

WLE Report from topasrec / nmon file

topasout -R *wle* { nmon_recording_file | topas_recording_file }

CEC reports

topasout -R *type* [**-i** *interval*] [**-b** *time*] [**-e** *time*] *topas_recording_file*

Comma-separated report

topasout [**-c**] *topas_recording_file*

Spread-sheet report

topasout -s *topas_recording_file*

Description

The **topasout** command is used to convert the binary recordings generated by the **xmwlm**, **xmtrend**, or **topasrec** utilities. The binary recording can be the local system recording, the central electronic complex (CEC) recording, or the cluster recording. Through SMIT, you can enable, configure, or disable a binary recording.

If there is more than one value for a metric within the user-specified interval, the **topasout** command averages out all of the values to get single value that can be printed in the report. For values that cannot be averaged out (like simultaneous multithreading, dedicated and shared modes), the **topasout** command takes the last or the first values that are recorded in the interval.

Local reports

There are several types of local reports: the Summary report, the Detailed report, the LAN report, the Disk report, the Comma-separated report, the Nmon analyzer report, the Adapter report, and the Virtual adapter report.

Summary report

A Summary report presents the consolidated view of system information.

The following column headings are in a summary report:

Item	Description
Time	Ending time of the report interval. Metric values are averaged out over this interval and printed in the report
InU	Memory that is used
Us	Percentage of processor time spent in the user mode
Sy	Percentage of processor time spent in the system mode
Wa	Percentage of processor time spent waiting for I/O
Id	Percentage of time that the processor is idle
PhysB	Percentage of physical processors that are busy
RunQ	The average number of threads that are ready to run but are waiting for a processor to become available
WtQ	The average number of threads that are waiting for paging to be completed
Cswitch	The number of context switches per second in the reporting interval
Syscall	The number of system calls executed per second in the reporting interval
PgFault	The number of I/O and other page faults
%don	Sum of **%idle** cycles donated and **%busy** cycles donated
%stl	Sum of **%idle** cycles stolen and **%busy** cycles stolen

The following sample shows the output of a local Summary report:

```
Report: System Summary  --- hostname: aixfvt19                    version:1.1
Start:01/24/07 04:45:50   Stop:01/24/07 04:48:07  Int: 5 Min  Range:   2 Min
Mem:  1.2 GB  Dedicated SMT: ON   Logical CPUs:  2
Time       InU Us Sy Wa  Id  PhysB RunQ  WtQ CSwitch Syscall PgFault
04:48:07   1.2  3  0  0  88   3.43  1.1  0.0     168     893      23
```

Detailed report

A detailed report provides a detailed view of the system metrics.

The following column headings are in a detailed report:

Item	Description
Mode	The information about the following modes are reported: • **Don** represents donating dedicated partition. • Ded represents that the dedicated partition are not donating or the donation is not enabled. • **Shr** represents shared mode.
Lp	Number of logical processors.
SMT	Status of the SMIT. It is On when the SMT is enable. It is `Off` when the SMT is disabled.
Ent	Entitlement granted (shared-only).
Poolid	Pool ID. This column is applicable only if this partition belongs to a valid shared processor pool.
Kern	Percentage of processor time spent in the kernel mode.
User	Percentage of processor time spent in the user mode.
Wait	Percentage of processor time spent for waiting for I/O.
Idle	Percentage of time that the processor is idle.
PhysB	Percentage of physical processors that are busy.
Entc	Percentage of entitled capacity that is consumed. This heading is applicable for shared partition only.
Sz, GB (in Memory section)	Memory size in gigabytes.
InU (in Memory section)	Memory used in gigabytes.
%Comp	Percentage of real memory that is allocated to computational page frames. Computational page frames are backed by paging space.
%Nonc	Percentage of real memory that is allocated to non-computational page frames. Non-computational page frames are backed by file space: either data files, executable files, or shared library files.
%Clnt	Percentage of real memory that is allocated to cache, remotely mounted files.
Sz, GB (in Paging section)	Paging space in gigabytes.
InU (in Paging section)	Paging space used in gigabytes.
Flt	Total number of page faults that are taken per second in the reporting interval. This includes page faults that do not cause paging activity.
Pg-I	Number of 4 K pages that are read per second in the reporting interval.
Pg-O	Number of 4 K pages that are written per second in the reporting interval.
Bdon	Percentage of physical processor that is used while busy cycles are being donated. This metric is applicable only for donating dedicated partitions.
Idon	Percentage of physical processor that is used while explicitly donating idle cycles. This metric is applicable only for donating dedicated partitions.

Item	Description
Istl	Percentage of physical processor that is used while idle cycles are being stolen by the hypervisor. This metric is applicable only for dedicated partitions.
Bstl	Percentage of physical processor that is used while busy cycles are being stolen by the hypervisor. This metric is applicable only for dedicated partitions. The **%idon** and **%bdon** metrics are not displayed when no dedicated partition is donating.
Vcsw	Average number of virtual context switches per second in the reporting interval.
Phint	Average number of phantom interrupts per second in the reporting interval. This column is applicable only to shared partitions.
Cswth	Number of process context switches per second in the reporting interval.
Syscl	Number of system calls per second run in the reporting interval.
RunQ	Average number of threads that are ready to run but are waiting for a processor to become available.
WtQ	Average number of threads that are waiting for paging to complete.
SrvV2	Number of NFS Server V2 calls per second in the reporting interval.
CltV2	Number of NFS Client V2 calls per second in the reporting interval.
SrvV3	Number of Server V3 calls per second in the reporting interval.
CltV3	Number of Client V3 calls per second in the reporting interval.
Network	Name of the network interface.
I-Pack	Number of data packets that are received per second.
O-Pack	Number of data packets that are sent per second in the reporting interval.
KB-I	Number of kilobytes that are received per second in the reporting interval.
KB-O	Number of kilobytes that are sent per second in the reporting interval.
Disk	Name of the physical disk.
Busy%	Percentage of time that the physical disks are active (bandwidth utilization for the drive).
KBPS	Number of kilobytes that are read and written per second in the reporting interval. This column is the sum of the **KB-R** and **KB-W** metrics.
TPS	Number of transfers per second that are issued to the physical disk. A transfer is an I/O request to the physical disk. Multiple logical requests can be combined into a single I/O request to the disk. The size of a transfer is not determinate.
KB-R	Number of kilobytes that are read per second from the physical disk in the reporting interval.
KB-W	Number of kilobytes that are written per second to the physical disk in the reporting interval.

The following sample shows a local Detailed report:

```
Sample output
#Report: System Detailed --- hostname: ptoolsl1                 version: 1.2
Start:12/21/05 10.00.00 Stop:12/21/05 11.00.00  Int: 5 Min  Range: 60 Min
Time: 10.00.00 -----------------------------------------------------
CONFIG        CPU             MEMORY        PAGING
Mode   Don    Kern   12.0     Sz,GB  16.0   Sz,GB  4.0
LP     4      User    8.0     InU     4.3   InU    2.3
SMT    ON     Wait    0.0     %Comp   3.1   Flt    221
Ent    3.0    Idle   80.0     %NonC   9.0   Pg-I    87
Poolid 3      PhyB    0.7     %Clnt   2.0   Pg-O    44
              EntC    8.0

PHYP          EVENTS/QUEUES   NFS
Bdon    0.1   Cswth    3213   SrvV2    32
Idon    0.5   Syscl   43831   CltV2    12
Bstl    0.5   RunQ        1   SrvV3    44
Istl    0.4   WtQ         0   CltV3    18
Vcsw    1214
Phint   120

Network  KBPS   I-Pack  O-Pack    KB-I    KB-O
en0       0.6      7.5     0.5     0.3     0.3
en1      22.3    820.1   124.3   410.0    61.2
lo0       0.0      0.0     0.0     0.0     0.0

Disk    Busy%     KBPS     TPS    KB-R    KB-W
hdisk0    0.0      0.0     0.0     0.0     0.0
hdisk1    0.0      0.0     0.0     0.0     0.0

        topasout local report – detailed report
```

Disk reports

A Disk report provides information about the amount of data that are read or written to disks.

The following column headings are in a Disk report:

Item	Description
Mem	Total memory that are available in gigabytes at the first reporting interval.
Logical CPUs	Number of logical processors at the first reporting interval.
Time	Ending time of the reporting interval. Metric values are averaged out over this time interval and printed in the report.
InU	Total memory that are used in gigabytes.
PhysB	Percentage of physical processors that are busy.
MBPS	Number of megabytes that are read and written per second. This column is the sum of the **MB-W** and **MB-R** metrics.
TPS	Number of transfers per second that are issued to the physical disk. A transfer is an I/O request to the physical disk. Multiple logical requests can be combined into a single I/O request to the disk. The size of a transfer is not fixed.
MB-R	Data that are read in megabytes per second from the physical disk.
MB-W	Data that are written in megabytes per second to the physical disk.

The following sample shows the output of a local Disk report:

```
Sample output
Report: Total Disk I/O Summary  --- hostname: aixfvt19   version:1.1
Start:01/24/07 04:45:50   Stop:01/24/07 04:48:07  Int: 5 Min  Range:15 Min
Mem:  1.2 GB   Dedicated SMT: ON   Logical CPUs:  2
Time      InU   PhysB   MBPS    TPS   MB-R   MB-W
04:48:07  1.2     3.4    0.2    2.1    0.1    0.1
04:53:07  1.2     3.4    0.3    2.1    0.0    0.3
...
```

LAN reports

A LAN report provides the amount of data that are received or sent in the network interfaces.

The following column headings are in a LAN report:

Item	Description
Mem	Total memory available in gigabytes at the first reporting interval.
Logical processors	Number of logical processors at the first reporting interval.
Time	Ending time of the reporting interval. Metric values are averaged out over this time interval and printed in the report.
InU	Total memory used in gigabytes.
PhysB	Percentage of physical processors that are busy.
MBPS	Sum of the **MB-I** and **MB-O** values. It equals to the data in megabytes sent and received per second.
MB-I	Data in megabytes that are received per second in the reporting interval.
MB-O	Data in megabytes that are sent per second in the reporting interval.
Xmtdrp	Average amount of transmitted packets that are dropped per second at device driver level in the reporting interval.
Rcvdrp	Average amount of received packets that are dropped per second at device driver level in the reporting interval.

The following sample shows the output of a local LAN report:

```
#Report: System LAN Summary  --- hostname: tooltime2                    version:1.1
Start:03/02/07 00:38:18 Stop:03/02/07 07:08:32   Int:  5 Min   Range:  390 Min
Mem:  4.0 GB  Shared SMT: ON  Logical CPUs:  2
Time       InU   PhysB   MBPS  MB-I  MB-O Rcvdrp Xmtdrp
00:43:18   0.6     0.1    0.0   0.0   0.0      0      0
00:48:18   0.6     0.3    0.0   0.0   0.0      0      0
00:53:19   0.7     0.2    0.0   0.0   0.0      0      0
...
```

Nmon analyzer style output

The **topasout** command generates a Nmon analyzer report that can be viewed with the **nmon** analyzer.

The **topasout** command is used to post process the binary recordings generated by the **xmwlm** utility the **xmtrend** utility and the**topasrec** utility. The binary recording can be the Local System recording, Central Electronic Complex (CEC) recording or Cluster recording. Through SMIT you can enable, configure or disable a binary recording.

Note: The **xmwlm** and **xmtrend** utilities are obsolete and are replaced by the **topasrec** utility.

Use the **topasout** command with the **-a** flag to generate this report. You can open the generated **.csv** file with a **nmon** analyzer. For example, to generate a **xmwlm.061016.csv** file, enter the following command:

```
topasout -a /etc/perf/daily/xmwlm.061016
```

The generated **.csv** file locates in the same directory of the original file, that is, in the **/etc/perf/daily/** directory. The file name is **xmwlm.061016.csv.**

Comma-separated report

The **topasout** command generates a report that contains data that is separated with comma.

Use the **topasout** command with the **-c** flag to generate this report. The output file is written to *recordedfilename*_01 file.

For example, to generate a comma-separated report for the **xmwlm.060503** file, enter the following command:

```
topasout -c /etc/perf/daily/xmwlm.060503
```

The output file is the **xmwlm.060503_01** file which locates in the same directory as the original file.

When you specify the **-m** flag, the **topasout** command writes the *min*, *max*, *mean*, *stdev*, and the *exp* values of the recorded metrics in the report.

The following sample shows the output of a local report with the data separated by commas:

```
#Monitor: xmtrend recording--- hostname: aixfvt19 ValueType: mean
Time="2007/01/24 04:45:50", CPU/gluser=0.02
Time="2007/01/24 04:45:50", CPU/glkern=0.28
Time="2007/01/24 04:45:50", CPU/glwait=0.00
Time="2007/01/24 04:45:50", CPU/glidle=99.69
Time="2007/01/24 04:45:50", NFS/Server/v3calls=0.00
Time="2007/01/24 04:45:50", NFS/Server/v2calls=0.00
…
```

Spreadsheet format report

The **topasout** command generates a report in spreadsheet format.

Use the **topasout** command with **-s** flag to generate this report. The output file is written to *recordedfilename*_01 file.

For example, to generate a report in spreadsheet format for the **xmwlm.060503** file, enter the following command:

```
topasout -s /etc/perf/daily/xmwlm.060503
```

The output file is the **xmwlm.060503_01** file which locates in the same directory as the original file.

When you specify the **-m** flag, the **topasout** command writes the *min*, *max*, *mean*, *stdev*, and the *exp* values of the recorded metrics in the report.

Adapter report

An Adapter report provides information about the amount of data that is read or written to adapters.

The following metrics of the adapter are in the report:

Item	Description
Adapter	Name of the adapter
KBPS	Amount of data transferred (read or written) in the adapter in kilobytes per second
TPS	Number of transfers per second that are issued to the adapter
KB-R	Number of kilobytes read from the adapter
KB-W	Number of kilobytes written to the adapter

The following metrics of the disks are in the report:

Item	Description
Vtargets/Disk	Name of the virtual target device or disk.
Busy%	Percentage of time that the virtual target device or disk is active (bandwidth use for the drive).
KBPS	Number of kilobytes read and written per second over the monitoring interval. This field is the sum of the values of the **KB-R** and **KB-W** metrics.

Item	Description
TPS	Number of transfers per second that are issued to the virtual target device or disk. A transfer is an I/O request to the virtual target device or disk. Multiple logical requests can be combined into a single I/O request to the disk. A transfer is of medium size.
KB-R	Number of kilobytes read per second from the virtual target device or disk.
KB-W	Number of kilobytes written per second to the virtual target device or disk.
AQD	Average number of requests waiting to be sent to the virtual target device or disk.
AQW	Average queue that is waiting per request reported in millisecond. The suffix indicates the unit of time. The default time unit is millisecond.
ART	Average time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond.
AWT	Average time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond.
MRT	Maximum time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond.
MWT	Maximum time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond.

Virtual adapter report

The following metrics of the adapter are reported in the Virtual adapter report:

Item	Description
vAdapter	Name of the adapter.
KBPS	Amount of data transferred (read or written) in the adapter in kilobytes per second.
TPS	Number of transfers per second that are issued to the adapter.
KB-R	Number of blocks received per second from the hosting server to this adapter.
KB-W	Number of blocks sent per second from this adapter to the hosting server.
AQD	Number of requests waiting to be sent to adapter.
AQW	Time spent by a transfer request in the wait queue. Reported in millisecond. The suffix indicates the unit of time. The default time unit is millisecond.
ART	Time to receive a response from the hosting server for the read request sent The suffix indicates the unit of time. The default time unit is millisecond.
AWT	Time to receive a response from the hosting server for the write request sent The suffix indicates the unit of time. The default time unit is millisecond.
MRT	Maximum time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond.

Item	Description
MWT	Maximum time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond.

The following metrics of the disks are in the report:

Item	Description
Vtargets/Disk	Name of the virtual target device or disk.
Busy%	Percentage of time that the virtual target device or disk is active (bandwidth use for the drive).
KBPS	Number of kilobytes read and written per second over the monitoring interval. This field is the sum of the values of the **KB-R** and **KB-W** metrics.
TPS	Number of transfers per second that are issued to the virtual target device or disk. A transfer is an I/O request to the virtual target device or disk. Multiple logical requests can be combined into a single I/O request to the disk. A transfer is of medium size.
KB-R	Number of kilobytes read per second from the virtual target device or disk.
KB-W	Number of kilobytes written per second to the virtual target device or disk.
AQD	Average number of requests waiting to be sent to the virtual target device or disk.
AQW	Average queue that is waiting per request reported in milliseconds. The suffix indicates the unit of time. The default time unit is millisecond.
ART	Average time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond.
AWT	Average time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond.
MRT	Maximum time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond.
MWT	Maximum time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond.

On Demand WLE input from topasrec / nmon recordings

In addition to weekly peak inputs for WLE through SMIT, the user can invoke an On Demand WLE input file to study a particular workload and use that data to size the systems and generate reports. The **topasout** command has the capability to study a particular **topas** or **nmon** recording and generate WLE readable reports in xml format using this option.

Use topasout -R wle *-Oifile=<filename>* option to generate the WLE report. For example, to generate a report from file, use the following command.

```
topasout –R wle -Oifile=/etc/perf/daily/xmwlm_130504.topas
```

If it is **nmon** recording, specify the *-Otype* option along with the *-Oifile* option as shown below:

```
topasout -R wle -Oifile=/etc/perf/daily/xmwlm_130504.nmon -Otype=nmon
```

The *wle* option is different from the other types of **–R** in a way that both **topas** and **nmon** recordings can be given as an input to this option while only **topas** recordings (recordings generated through **xmwlm** and **topasrec**) can be given as an input file for the other options.

CEC reports

There are five types of CEC reports: the Summary report, the Detailed report, the Shared processor pool report, the Comma-separated report, and the Spread-sheet report.

Summary report

This report provides a summary of the CEC system. The reporting is based on the partitions that actually responded to the **topas** command. If the partitions in the CEC do not have the **xmtopas** or **xmservd** configured, the partitions cannot be monitored.

A CEC summary report contains the following column headings:

Header (partition details):

Item	Description
Mon	Number of the partitions that are monitored in the first reporting time interval
UnM	Number of the partitions that are not monitored in the first reporting time interval
Shr	Number of the shared partitions in the first reporting time interval
Ded	Number of the dedicated partitions in the first reporting time interval
Cap	Number of the capped partitions in the first reporting time interval
UnC	Number of the uncapped partitions in the first reporting time interval

CEC:

Item	Description
ShrB	Shared physical processor busy. (Sum of physical busy of processors in the shared partitions.)
DedB	Dedicated physical processor busy. (Sum of physical busy of processors in the dedicated partitions.)
Don	Total number of the processors that are donated to the physical pool.

Processors:

Item	Description
Mon	Number of the physical processors that are monitored
UnMon	Number of the physical processors that are not monitored.
Shr	Number of the processors in shared partitions
Ded	Number of the processors in dedicated partitions
PSz	Number of the active shared processors in the physical pool
APP	Available physical processors in the pool

Memory (GB):

Item	Description
Mon	Total memory of the monitored partitions
UnM	Total memory of the partitions that are not monitored
Avl	Memory available to partitions

Item	Description
InUse	Memory in used in the monitored partitions
UnA	Memory that is not available for partitions

The following sample shows the output of a CEC Summary report:

```
Sample Output
#Report: CEC Summary  --- hostname: ptoolsl3                    version:1.2
Start:02/22/07 00:44:06   Stop:02/22/07 23:59:06  Int: 5 Min  Range:1395 Min
Partition Mon:  3  UnM:  0  Shr:  1  Ded:  2  Cap:  2  UnC:  1
       -CEC---------------  -Processors---------------- -Memory (GB)-----------
Time   ShrB  DedB  Don  Stl  Mon  UnM  Shr Ded  PSz  APP Mon  UnM  Avl  UnA InU
00:49  0.00  0.00   -    -   2.2  0.0  0.2   2  2.0  2.0 9.4  0.0  8.0  0.0 1.0
00:54  0.00  0.00   -    -   2.2  0.0  0.2   2  2.0  2.0 9.4  0.0  8.0  0.0 1.0
00:59  0.00  0.00   -    -   2.2  0.0  0.2   2  2.0  2.0 9.4  0.0  8.0  0.0 1.0
```

Detailed report

A CEC Detailed report gives a detailed view of all the partitions that the **topas** command is able to record data from.

The followings column headings are in a CEC Detailed report:

Partition Info:

Item	Description
Monitored	Number of partitions that are monitored
Unmonitored	Number of partitions that are not monitored
Shared	Number of shared partitions
Uncapped	Number of uncapped shared partitions
Capped	Number of capped shared partitions
Dedicated	Number of dedicated partitions
Donating	Number of partitions that are donating

Memory:

Item	Description
Monitored	Total memory that is monitored
UnMonitored	Total memory that is not monitored
Available	Total memory that is available
UnAllocated	Total memory that is not allocated to any partition
Consumed	Total memory that is consumed by the partitions

Processor:

Item	Description
Monitored	Number of physical processors that are monitored
UnMonitored	Number of physical processors that are not monitored
Available	Number of physical processors that are available in the CEC system
UnAllocated	Number of physical processors that are not allocated to any partition
Shared	Number of processors in shared partitions

Item	Description
Dedicated	Number of processors in dedicated partitions
Donated	Sum of the number of processors in all of the partitions that are currently donating
Pool Size	Number of active shared processors in the physical pool
Avail Proc Pool	Available physical processors in pool. This is the idle cycles in the pool reported as a number of processors
Shr Physical Busy	Sum of the busy physical processors of all of the shared partitions
Ded Physical CPUs	Sum of the busy physical processors of all of the dedicated partitions
Donated Phys. CPUs	Sum of the donated processor cycles (reported as a number of processors) from all partitions
Stolen Phys. CPUs	Sum of the stolen processor cycles (reported as a number of processors) from all partitions
Virtual Pools	Number of the virtual pools
Virt. Context Switch	Total number of the virtual context switches per second in the monitoring interval
Phantom Interrupts	Total number of the phantom interrupts per second in the monitoring interval

Individual partition data:

Item	Description
Host	Host name
OS	Operating system level
M	The **M** column heading represents the mode. In shared partitions, it displays the following attributes: • C- SMT is enabled and capped • c- SMT is disabled and capped • U- SMT is enabled and uncapped • u- SMT is disabled and uncapped In dedicated partitions, it displays the following attributes: • S- SMT is enabled and is not donating • d- SMT is disabled and donating • D- SMT is enabled and donating
Mem	Total memory in gigabytes
InU	Memory in used in gigabytes
Lp	Number of logical processors
Us	Percentage of processor that is used by programs executing in the user mode
Sy	Percentage of processor that is used by programs executing in kernel mode
Wa	Percentage of time that is spent waiting for I/O
Id	Percentage of time that the processor is idle

Item	Description
PhysB	Number of physical processors that are busy
Ent	Entitlement granted (shared only)
%Entc	Percentage of entitlement consumed (shared only)
Vcsw	Virtual context switches average per second (shared only)
PhI	Phantom interrupts average per second (shared only)
%idon	Percentage of physical processor that is used while explicitly donating idle cycles. This metric is applicable only for donating dedicated partitions.
%bdon	Percentage of physical processor that is used while busy cycles are being donated. This metric is applicable only for donating dedicated partitions
%istl	Percentage of physical processor that is used while busy cycles are being stolen by the hypervisor. This metric is applicable only for dedicated partitions

The following sample shows the output of a CEC Detailed report:

```
#Report: CEC Detailed --- hostname: ptoolsl3                  version:1.2
Start:03/06/07 07:19:39   Stop:03/06/07 07:28:39  Int: 5 Min  Range:   9 Min

Time: 07:24:38 ----------------------------------------------------------------
Partition Info   Memory (GB)        Processors         Avail Pool :  2.0
Monitored  :  3  Monitored  :  9.4  Monitored  :  2.2  Shr Physcl Busy:  0.01
UnMonitored:  0  UnMonitored:  0.0  UnMonitored:  0.0  Ded Physcl Busy:  0.01
Shared     :  1  Available  :  0.0  Available  :  0.0  Donated Phys. CPUs: 0.00
UnCapped   :  1  UnAllocated:  0.0  Unallocated:  0.0  Stolen Phys. CPUs : 0.00
Capped     :  2  Consumed   :  0.0  Shared     :  0.2  Hypervisor
Dedicated  :  2                     Dedicated  :  2.0  Virt Cntxt Swtch:   545
Donating   :  0                     Donated    :  0    Phantom Intrpt  :     0
                                    Pool Size  :  2.0

Host         OS  M  Mem  InU Lp Us Sy Wa Id PhysB  Vcsw  Ent  %EntC PhI
-------------------------------------shared-------------------------------------
ptoolsl1     A53 U  3.1  1.9  4  0  1  0 98  0.01   317  0.2   2.55   0

Host         OS  M  Mem  InU Lp Us Sy Wa Id PhysB  Vcsw  %istl %bstl
-----------------------------------dedicated----------------------------------
ptoolsl3     A54    3.1  0.9  2  0  0  0 99  0.00   228    -     -
ptoolsl1     A52    3.1  2.7  1  0  1  0 99  0.01     0    -     -

Time: 07:28:39 ----------------------------------------------------------------
```

Shared-processor-pool report

The CEC Shared-processor-pool report contains information about the shared processor pools.

The following column headings are included in a Shared-processor-pool report:

Item	Description
psize	Effective maximum capacity of the pool.
entc	Entitled capacity of the pool.
maxc	Maximum capacity of the pool.
physb	Sum of the physical busy of processors in the shared partitions of a pool. (The "physical busy" refers to fraction of physical processors that are busy.)
app	Available physical processors in the pool.
mem	Sum of the monitored memory for all of the shared partitions in the pool.

Item	Description
muse	Sum of the memory consumed by all of the shared partitions in the pool.

The following sample shows the output of a CEC Shared-processor-pool report:

```
Sample Output

#Report: Topas CEC Pool Detailed --- hostname: ptoolsl1               version: 1.0
pool  psize entc  maxc physb app   mem  muse
0     3.0   2.0   3.0  0.1   1.0   2.0  1.0
1     4.0   3.0   5.0  0.5   1.5   1.0  0.5
2     3.0   2.5   4.0  0.2   2.0   1.0  0.5

Host       Pi OS  M Mem InU Lp  Us Sy Wa Id  PhysB Vcsw Ent  %EntC PhI
-------------------------------shared-------------------------------------------
ptools1    0  53 U  11   9  2  11  13  0 75  0.10  121 0.25   0.3   3

ptools5    1  53 U  12  10  2  12  3  0 85   0.20  121 0.25   0.3   3
ptools3    1  53 C 5.0 2.6  2  10  1  0 89   0.15   52 0.25   0.3   2

ptools7    2  53 c 2.0 0.4  1   0  1  0 99   0.05    2 0.10   0.3   2

Host       OS  M Mem InU Lp  Us Sy Wa Id  PhysB Vcsw %istl %bstl %bdon %idon
-----------------------------dedicated-----------------------------------------
ptools6    52   1.1 0.1  1  11  7  0 82   0.50   50   10      5   10   0
ptools8    52   1.1 0.1  1  11  7  0 82   0.50   60    0      1    -   -
ptools2    52   1.1 0.1  1  11  7  0 82   0.50  200    0      15  25   10
```

Memory pool report

The **topasout** command generates the Memory pool report that contains information about the memory pools in the CEC and the partitions that belong to the memory pools. The following values are displayed in the header section:

Item	Description
Mshr	Number of LPAR that are running in the shared-memory mode
Mded	Number of LPAR that are running in the dedicated-memory mode
Pools	Total number of memory pools in the system
Mpsz	Total size of physical memory of all the memory pools in gigabytes
MPuse	Total memory used by LPAR associated with all the pools in gigabytes
Entl	Total I/O memory entitlement of all of the LPAR in all of the pools in gigabytes
Use	Total I/O memory entitlement in use of all of the LPAR in all of the pools in gigabytes
Mon	Total monitored memory of the system in gigabytes
InUse	Total memory in use of the system in gigabytes
Avl	Total free memory available in the system in gigabytes

The following values are displayed in the memory pools section:

Item	Description
mpid	ID of the memory pool
mpsz	Size of the total physical memory of the memory pool in gigabytes
mpus	Total memory of the memory pool in use (this value is the sum of the physical memory allocated to all of the LPAR in the pool)
mem	Aggregate logical memory size of all of the partitions in the pool in gigabytes

Item	Description
memu	Aggregate logical memory used for all of the partitions in the pool in gigabytes
iome	Aggregate of I/O memory entitlement that is configured for all of the LPAR in the pool in gigabytes
iomu	Aggregate of the I/O memory entitlement that is used for all of the LPAR in the pool in gigabytes
hpi	Aggregate number of hypervisor page faults that have occurred for all of the LPAR in the pool
hpit	Aggregate amount of time spent waiting for hypervisor page-ins by all of the LPAR in the pool in milliseconds

The following values are displayed in the partitions section:

Item	Description
mem	Logical memory size of the partition in gigabytes
memu	Logical memory that is used for the partition in gigabytes
meml	Logical memory that is loaned to the hypervisor by the LPAR
pmem	Physical memory allocated to the partition from the memory pool in gigabytes
iom	Amount of I/O memory entitlement that is configured for the LPAR in gigabytes
iomu	Amount of I/O memory entitlement that is used for the LPAR in gigabytes
hpi	Number of hypervisor page faults
hpit	Time spent waiting for hypervisor page-ins in milliseconds
vcsw	Virtual context switches as an average per second
physb	Physical processor busy
%entc	Percentage of the processor entitlement that is consumed

Comma-separated reports

The **topasout** command generates a CEC report that contains data that are separated with comma.

Use the **topasout** command with the **-c** flag to generate this report. The output file is written to *recordedfilename*_01 file.

For example, to generate a report in spreadsheet format for the **topas_CEC.070221** file in the **/etc/perf/** directory, enter the following command:

```
topasout -c /etc/perf/topas_CEC.070221
```

The output file is the **topas_CEC.070221_01** file, which locates in the same directory as the original file.

The **topas** recordings support only the**-m mean** option.

The following sample shows the output of a **topas_CEC** report:

```
#Monitor: topas_CEC recording--- hostname: ptoolsl3 ValueType: mean
Time="2007/03/06 07:19:39", CEC/Lpars/monitored=3.00
Time="2007/03/06 07:19:39", CEC/Lpars/unmonitored=0.00
Time="2007/03/06 07:19:39", CEC/Lpars/shared=1.00
Time="2007/03/06 07:19:39", CEC/Lpars/dedicated=2.00
Time="2007/03/06 07:19:39", ptoolsl1/LPAR/Sys/osver=5.30
Time="2007/03/06 07:19:39", ptoolsl1/LPAR/Sys/shared=1.00
Time="2007/03/06 07:19:39", ptoolsl1/LPAR/Sys/capped=0.00
```

```
Time="2007/03/06 07:19:39", ptoolsl1/LPAR/Sys/smt=1.00
…
```

Spreadsheet format reports

The **topasout** command generates a CEC report in spreadsheet format.

Use the **topasout** command with the **-s** flag to generate this report. The output file is written to *recordedfilename*_01 file.

For example, to generate a report in spreadsheet format for the **topas_CEC.070221** file in the **/etc/perf/** directory, enter the following command:

```
topasout -s /etc/perf/topas_CEC.070221
```

The output file is the **topas_CEC.070221_01** file, which locates in the same directory as the original file.

The **topas** recordings can use only the **-m mean** option.

VIOS report

The VIOS report contains information about Virtual I/O Server/Client throughput. The following column headings are included in a Virtual I/O Server/Client throughput report:

Item	Description
Server	Name of the VIO Server.
Client	Name of the VIO Client.
KBPS	Number of kilobytes read and written per second over the monitoring interval. This field is the sum of the values of the **KB-R** and **KB-W** metrics.
TPS	Number of transfers that are issued per second.
KB-R	Number of kilobytes read per second.
KB-W	Number of kilobytes written per second.
AQD	Average number of requests waiting to be sent.
AQW	Average queue that is waiting per request reported in millisecond. The suffix indicates the unit of time. The default time unit is millisecond.
ART	Average time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond.
AWT	Average time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond.
MRT	Maximum time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond.
MWT	Maximum time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond.

VIOS adapter report

The VIOS adapter report contains information on virtual I/O server or client (VIOS) adapter and disk details. The following details on the disks are reported:

Item	Description
Adapter	Name of the server adapter.

Item	Description
Vtargets	Name of the virtual target device belonging to the server adapter.
Client_disk	Name of the client disk that is mapped to the virtual target device of the server adapter.

The following details of the adapters are displayed:

Item	Description
KBPS	Amount of data transferred (read or written) in the adapter in kilobytes per second.
TPS	Number of transfers per second issued to the adapter.
KB-R	Total number of kilobytes read from the adapter.
KB-W	Total number of kilobytes written to the adapter.
AQD	Average number of requests waiting to be sent to the virtual target device or disk.
AQW	Average queue waiting per request reported in millisecond. The suffix indicates the unit of time. The default time unit is millisecond.
ART	Average time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond.
AWT	Average time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond.
MRT	Maximum time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is millisecond.
MWT	Maximum time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is millisecond.

The following details of the virtual target device and the client disk are reported:

Item	Description
Busy%	Percentage of time that the virtual target device or disk is active.
KBPS	Number of kilobytes read and written per second over the monitoring interval. This field is the sum of the value of the **KB-R** and **KB-W** metrics.
TPS	Number of transfers per second that are issued to the virtual target device or disk. A transfer is an I/O request to the virtual target device or disk. Multiple logical requests can be combined into a single I/O request to the disk. A transfer is of medium size.
KB-R	Number of kilobytes read per second from the virtual target device or disk.
KB-W	Number of kilobytes written per second to the virtual target device or disk.
AQD	Average number of requests waiting to be sent to the virtual target device or disk.
AQW	Average queue waiting per request that is reported in milliseconds. The suffix indicates the unit of time. The default time unit is milliseconds.

Item	Description
ART	Average time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is milliseconds.
AWT	Average time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is milliseconds.
MRT	Maximum time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is milliseconds.
MWT	Maximum time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is milliseconds.

Flags

Item	Description
-a	The **-a** flag is used only for **nmon** analyzer report.
-b *time*	The time in the recorded file that the **topasout** command begins to generate reports from. The time can either be in the YYMMDDHHMM format or the HHMM format. You must use the same time format for end time if it is specified. YYMMDD represents year, month, and day. HHMM represents hour and minute. In HHMM format, the value must range from 0000 through 2359. The default value for begin time is 0000. The report is generated for the first day of the recording within the given time range. In YYMMDDHHMM format, the default value is the time of the first recorded data in the recording file. The command generates report for the data between the begin and end time range.
-c	Specifies that the **topasout** command should format the output files as comma-separated ASCII. Each line in the output files contains one time stamp and one observation.
-e *time*	The time in the recorded file that the **topasout** command stop generating reports from. The time can be in the YYMMDDHHMM format or the HHMM format. You must use the same time format for the begin time if it is specified. YYMMDD represents year, month, and day. HHMM represents hour and minute. In YYMMDDHHMM format, the default value is the time of the last recorded data in the recording file. The report is generated for the data between the begin and end date and time range. In HHMM format, the default value for end time is 2359. The report is generated for the first day of the recording within the given time range.
-i *interval*	The **-i** flag defines the interval in minute that the **topasout** command need to average the values. The valid values of the **-i** flag are 5, 10, 15, 30, or 60. The default value is 5 minutes.

Item	Description
-m *type*	By default, the **topasout** only outputs the **mean** values. Other recorded values and the full set for local recordings are available through other options including the *min*, *max*, *mean*, *stdev*, *set*, and *exp* options.
-O	The **-O** flag can have the following values: fullhostname=[on|off]. When the **-O** flag is set to on, it displays the full hostname in a new column. The default value is off for the fullhostname.
-R *type*	Use the **-R** flag to specify the type of a report for **xmwlm** recordings or **topasout** recordings. The *type* parameter has the following variables: **summary** Generates Summary report. **detailed** Generates Detailed report. **lan** Generates LAN report. **disk** Generates Disk report. **poolinfo** Generate Shared-processor-pool report. **mempool** Generates memory pool report. If there is no memory pool, the header will be displayed without any values. **adapter** Generates adapter report. **vadapter** Generates virtual adapter report. **vios** Generates Virtual I/O Server/Client throughput report. **vios_adapter** Generates Virtual I/O Server/Client adapter and disk detailed report. The reports generated with the **-R** flag are printed to the console.
-s	Specifies that `topasout` should format the output files in a format suitable for input to spreadsheet programs.

Parameters

Item	Description
xmwlm_recording_file	Specifies that the input file is a recording created using the **topasrec/ xmwlm** command.
topas_recording_file	Specifies that the input file is a recording created using the **topasrec/ topas** command.
nmon_recording_file	Specifies that the input file is a recording created using the **nmon** command.

Examples

1. To generate a Detailed report from an **xmwlm** recording file from 10:00 a.m. to 11:00 p.m., enter the following command:

```
topasout -R detailed  -i 15 –b 1000 -e  2300 /etc/perf/daily/xmwlm.070226
```

2. To generate a Summary report from an **xmwlm** recording file, enter the following command:

```
topasout -R summary /etc/perf/daily/xmwlm.070226
```

3. To generate a Disk report from an **xmwlm** recording file, enter the following command:

```
topasout -R disk  /etc/perf/daily/xmwlm.070226
```

4. To generate a LAN report from an **xmwlm** recording file, enter the following command:

```
topasout -R lan  /etc/perf/daily/xmwlm.070226
```

5. To generate an adapter report from an **xmwlm** recording file, enter the following command:

```
topasout -R adapter  /etc/perf/daily/xmwlm.070226
```

6. To generate a virtual adapter report from an **xmwlm** recording file, enter the following command:

```
topasout -R vadapter  /etc/perf/daily/xmwlm.070226
```

7. To generate a **nmon** analyzer report from an **xmwlm** recording file named **xmwlm.070226** in the **/etc/perf/daily/** directory, enter the following command:

```
topasout  –a /etc/perf/daily/xmwlm.070226
```

   The output is written to **/etc/perf/daily/xmwlm.070226.csv**

8. To generate a Shared-processor-pool report from **topas CEC** recording, enter the following command:

```
topasout -R poolinfo /etc/perf/topas_CEC.070302
```

9. To generate a Summary report from **topas CEC** recording from 2:00 p.m. to 4:00 p.m. on the first day of recorded data, enter the following command:

```
topasout -R summary  -b 1400 –e 1600 /etc/perf/topas_CEC.070302
```

10. To generate a VIOS report from a **topas CEC** recording, enter the following command:

```
topasout -R vios  /etc/perf/topas_CEC.070302
```

11. To generate a VIOS adapter report from a **topas CEC** recording, enter the following command:

```
topasout -R vios_adapter  /etc/perf/topas_CEC.070302
```

12. To generate a memory pool report from a **topas CEC** recording, enter the following command:

```
topasout -R mempool /etc/perf/topas_CEC.070302
```

13. To generate a summary report from a **topas CEC** recording from 2:00 p.m., March 10, 2008 to 4:00 p.m., March 12,2008, enter the following command:

```
topasout -R summary -b 0803101400 -e 0803121600 /etc/perf/ptoolsl1_cec_080310.topas
```

14. To generate a detailed report from a **topas Cluster** recording from 2:00 p.m., March 10, 2008 to 4:00 p.m., March 12,2008, enter the following command:

```
topasout -R summary -b 0803101400 -e 0803121600 /etc/perf/ptoolsl1_cluster_080310.topas
```

15. To generate a **nmon** analyzer report from an **CEC Recording** file named ptoolsl1_cec_080310.topas in the **/etc/perf/** directory enter the following command:

```
topasout  -a /etc/perf/ptoolsl1_cec_080310.topas
```

16. To generate a **nmon** analyzer report from an **Cluster Recording** file named ptoolsl1_cluster_080310.topas in the **/etc/perf/** directory, enter the following command:

```
topasout  -a /etc/perf/ptoolsl1_cluster_080310.topas
```

17. To report the full hostname in the detailed reported, enter the following command:

```
topasout -R detailed -O fullhostname=on
```

Location

/usr/bin/topasout

Files

Item	Description
`/usr/bin/topas`	Contains the **topas** command.
/usr/bin/xmwlm	Contains the **xmwlm** command.
`/usr/bin/topasout`	Contains the `topasout` command. The `topasout` command is included in the **perfagent.tools** fileset.

topasrec Command

Purpose

The **topasrec** command generates binary recording of the local system metrics, CEC (Central Electronic Complex) metrics, and Cluster metrics.

Note: The **xmwlm** and **xmtrend** utilities are obsolete and are replaced by **topasrec** command.

Syntax

Local binary recording:

topasrec -L [**-c** *sample_count*] [**-o** < *output_filename* >] [**-s** *seconds*] [**-t** *trace level*]

Local Azizo recording:

topasrec -L -O type=azizo

CEC recording:

topasrec -C [**-c** *sample_count*] [**-o** < *output_filename* >] [**-s** *seconds*] [**-O xmtopas=<hostname>**]

Cluster recording:

topasrec -G [**-c** *sample_count*] [**-o** < *output_filename* >] [**-s** *seconds*] [**-O xmtopas=<hostname>**]

List running recording:

topasrec -l

Description

Note:

1. You cannot run the **topasrec** command inside a workload partition (WPAR).

2. The CEC or cluster recording re-spawns after the partition migration or hibernation is complete. The active recording file is renamed to **<current_file_name>.mig.<HH>.<MM>.<SS>** after migration of the partition, and **<current_file_name>.hib.<HH>.<MM>.<SS>** after hibernation of the partition.

The **topasrec** command records the local system data, the cross-partition data (CEC statistics), and the cluster data in binary format.

When you run the **topasrec** command for a CEC recording, the **topasrec** command collects a set of metrics from the AIX partitions running on the same CEC. The **topasrec** command collects dedicated and shared partition data, and a set of aggregated values to provide an overview of the partition set on the same CEC.

The **topasrec** command finds metrics to be recorded from the **/usr/lpp/perfagent/daily.cf** file, and you should not alter the **daily.cf** file. Altering the **daily.cf** file affects the following recording files:

1. Persistent/nonpersistent local recordings
2. WLE Recording
3. Performance management service data collection
4. Performance PMR **(perfpmr)** data collected for performance problem analysis

The nmon, CEC, and cluster recordings are not affected by altering the **daily.cf** file. If you wants to have a reduced subset of metrics for recording, you can back up the existing **daily.cf** file, and alter it to remove the metrics that you do not want to record. Removing these metrics affects all the recording files previously listed. For example, if you do not want **Disk/*/busy** metrics to be recorded by using the **topasrec** command, you can remove this line from the **/usr/lpp/perfagent/daily.cf** file.

Note: For any dynamic configuration changes to the system, the tool has to be restarted to reflect the new changes.

Flags

Item	Description
-C	Records CEC statistics in binary format. The **-C** flag specifies that the cross-partition statistics are to be recorded.
-c *sample_count*	Records the specified number of records and then stops. If the **-c** flag is not specified, or if the value of the *sample_count* parameter is zero, the recording is continuous and the **topasrec** command writes to the recording file until it is stopped.
-L	Records local statistics in binary format.
-l	Lists the recordings that are running.
-s *seconds*	Specifies the recording interval in seconds. The value of the *seconds* parameter should be a multiple of 60. For continuous recordings (**topasrec -c 0**) of CEC and local statistics, the default value of recording interval is 900 seconds. For a sample count that is greater than zero, the default value of the recording interval is 300 seconds.
-O *xmtopas=<hostname>*	Specifies the name of the host that aggregates the data and provides it to **topasrec**. If this is not specified, **topasrec** will get data from one of the known aggregators. **Note:** You cannot use the override option with persistent recording.

Item	Description
-o < *output_filename* >	Specifies the name of the output file. The value of the *output_filename* parameter can be a directory with an optional file prefix. You can specify one of the following types of file names to the *output_filename* parameter: • A directory. The directory should always end with **/**. For example, the **/etc/perf/** directory. • A directory with a file name. For example, the **/home/tester/perf_load** file. • A file name. For example, the **perf_load** file. The default output file is the current directory (**./**). In CEC recording, Cluster Recording and local recording, the default prefix of the file name is the host name. If you provide a file name that contains a directory and a file name prefix in the **-o** *output_filename* flag, the name of the recorded file is in the following format: • For CEC metrics, the output is in the following format: `<filename>_cec_YYMMDD_HHMM.topas` • For Cluster metrics, the output is in the following format: `<filename>_cluster_YYMMDD_HHMM.topas` • For local metrics, the output is in the following format: `<filename>_YYMMDD_HHMM.topas` If you provide a file name that contains only the directory prefix, the name of the recorded file is in the following format: • For CEC metrics, the output is in the following format: `<filename/hostname>_cec_YYMMDD_HHMM.topas` • For Cluster metrics, the output is in the following format: `<filename/hostname>_cluster_YYMMDD_HHMM.topas` • For local metrics, the output is in the following format: `<filename/hostname>_YYMMDD_HHMM.topas` In these formats, year (YY), month (MM), day (DD), hour (HH), and minute (MM) correspond to the time when the recording file is created. **Note:** For CEC/Cluster Recording , if **xmtopas** override option is used then filename will be the value specified for xmtopas=<value>. **Example**: `< value>_cec_YYMMDD_HHMM.topas` `< value>_cluster_YYMMDD_HHMM.topas`
-r *retention*	Specifies the number of days for which the file must be retained. The minimum value is 1. For example, **`-r 5`** specifies that the file is retained for five days.

Item	Description
-R *max_days_per_file*	Specifies the number of days for which the performance data needs to be written to a file. The minimum value is 1 and maximum value is 366. For example, if we start a persistent recording with option **-R 2** on day 1, the performance data of day 1 and day 2 are written to the same file. On day 3, a new file is created that contains the performance data of day 3 and day 4.
-t *trace level*	Specifies the trace level. The trace level can be set from 1 to 9.

Parameters

Item	Description
sample_count	Specifies the number of records to generate.
output_filename	Specifies the name of the output file.
seconds	Specifies the recording interval in seconds.

Examples

1. To start a local binary recording that runs for 5 minutes and contains system metrics every 1 minute, enter the following command:

   ```
   topasrec -L -c 5 -s 60
   ```

   If the file is created at `23:14, Mar 10, 2008`, and the host name is `ses15`, then the output file name is `./ses15_080310_2314.topas`.
2. To start a continuous local binary recording with a `/home/test/sample` file name, enter the following command:

   ```
   topasrec -L -o /home/test/sample
   ```

   If the file is created at `12:05, Mar 10, 2008`, and the host name is `ses15`, then the output file name is `/home/test/sample_080310_1205.topas`.
3. To start a CEC recording that runs for `20` minutes with metrics recorded at `120`-second intervals, and generate an output file named `sample`, enter the following command:

   ```
   topasrec -C -o sample -s 120 -c 10
   ```

   If the file is created at `08:07, Feb 1, 2008`, and the host name is `ses15`, then the output file name is `./sample_cec_080201_0807.topas`.
4. To start a continuous local binary recording with a `/home/test/sample_bin` file name, enter the following command:

   ```
   topasrec -C -o /home/test/sample_bin
   ```

   If the file is created at `04:20, Feb 1, 2008`, and the host name is `ses15`, then the output file name is `/home/test/sample_bin_080201_0420.topas`.
5. To list the details of the running recordings, enter the following command:

   ```
   topasrec –l
   ```

6. To enable trace, enter the following command:

   ```
   topasrec –L –t 1
   ```

7. To start a Cluster recording that runs for 20 minutes with metrics recorded at 120-second intervals, and generate an output file named sample, enter the following command:

```
topasrec -G -o sample -s 120 -c 10
```

   If the file is created at `08:07, Feb 1, 2008` and the host name is `ses15` then the output file name is `/sample_cluster_080201_0807.topas.`.
8. To start a continuous local Cluster recording with a `/home/test/sample_bin` file name, enter the following command:

```
topasrec -G -o /home/test/sample_bin
```

9. To manually start a local azizo recording, enter the following command:

```
topasrec -L -O type=azizo
```

   If a valid `/etc/perf/xmtopas.cf` file is present, the azizo recording is automatically started by the **xmtopas** command. After the recording is started, it generates the azizo.<yymmdd> file in the `/etc/perf/` directory and runs only if the **xmtopas** command is running

Files

Item	Description
/usr/bin/topasrec	Contains the **topasrec** command.

topsvcs Command

Purpose

Starts or restarts topology services on a cluster node.

Syntax

`topsvcs`

Description

Use `topsvcs` script to start the operation of topology services for a cluster.

The `topsvcs` script is not normally executed from the command line. It is normally called by the `topsvcsctrl` control script, which is in turn called by the HACMP/ES startup process.

The `topsvcs` script issues these commands:

```
no -o nonlocsrcroute=1
no -o ipsrcroutesend=1
no -o ipsrcrouterecv =1
no -o ipsrcrouteforward=1
```

These commands enable IP source routing. Do not change this setting, because the topology services subsystem requires this setting to work properly. If you change the setting, the topology services subsystem and a number of other subsystems that depend on it will no longer operate properly.

Flags

-s
Instructs the topology services daemon to reject messages that are apparently delayed.

-d
Instructs the topology services daemon not to reject messages that are apparently delayed (this is the default).

Security

You must have `root` privilege to run this command.

Exit Status

0
Indicates the successful completion of the command.

1
Indicates the command was unsuccessful.

Environment Variables

HB_SERVER_SOCKET
This environment variable should be set before this command can be executed. It must be set to the location of the UNIX-domain socket used by topology services clients to connect to the topology services daemon. This environment variable must be set to `/var/ha/soc/hats/server_socket.`*partition name*.

HA_SYSPAR_NAME
If HB_SERVER_SOCKET is not set, then HA_SYSPAR_NAME must be set to the partition name.

Restrictions

This command is valid in an HACMP environment only.

Use this command *only* under the direction of the IBMSupport Center.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

This command writes error messages (as necessary) to standard error.

Examples

To instruct the topology services daemon on the local node to start discarding apparently delayed messages, enter:

```
export HA_SYSPAR_NAME=partition1

/opt/rsct/bin/hatsoptions -s
```

Location

/opt/rsct/bin/topsvcs
Contains the `topsvcs` script

Files

`/var/ha/soc/hats/server_socket.`*partition name*

topsvcsctrl Command

Purpose

Starts the topology services subsystem.

Syntax

`topsvcsctrl { -a | -s | -k | -d | -c | -u | -t | -o | -r | -h }`

Description

The `topsvcsctrl` control script controls the operation of the topology services subsystem. The subsystem is under the control of the system resource controller (SRC) and belongs to a subsystem group called `topsvcs`. This script is normally started by the HACMP/ES startup process.

An instance of the topology services subsystem runs on every node of a cluster.

From an operational point of view, the topology services subsystem group is organized as follows:

Subsystem
topology services

Subsystem group
`topsvcs`

SRC subsystem
`topsvcs`

The `topsvcs` subsystem is associated with the `hatsd` daemon and the `topsvcs` script. The `topsvcs` script configures and starts the `hatsd` daemon. The subsystem name on the nodes is `topsvcs`. There is one of each subsystem per node and it is associated with the cluster to which the node belongs.

Daemons
`hatsd`

Provides the topology services. The `topsvcs` script configures and starts the `hatsd` daemon.

The `topsvcsctrl` script is not normally executed from the command line. It is normally called by the HACMP/ES startup command.

The `topsvcsctrl` script provides a variety of controls for operating the topology services subsystems:

- Adding, starting, stopping, and deleting the subsystems
- Cleaning up the subsystems, that is, deleting them from all system partitions
- Turning tracing on and off
- Refreshing the subsystem

Before performing any of these functions, the script obtains the current cluster name (using the `cllsclstr` command) and the node number (using the `clhandle` command). If the node number is `0`, the control script is running on the control workstation.

Except for the clean and unconfigure functions, all functions are performed within the scope of the current system partition.

Adding the subsystem: When the `-a` flag is specified, the control script uses the `mkssys` command to add the topology services subsystem to the SRC. The control script operates as follows:

1. It makes sure the `topsvcs` subsystem is stopped.
2. It removes the `topsvcs` subsystem from the SRC (in case it is still there).
3. It adds the `topsvcs` subsystem to the SRC.

Starting the subsystem: When the `-s` flag is specified, the control script uses the `startsrc` command to start the topology services subsystem, `topsvcs`.

Stopping the subsystem: When the `-k` flag is specified, the control script uses the `stopsrc` command to stop the topology services subsystem, `topsvcs`.

Deleting the subsystem: When the `-d` flag is specified, the control script uses the `rmssys` command to remove the topology services subsystem from the SRC. The control script operates as follows:

1. It makes sure that the `topsvcs` subsystem is stopped.
2. It removes the `topsvcs` subsystem from the SRC using the `rmssys` command.
3. It removes the port number from the `/etc/services` file.

Cleaning up the subsystems: When the `-c` flag is specified, the control script stops and removes the topology services subsystems for all clusters partitions from the SRC. The control script operates as follows:

1. It stops all instances of subsystems in the clusters, using the `stopsrc -g topsvcs` command.
2. It removes all entries for the `topsvcs` subsystem from the `/etc/services` file.

Turning tracing on: When the `-t` flag is specified, the control script turns tracing on for the `hatsd` daemon, using the `traceson` command.

Turning tracing off: When the `-o` flag is specified, the control script turns tracing off (returns it to its default level) for the `hatsd` daemon, using the `tracesoff` command.

Refreshing the subsystem: When the `-r` flag is specified, the control script refreshes the subsystem, using the `topsvcs refresh` command and the `refresh` command. It rebuilds the information about the node and adapter configuration in the global object data manager (ODM) and signals the daemon to read the rebuilt information.

Logging: While it is running, the topology services daemon (`hatsd`) provides information about its operation and errors by writing entries in a log file called `/var/ha/log/topsvcs.`*cluster_name*.

Flags

Item	Description
`-a`	Adds the subsystem.
`-s`	Starts the subsystem.
`-k`	Stops the subsystem.
`-d`	Deletes the subsystem.
`-c`	Cleans the subsystems.
`-u`	Removes the topology services subsystem from all partitions.
`-t`	Turns tracing on for the subsystem.
`-o`	Turns tracing off for the subsystem.
`-r`	Refreshes the subsystem.
`-h`	Writes the script's usage statement to standard output.

Security

You must be running with an effective user ID of `root` to use this script.

Exit Status

0
Indicates that the script completed successfully.

1
Indicates that an error occurred.

Environment Variables

HB_SERVER_SOCKET
This environment variable should be set before this command can be executed. It must be set to the location of the UNIX-domain socket used by topology services clients to connect to the topology services daemon. This environment variable must be set to `/var/ha/soc/hats/server_socket.`*partition name*.

HA_SYSPAR_NAME
If HB_SERVER_SOCKET is not set, then HA_SYSPAR_NAME must be set to the partition name.

Restrictions

This command is valid in an HACMP environment only.

Use this command *only* under the direction of the IBM Support Center.

Standard Output

When the `-h` flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

This script writes error messages (as necessary) to standard error.

Examples

1. To add the topology services subsystem to the SRC, enter:

   ```
   topsvcsctrl -a
   ```

2. To start the topology services subsystem, enter:

   ```
   topsvcsctrl -s
   ```

3. To stop the topology services subsystem, enter:

   ```
   topsvcsctrl -k
   ```

4. To delete the topology services subsystem from the SRC, enter:

   ```
   topsvcsctrl -d
   ```

5. To clean up the topology services subsystem, enter:

   ```
   topsvcsctrl -c
   ```

6. To turn tracing on for the topology services daemon, enter:

   ```
   topsvcsctrl -t
   ```

7. To turn tracing off for the topology services daemon, enter:

   ```
   topsvcsctrl -o
   ```

Location

/opt/rsct/bin/topsvcsctrl
Contains the `topsvcsctrl` script

Files

/var/ha/log/topsvcs.*cluster_name*
Contains the log of the `hatsd` daemon on the cluster named `cluster_name`

touch Command

Purpose

Updates the access and modification times of a file.

Syntax

touch [**-a**] [**-c**] [**-m**] [**-f**] [**-r** *RefFile*] [*Time* | **-t** *Time* | **-d** *date_time*] { *File* ... | *Directory* ... }

Note: The preceding syntax is applicable only when the UNIX03 mode is not enabled in the AIX operating system.

touch [**-a c m f**] [**-r** *ref_file* | **-t** time | **-d** *date_time*] *file*... | *Directory*

Note: The preceding syntax is applicable only when the UNIX03 mode is enabled. To enable the UNIX03 mode in the AIX operating system, you must set the value of the **XPG_SUS_ENV** environment variable to **ON**.

Description

The **touch** command updates the access and modification times of each file specified by the *File* parameter of each directory specified by the *Directory* parameter. If you do not specify a value for the *Time* variable, the **touch** command uses the current time. If you specify a file that does not exist, the **touch** command creates the file unless you specify the **-c** flag.

The return code from the **touch** command is the number of files for which the times could not be successfully modified (including files that did not exist and were not created).

The **-a** and **-m** flags are active even when you do not specify them in the **touch** command.

Flags

Item	Description
-a	Changes the access time of the file specified by the *File* variable. Does not change the modification time unless **-m** is also specified.
-c	Does not create the file if it does not already exist. No diagnostic messages are written concerning this condition.

Item	Description
-d *Date_Time*	Uses the specified date and time instead of the current time. The *date_time* variable is specified in the decimal format, *YYYY-MM-DDThh:mm:SS[.frac][tz]* or *YYYY-MM-DDThh:mm:SS[,frac][tz]*, where: ***YYYY*** Specifies the four digits of the year (0000 to 9999). ***MM*** Specifies the month of the year (01 to 12). ***DD*** Specifies the day of the month (01 to 31). ***hh*** Specifies the hour of the day (00 to 23). ***mm*** Specifies the minute of the hour (00 to 59). ***SS*** Specifies the second of the minute (00 to 59). ***T*** Indicates a time designator and can be replaced with a single space. ***[.frac]*** Specifies a fractional second. It can either be blank or a period (.) followed by one or more decimal digits. ***[,frac]*** Specifies a fractional second. It is a comma (,) followed by one or more decimal digits. ***[tz]*** If the value of the *[tz]* parameter is blank, the local time zone is used for the resulting time. If the value of the *[tz]* parameter is a character Z, the Coordinate Time Zone (UTC) is used for the resulting time. If the value of the [tz] parameter is blank, the value of the TimezoneInfo (TZ) environment variable is used to identify the resulting time. The value of the *[tz]* parameter is implementation-specific if the resulting time is earlier than the Epoch time. Also, if the resulting time cannot be represented as a timestamp of the file that is specified by the *File* parameter, the **touch** command exits with an error status.
-f	Attempts to force the touch in spite of read and write permissions on a file.
-m	Changes the modification time of *File*. Does not change the access time unless **-a** is also specified.
-r *RefFile*	Uses the corresponding time of the file specified by the *RefFile* variable instead of the current time.

Item	Description
Time	Specifies the date and time of the new timestamp in the format *MMDDhhmm*[YY], where: ***MM*** Specifies the month of the year (01 to 12). ***DD*** Specifies the day of the month (01 to 31). ***hh*** Specifies the hour of the day (00 to 23). ***mm*** Specifies the minute of the hour (00 to 59). ***YY*** Specifies the last two digits of the year. If the *YY* variable is not specified, the default value is the current year (70 to 99 or 00 to 37). If the value of the *YY* digits is between 70 and 99, the century is assumed to be 19. If the value of the *YY* digits is between 00 and 37, the century is assumed to be 20.
-t *Time*	Uses the specified time instead of the current time. The *Time* variable is specified in the decimal form [[*CC*]*YY*]*MMDDhhmm*[**.***SS*] where: ***CC*** Specifies the first two digits of the year (19 to 21). ***YY*** Specifies the last two digits of the year (00 to 99). If the value of the *YY* digits is between 70 and 99, the value of the *CC* digits is assumed to be 19. If the value of the *YY* digits is between 00 and 37, the value of the *CC* digits is assumed to be 20. For years after 2038, specify the year in the *yyyy* format. ***MM*** Specifies the month of the year (01 to 12). ***DD*** Specifies the day of the month (01 to 31). ***hh*** Specifies the hour of the day (00 to 23). ***mm*** Specifies the minute of the hour (00 to 59). ***SS*** Specifies the second of the minute (00 to 59).

Note:

1. The **touch** command calls the **utimenstat()** subroutine to change the modification and access times of the file touched. This may cause the **touch** command to fail when flags are used if you do not actually own the file, even though you may have write permission to the file.
2. Do not specify the full path name **/usr/bin/touch** if you receive an error message when using the **touch** command.

Exit Status

This command returns the following exit values:

Item	Description
0	The command executed successfully. All requested changes were made.
>0	An error occurred.

Security

hm

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To update the access and modification times of a file, enter:

   ```
   touch program.c
   ```

   This sets the last access and modification times of the `program.c` file to the current date and time. If the `program.c` file does not exist, the **touch** command creates an empty file with that name.
2. To avoid creating a new file, enter:

   ```
   touch -c program.c
   ```
3. To update only the modification time, enter:

   ```
   touch -m *.o
   ```

   This updates the last modification times (not the access times) of the files that end with a `.o` extension in the current directory. The **touch** command is often used in this way to alter the results of the **make** command.
4. To explicitly set the access and modification times, enter:

   ```
   touch -c -t 02171425 program.c
   ```

   This sets the access and modification dates to 14:25 (2:25 p.m.) February 17 of the current year.
5. To use the time stamp of another file instead of the current time, enter:

   ```
   touch -r file1 program.c
   ```

   This gives the `program.c` file the same time stamp as the `file1` file.
6. To touch a file using a specified time other than the current time, enter:

   ```
   touch -t 198503030303.55 program.c
   ```

   This gives the `program.c` file a time stamp of 3:03:55 a.m. on March 3, 1985.

Files

Item	Description
/usr/bin/touch	Contains the **touch** command.

tpm_activate Command

Purpose

Changes the Trusted Platform Module (TPM) active states.

Syntax

tpm_activate [**-a**] [**-h**] [**-i**] [**-l** [**none | error | info | debug**]] [**-s**] [**-t**] [**-v**]

Description

The **tpm_activate** command reports the status of the TPM flags regarding the active state of the TPM. This is the default behavior, and it is also accessible through the **-s** (or **--status**) option. It prompts for the owner password when it reports the TPM status.

The **-a** (or **--active**) option changes the TPM to the active state (through the **TPM_PhysicalSetDeactivated** API). This operation is persistent. It requires physical presence for authorization, and a system reboot operation to take effect.

The **-i** (or **--inactive**) option (through the **TPM_PhysicalSetDeactivated** API) changes the TPM to the inactive state. This operation is persistent. It requires physical presence for authorization, and a system reboot operation to take effect. Although an inactive TPM can be considered to be off, it still allows the **tpm_takeownership** command to run.

The **-t** (or **--temp**) option causes immediate TPM deactivation (through the **TPM_SetTempDeactivated** API) to occur but persists only for the current boot cycle.

The **-s** (or **--status**), **-a** (or **--active**), **-i** (or **--inactive**), and **-t** (or **--temp**) options are mutually exclusive and the last option on the command line is carried out.

Flags

Item	Description
-a (or **--active**)	Makes the TPM active. This operation is persistent. The operation requires physical presence for authorization, and a system reboot operation to take effect.
-h (or **--help**)	Displays the command usage information.
-i (or **--inactive**)	Makes the TPM inactive. This operation is persistent. The operation requires physical presence for authorization, and a system reboot operation to take effect.
-l (or **--log**) [**none | error | info | debug**]	Sets the logging level to none, error, info, or debug as specified.
-s (or **--status**)	Reports the status of flags regarding the TPM active states.
-t (or **--temp**)	Makes the TPM inactive for the current boot cycle only.
-v (or **--version**)	Displays the command version information.

tpm_changeauth Command

Purpose

Changes the authorization data that is associated with the owner or storage root key.

Syntax

tpm_changeauth [**-g**] [**-h**] [**-l** [**none | error | info | debug**]] [**-n**] [**-o**] [**-r**] [**-s**] [**-u**] [**-v**] [**-z**]

Description

The **tpm_changeauth** command is used to change the authorization data for the Trusted Platform Module (TPM) owner or the TPM storage root key (through the **TPM_ChangeAuthOwner** API). This operation prompts for the current password, prompts for the new password, and prompts for a confirmation of the new password. The **-o** (or **--owner**) option changes the TPM owner password and the **-s** (or **--srk**) option changes the TPM storage root key (SRK) password.

Flags

Item	Description
-g (or **--original_password_unicode**)	Uses the Trusted Computing Group Software Stack (TSS) UNICODE encoding for the original password to comply with the applications that are using the TSS popup boxes.
-h (or **--help**)	Displays the command usage information.
-l (or **--log**) [**none	error	info	debug**]	Sets the logging level to none, error, info, or debug as specified.
-o (or **--owner**)	Changes the authorization data for the TPM owner.
-n (or **--new_password_unicode**)	Uses the TSS UNICODE encoding for the new password to comply with the applications that are using the TSS popup boxes.
-r (or **--set-well-known**)	Changes the password to a secret of all zeros (20 bytes of zeros). It must be specified which password (owner, SRK or both) needs to be changed.
-s (or **--srk**)	Changes the authorization data for the TPM storage root key.

Item	Description
-u (or **--unicode**)	Use the TSS UNICODE encoding for the passwords to comply with the applications that are using the TSS popup boxes.
-v (or **--version**)	Displays the command version information.
-z (or **--well-known**)	Changes the password to a new one when the current owner password is a secret of all zeros (20 bytes of zeros). It must be specified which password (owner, SRK, or both) needs to be changed.

tpm_clear Command

Purpose

Returns the Trusted Platform Module (TPM) to the default state (unowned, disabled, and inactive).

Syntax

tpm_clear [**-f**] [**-h**] [**-l** [**none | error | info | debug**]] [**-u**] [**-v**] [**-z**]

Description

The **tpm_clear** command requests the system TPM to perform a clear operation (through the **TPM_OwnerClear** API), which clears all the ownership information. Consequently, it invalidates all keys and the data that is tied to the TPM and disables and deactivates the TPM. This operation prompts for the owner password. The **-f** (or **--force**) option relies on the physical presence to authorize the command (through the **TPM_ForceClear** API) by skipping the owner password prompt.

Note: The **TPM_OwnerClear** API can be disabled until the current owner is cleared by using the **-f** (or **--force**) option with the **tpm_setclearable** command. The **TPM_ForceClear** API can be disabled for the current boot cycle with the **tpm_setclearable** command. This command requires you to reboot the system to complete the operation.

Flags

Item	Description
-f (or **--force**)	Lets the TPM rely on the physical presence for authorization, thus, skipping the owner password prompt.
-h (or **--help**)	Displays the command usage information.
-l (or **--log**) [**none | error | info | debug**]	Sets the logging level to none, error, info, or debug as specified.

Item	Description
-u (or **--unicode**)	Uses the Trusted Computing Group Software Stack (TSS) UNICODE encoding for the passwords to comply with the applications that are using the TSS popup boxes.
-v (or **--version**)	Displays the command version information.
-z (or **--well-known**)	Changes the password to a new one when the current owner password is a secret of all zeros (20 bytes of zeros). It must be specified which password (owner, storage root key, or both) needs to be changed.

tpm_clearable Command

Purpose

Disables the Trusted Platform Module (TPM) clear operations.

Syntax

tpm_clearable [**-f**] [**-h**] [**-l** [**none | error | info | debug**]] [**-o**] [**-s**] [**-u**] [**-v**] [**-z**]

Description

The **tpm_clearable** command reports the status of TPM flags regarding how the TPM can be cleared. This behavior is the default behavior, and it is also accessible through the **-s** (or **--status**) option. For requesting the TPM status report, it prompts for the owner password.

The **-o** (or **--owner**) option requests the TPM to disable the clear operations (through the **TPM_DisableOwnerClear** API) thus, disabling the owner from clearing out the ownership information. This operation prompts for the owner password. This operation remains in effect until the current owner is cleared.

The **-f** (or **--force**) option (through the **TPM_DisableForceClear** API) disables TPM clear operations by using physical presence to authorize a clear operation. This operation does not require authorization and skips the owner password prompt. This operation remains in effect only until a system reboot operation.

Flags

Item	Description
-f (or **--force**)	Disables the use of physical presence for authorizing a clear operation until a system reboot operation occurs.
-h (or **--help**)	Displays the command usage information.
-l (or **--log**) [**none | error | info | debug**]	Sets the logging level to none, error, info, or debug as specified.
-o (or **--owner**)	Disables the use of owner authorization for authorizing a clear operation until a new owner exists.
-s (or **--status**)	Report the status of flags regarding how the TPM can be cleared.
-u (or **--unicode**)	Uses the Trusted Computing Group Software Stack (TSS) UNICODE encoding for the passwords to comply with the applications that are using the TSS popup boxes.
-v (or **--version**)	Displays the command version information.
-z (or **--well-known**)	Changes the password to a new one when the current owner password is a secret of all zeros (20 bytes of zeros). It must be specified which password (owner, storage root key, or both) needs to be changed.

tpm_createek Command

Purpose

Creates an endorsement key pair on the Trusted Platform Module (TPM).

Syntax

tpm_createek [**-h**] [**-l** [**none | error | info | debug**]] [**-v**]

Description

The **tpm_createek** command creates an endorsement key pair on the TPM (through the **TPM_CreateEndorsementKeyPair** API). The endorsement key pair is not often required because it is normally installed as a part of manufacturing. However, you might need to run this command if commands such as **tpm_getpubek** are returning error code from the TPM layer.

Flags

Item	Description
-h (or **--help**)	Displays the command usage information.
-l (or **--log**) [**none | error | info | debug**]	Sets the logging level to none, error, info, or debug as specified.
-v (or **--version**)	Displays the command version information.

tpm_enable Command

Purpose

Changes the Trusted Platform Module (TPM) enabled states.

Syntax

tpm_enable [**-e**] [**-d**] [**-h**] [**-l** [**none | error | info | debug**]] [**-o**] [**-s**] [**-u**] [**-v**] [**-z**]

Description

The **tpm_enable** command reports the status of the TPM flags regarding the enabled state of the TPM. This is the default behavior, and it is also accessible through the **-s** (or **--status**) option. For requesting the TPM status report, it prompts for the owner password.

The **-e** (or **--enable**) option changes the system TPM to the enabled state (through the **TPM_OwnerSetDisable** API). This operation is persistent, and it prompts for the owner password.

The **-d** (or **--disable**) option (through the **TPM_OwnerSetDisable** API) changes the system TPM to the disabled state. This operation is persistent, and it prompts for the owner password. A disabled TPM can be considered to be off, and it does not allow the **tpm_takeownership** command to run.

The **-f** (or **--force**) option overrides the owner password prompt, and it relies on physical presence for the operation authorization (through the **TPM_PhysicalEnable** and **TPM_PhysicalDisable** APIs).

The **--enable**, **--disable**, and **--status** options are mutually exclusive, and the last option on the command line is carried out.

Flags

Item	Description
-e (or **--enable**)	Enables the TPM. This operation is persistent, and it prompts for owner authorization.
-d (or **--disable**)	Disables the TPM. This operation is persistent, and it prompts for owner authorization.
-h (or **--help**)	Displays the command usage information.
-l (or **--log**) [**none | error | info | debug**]	Sets the logging level to none, error, info, or debug as specified.
-o (or **--owner**)	Overrides the prompt for owner authorization and uses physical presence to authorize the action.
-s (or **--status**)	Reports the status of flags regarding the TPM-enabled states.
-u (or **--unicode**)	Uses the Trusted Computing Group Software Stack (TSS) UNICODE encoding for the passwords to comply with the applications that are using the TSS popup boxes.
-v (or **--version**)	Displays the command version information.
-z (or **--well-known**)	Changes the password to a new one when the current owner password is a secret of all zeros (20 bytes of zeros). It must be specified which password (owner, storage root key, or both) needs to be changed.

tpm_getpubek Command

Purpose

Displays the public part of the Trusted Platform Module (TPM) endorsement key.

Syntax

tpm_createek [**-h**] [**-l** [**none | error | info | debug**]] [**-u**] [**-v**] [**-z**]

Description

The **tpm_getpubek** command requests the TPM's public part of the endorsement key (through the **TPM_ReadPubek** API). This operation can be restricted to require owner authorization. In that case, the command prompts for the owner password and requests the data (through the **TPM_OwnerReadPubek** API). The public key information is displayed on a successful call.

Flags

Item	Description
-h (or **--help**)	Displays the command usage information.
-l (or **--log**) [**none | error | info | debug**]	Sets the logging level to none, error, info, or debug as specified.
-u (or **--unicode**)	Uses the Trusted Computing Group Software Stack (TSS) UNICODE encoding for the passwords to comply with the applications that are using the TSS popup boxes.
-v (or **--version**)	Displays the command version information.

Item	Description
-z (or **--well-known**)	Changes the password to a new one when the current owner password is a secret of all zeros (20 bytes of zeros). It must be specified which password (owner, storage root key, or both) needs to be changed.

tpm_ownable Command

Purpose

Verifies whether the Trusted Platform Module (TPM) allows the **tpm_takeownership** command to run.

Syntax

tpm_ownable [**-a**] [**-h**] [**-l** [**none | error | info | debug**]] [**-p**] [**-s**] [**-u**] [**-v**] [**-z**]

Description

The **tpm_ownable** command reports the status of the TPM flags regarding whether the TPM can be owned. This is the default behavior, and it is also accessible through the **-s** (or **--status**) option. Requesting a report of this status prompts for the owner password. The **-a** (or **--allow**) option sets the system TPM to allow **tpm_takeownership** operations (through the **TPM_SetOwnerInstall** API). This operation requires physical presence.

The **-p** (or **--prevent**) option (through the **TPM_SetOwnerInstall** API) prevents the TPM from accepting the **tpm_takeownership** command. This operation requires physical presence. These operations are persistent, and the **tpm_takeownership** command requires the TPM be enabled.

Flags

Item	Description
-a (or **--allow**)	Allows the **tpm_takeownership** command to run.
-h (or **--help**)	Displays the command usage information.
-l (or **--log**) [**none | error | info | debug**]	Sets the logging level to none, error, info, or debug as specified.
-p (or **--prevent**)	Prevents the **tpm_takeownership** command to run.
-s (or **--status**)	Reports the status of flags regarding whether the TPM can be owned.
-u (or **--unicode**)	Uses the Trusted Computing Group Software Stack (TSS) UNICODE encoding for the passwords to comply with the applications that are using the TSS popup boxes.
-v (or **--version**)	Displays the command version information.
-z (or **--well-known**)	Changes the password to a new one when the current owner password is a secret of all zeros (20 bytes of zeros). It must be specified which password (owner, storage root key, or both) needs to be changed.

tpm_present Command

Purpose

Changes the physical presence states and settings of the Trusted Platform Module (TPM).

Syntax

tpm_present [**-a**] [**-c**] [**--disable-cmd**] [**--disable-hw**] [**--enable-cmd**] [**--enable-hw**] [**-h**] [**-l** [**none | error | info | debug**]] [**--lock**] [**--set-lifetime-lock**] [**-u**] [**-v**] [**-z**] [**-y**]

Description

The **tpm_present** command reports the status of the TPM flags regarding TPM physical presence. This behavior is the default behavior, and it is also accessible through the **--status** option. It prompts for the owner password when it reports the TPM status. All changes are made with the **TSC_Physical Presence** API.

Flags

Item	Description
-a (or **--assert**)	Asserts that an administrator is physically present at the system.
-c (or **--clear**)	Removes the assertion that an administrator is physically present at the system.
--disable-cmd	Disallows the use of commands to signal that an administrator is physically present.
--disable-hw	Disallows the use of hardware signals to signal that an administrator is physically present.
--enable-cmd	Allows the use of commands to signal that an administrator is physically present.
--enable-hw	Allows the use of hardware signals to signal that an administrator is physically present.
-h (or **--help**)	Displays the command usage information.
-l (or **--log**) [**none | error | info | debug**]	Sets the logging level to none, error, info, or debug as specified.
--lock	Locks the assertions of physical presence in the current states until a system reboot operation.
--set-lifetime-lock	Allows no further changes to the flags controlling how physical presence can be signaled permanently. This option can never be undone.
-u (or **--unicode**)	Uses the Trusted Computing Group Software Stack (TSS) UNICODE encoding for the passwords to comply with the applications that are using the TSS popup boxes.
-v (or **--version**)	Displays the command version information.
-z (or **--well-known**)	Changes the password to a new one when the current owner password is a secret of all zeros (20 bytes of zeros). It must be specified which password (owner, storage root key, or both) needs to be changed.
-y (or **--yes**)	Answers yes to all questions. This flag is applicable only with the **--set-lifetime-lock** flag.

tpm_restrictpubek Command

Purpose

Restricts the ability to display the public part of the endorsement key to the owner.

Syntax

tpm_restrictpubek [**-h**] [**-l** [**none | error | info | debug**]] [**-r**] [**-s**] [**-v**]

Description

The **tpm_restrictpubek** command reports the status of who can display the public part of the endorsement key. This is the default behavior, and it is also available with the **-s** (or **--status**) option. This operation remains in effect until the owner is cleared and it prompts for the owner password. With the **-r** (or **--restrict**) option, the ability to display the public part of the endorsement key is restricted to the owner (through the **TPM_DisablePubekRead** API). The command prompts for the owner password to complete the operation. The **--status** and **--restrict** options are mutually exclusive, and the last option on the command line is carried out.

Flags

Item	Description
-h (or **--help**)	Displays the command usage information.
-l (or **--log**) [**none | error | info | debug**]	Sets the logging level to none, error, info, or debug as specified.
-r (or **--restrict**)	Restricts the owner to see the public part of the endorsement key.
-s (or **--status**)	Displays the status of who can see the public part of the endorsement key to the owner.
-u (or **--unicode**)	Uses the Trusted Computing Group Software Stack (TSS) UNICODE encoding for the passwords to comply with the applications that are using the TSS popup boxes.
-v (or **--version**)	Displays the command version information.
-z (or **--well-known**)	Changes the password to a new one when the current owner password is a secret of all zeros (20 bytes of zeros). It must be specified which password (owner, storage root key, or both) needs to be changed.

tpm_selftest Command

Purpose

Requests that the Trusted Platform Module (TPM) perform a self-test and report the results.

Syntax

tpm_selftest [**-h**] [**-l** [**none | error | info | debug**]] [**-r**] [**-v**]

Description

The **tpm_selftest** command requests that the system TPM performs a self-test (through the **TPM_SelfTestFull** API) and report the results. The **-r** (or **--results**) option reports the outcome of the last self-test operation without requesting another test to be run. If the TPM fails the self-test, it enters the failure mode where no commands are accepted. The results are reported in a manufacturer-specific format. The TPM self-test always runs automatically at every boot operation.

Flags

Item	Description
-h (or **--help**)	Displays the command usage information.
-l (or **--log**) [**none | error | info | debug**]	Sets the logging level to none, error, info, or debug as specified.

Item	Description
-r (or **--results**)	Reports results only.
-v (or **--version**)	Displays the command version information.

tpm_takeownership Command

Purpose

Sets up an owner on the Trusted Platform Module (TPM).

Syntax

tpm_takeownership [**-h**] [**-l** [**none | error | info | debug**]] [**-u**] [**-v**] [**-z**]

Description

The **tpm_takeownership** command sets up an owner on the system TPM (through the **TPM_TakeOwnership** API). This operation requires that the TPM be enabled and restricted by the **tpm_setownable** command. The command prompts for owner and security root key passwords and confirmations. This command can take a while to process.

Flags

Item	Description
-h (or **--help**)	Displays the command usage information.
-l (or **--log**) [**none | error | info | debug**]	Sets the logging level to none, error, info, or debug as specified.
-u (or **--unicode**)	Uses the Trusted Computing Group Software Stack (TSS) UNICODE encoding for the passwords to comply with the applications that are using the TSS popup boxes.
-v (or **--version**)	Displays the command version information.
-y (or **--owner-well-known**)	Sets the owner secret to all zeros (20 bytes of zeros).
-z (or **--well-known**)	Changes the password to a new one when the current owner password is a secret of all zeros (20 bytes of zeros). It must be specified which password (owner, storage root key, or both) needs to be changed.

tpm_version Command

Purpose

Reports the Trusted Platform Module (TPM) version and manufacturer information.

Syntax

tpm_version [**-h**] [**-l** [**none | error | info | debug**]] [**-v**]

Description

The **tpm_version** command reports the system TPM version and manufacturer information. The information reported is specific to the manufacturer.

Flags

Item	Description
-h (or **--help**)	Displays the command usage information.
-l (or **--log**) [**none | error | info | debug**]	Sets the logging level to none, error, info, or debug as specified.
-v (or **--version**)	Displays the command version information.

tprof Command

Purpose

Reports processor usage.

Syntax

tprof { [**-c**] [**-C** { **all** | *cpulist* }] [**-d**] **-D**] [**-e**] [-@ {**ALL** | *wparlist* }] [[{ **-E** [*mode* [**-b**] [**-B**] }]] [**-f** *frequency*]] [**-F**] [**-I**] [**-j**] [**-k**] [**-l**] [-L *objectlist*] [**-m** *objectslist*] [**-M** *sourcepathlist*] [**-N**] [**-p** *processlist*] [**-P** { **all** | *pidslist* }] [**-s**] [**-S** *searchpathlist*] [**-t**] [**-T** *buffersize*] [**-u**] [**-v**] [**-V** *verbosefilename*] [**-g**] [**-G** "*start=mmddhhmmssyy*",*end=mmddhhmmssyy*] [**-O** *options*]{ [**-z**] [**-Z**] | **-R** } { { **-r** *rootstring* } | { [**-A** { **all** | *cpulist* }[**-n**]] [**-r** *rootstring* [**-X** [*timedata* [, *buckets*=N]]]] { **-x** *program* | **-y** *program* } } } } { **-a** [**-A** [all]] [**-f** *frequency*] [**-F**] [**-v**] [**-z**] [**-V** *verbosefilename*] [**-T** *buffersize*] { { [**-r** *rootstring*] **-y** *program* } | { **-r** *rootstring* } } }

Note:

- All the list type inputs are separated by a comma except for pathlist, which is separated by a colon.
- Multi-cpu profiling mode is automatically disabled while running in real-time mode.
- Microprofiling is automatically disabled if per-processor profiling is turned on.
- Log Buffer size that was specified will be omitted if the **tprof** command runs in realtime mode.
- If the **-x** flag is specified without the **-A** flag, **tprof** runs in realtime mode.
- If the **-x** flag is specified with the **-A** flag, **tprof** runs in automated offline mode.
- If the **-x** flag is omitted **tprof** runs in post-processing mode or manual offline mode, depending on the presence of cooked files and the **-F** flag.
- The **-@** flag is automatically disabled if the **tprof** command runs in a workload partition in real-time or automated-offline modes.
- The **-y** flag can be used only with the **-E** flag or the **-a** flag.
- The **-O** *showaddrbytes=on* option cannot be used with the **-z** option.
- The **-O** *wrapfname=on* option should be used with the **-l** option.
- The **-G** option can be used only in post-processing mode.
- The **-O** *pdetails=on* option can be used only with the **-p** option.
- When manually collecting traces with the **-A** option for the **tprof** post-processing mode, it is mandatory to specify the **-pP** and **I** options of the **trace** command.

Description

The **tprof** command reports processor usage for individual programs and the system as a whole. This command is a useful tool for anyone with a Java, C, C++, or FORTRAN program that might be processor-bound and who wants to know which sections of the program are most heavily using the processor.

The **tprof** command can charge processor time to object files, processes, threads, subroutines (user mode, kernel mode and shared library) and even to source lines of programs or individual instructions. Charging processor time to subroutines is called profiling and charging processor time to source program lines is called micro-profiling.

For subroutine-level profiling, the **tprof** command can be run without modifying executable programs, that is no recompilation with special compiler flags is necessary. This is still true if the executables have been stripped, unless the traceback tables have also been removed. However, recompilation is required to get a micro-profile, unless a listing file is already available. To perform micro-profiling on a program, either the program should be compiled with the **-g** flag and the source files should be accessible to the **tprof** command or the program should be compiled with the **-q***list* flag and either both the object listing files and the source files or just the object listing files should be accessible to the **tprof** command. To take full advantage of **tprof** micro-profiling capabilities, it is best to provide both the `.lst` and the source file.

The **tprof** command can run in the following modes:

- Realtime or online
- Manual offline
- Automated offline
- Post-processing

If you specify the **-x** flag without the **-A** flag, the **tprof** command runs in realtime mode. In realtime mode, the **tprof** command starts the AIX **trace** utility in the background, and processes the trace data as it gets generated. When the program being profiled ends, **tprof** collects symbolic name information, and generates the **tprof** reports.

Note: This mode does not allow per-processor profiling.

If you specify the **-x** flag with the **-A** flag, the **tprof** command runs in automated offline mode. In this mode, the **tprof** command starts the AIX **trace** utility and logs the trace data into a file. Once the trace data collection is done, it collects symbolic name information, and the **tprof** command opens the trace log file and processes the data to generate reports. In this mode, the **tprof** command generates the following files in addition to the tprof report files:

- *rootstring*.**syms**
- *rootstring*.**trc** [**-cpuid**]

All of the input and report files used by the **tprof** command are named *rootstring.suffix*, where *rootstring* is either specified with the **-r** flag, or is the program name specified with the **-x** flag.

In realtime mode and automated offline mode, the *ulimit* value of the data area for the program that is being profiled is set to **unlimited**.

In automated offline mode, you can specify the **-N** flag to collect source line information into the generated **RootString.syms** file. And you can specify the **-I** flag to collect binary instructions into the generated **RootString.syms** file.

The **tprof** command can re-process these files any time to generate profiling reports. This is called manual offline mode. The *rootstring***.syms** file contains symbolic name information similar to the output of the **gensyms** command. The *rootstring.trc*[**-cpuid**] files are trace log files. The **-cpuid** is added to the names when per-processor tracing is on. In that case, each file contains trace data from one processor only.

If you specify the **-c** flag with the **-A** flag, the *rootstring***.syms** and *rootstring.trc*[**-cpuid**] files are not generated. Instead, the following two files are created:

- *rootstring***.csyms**
- *rootstring***.ctrc**[**-cpuid**]

Those files are *cooked*, that is they are a pre-processed version of the normal trace and name files. **tprof** post-processes cooked file much faster.

If you specify neither the **-A** flag nor the **-x** flag, the **tprof** command runs either in manual offline or in post-processing mode. To run the **tprof** command in post-processing mode, the following files must be available:

- *rootstring***.csyms**
- *rootstring***.ctrc**[**-cpuid**]

These files are generated when the **tprof** command runs (in any mode except post-processing mode) with the **-c** flag.

To run the **tprof** command in manual offline mode, the following files must be available:

- *rootstring***.syms**
- *rootstring***.trc** [**-cpuid**]

To generate these files, you need to manually run the **gensyms** command and AIX trace facility, or run the **tprof** command in automated offline mode without the **-c** flag.

The **tprof** command always first looks for *rootstring.csyms* and *rootstring***.ctrc**[**-cpuid**] files. Only if these files are not available, does it look for the *rootstring***.syms** and *rootstring***.trc**[**-cpuid**] files. To prevent the **tprof** command from looking for the *rootstring***.csyms** and *rootstring***.ctrc**[**-cpuid**] files, that is, force the manual offline mode, use the **-F** flag.

If the input symbols file contains demangled names, you cannot use the **-Z** flag.

The **tprof** command generates a **tprof** report file named *rootstring***.prof**, which holds the process, thread, object file and subroutine level profiling report. The file can contain the following sections and subsections:

- Summary report section:
  - Processor usage summary by process name
  - Processor usage summary by threads (tid)
- Global (pertains to the execution of all processes on system) profile section:
  - Processor usage of user mode routines
  - Processor usage of kernel routines, including milicode routines called in kernel mode
  - Processor usage summary for kernel extensions
  - Processor usage of each kernel extension's subroutines
  - Processor usage summary for privately loaded, global, and named shared libraries, and milicode routines called in user mode
  - Processor usage of each shared library's subroutines
  - Processor usage of each Java class
  - Processor usage of each Java methods of each Java class
- Process and thread level profile sections (one section for each process or thread) :
  - Processor usage of user mode routines for this process/thread
  - Processor usage of kernel routines for this process/thread, including milicode routines called in kernel mode
  - Processor usage summary for kernel extensions for this process/thread
  - Processor usage of each kernel extension's subroutines for this process/thread
  - Processor usage summary for privately loaded, global, and named shared libraries for this process/thread, and milicode routines called in user mode
  - Processor usage of each shared library's subroutines for this process/thread
  - Processor usage of each Java class for this process/thread
  - Processor usage of Java methods of each Java class for this process/thread

The summary report section is always present in the *rootstring***.prof** report file. You can turn on or turn off various subsections of the global profile section using the following profiling flags:

- **-u** turns on subsections a
- **-k** turns on subsection b
- **-e** turns on subsections c and d
- **-s** turns on subsections e and f
- **-j** turns on subsections g and h

If you specify the **-p**, **-P** and **-t** flags, the process and thread level profile sections are created for processes and threads. The subsections present within each of the per-process of per-thread sections are identical to the subsections present in the global section, they are selected using the profiling flags (**-u**,**-s**,**-k**,**-e**,**-j**).

Optionally, if you run the **tprof** command with the **-C** flag, the command also generates per-processor profiling reports, which contains one profiling report per processor. The generated **tprof** reports have the same structure and are named using the convention: *rootstring***.prof**[**-cpuid**].

If you specify the **-m** flag, the **tprof** command generates micro-profiling reports. The reports use the following naming convention: *rootstring*.*source***.mprof**, where source is the base name of a source file. If more than one source file has the same base name, a number to uniquely identify them is appended to the report file names. For example, *rootstring.Filename***.c.mprof-1**. The micro-profiling report has the following information:

- The full path name of the annotated source file.
- A hot line profile section which has all the line numbers from that source file hit by profiling samples, sorted by processor usage. For each source line, one line reports the percentage of time spent on behalf of all processes, followed by additional lines with the breakdown by individual process.
- A source line profile section for each of the functions in that source file, which have processor usage. This section contains the source line number, processor usage and source code. If a **.lst** file for that source file is accessible to tprof, then it interlaces the instruction lines from the .lst file with the source lines from the source file and charges processor usage appropriately. This provides breakdown by instruction for each source file.

  If a source file is not present, but a **.lst** file is present, **tprof** only shows the processor usage based on the source lines and the instructions from the **.lst** file.

  If neither the **.lst** file nor the source file is present, but the source file is compiled with the **-g** flag, the **tprof** command retrieves the source line numbers and generates a similar report, with the source code column missing.

  Note: If per-processor profiling is requested, micro-profiling is automatically disabled. The **tprof** command cannot report correct source line information if a **.c** file is included in another **.c** file. The **tprof** command cannot micro-profile Java classes or methods.

If you specify the **-m** flag, the **-N** flag is automatically specified to gather the source line info into a symbols file in automated offline mode.

If you specify the `-Z` flag with the `-m` flag, one report file is generated per subroutine. The following naming convention is used: `RootString.source.routine.mprof`, where *routine* is the name of one of the subroutines listed in the source file. In addition, a file named `RootString.source.HOT_LINES.mprof` containing the hot line profiling information described above is also created.

If you specify the `-L` flag, the `tprof` command generates annotated listing files. The files use the following naming convention: `RootString.source.alst`, where *source* is the base name of a source file. If more than one source file has the same base name, a number to uniquely identify them is appended to the report file name. For example, `RootString.Filename.c.alst-1`. If you specify the `-Z` flag with the `-L` flag, one report file is generated per subroutine. The following naming convention is then used: `RootString.source.routine.alst`, where *routine* is the name of one of the subroutines listed in the source file.

If you specify the **-N** flag or **-I** flag when profiling a Java program using JPA (**-x java -Xrunjpa** or **-x java -agentlib:jpa**), the JIT source line number and instructions can be collected if the corresponding parameter is added to the **-Xrunjpa** flag or the **-agentlib:jpa** flag:

- **source=1** turns on JIT source line collecting (requires IBM JRE 1.5.0 or later version).
- **instructions=1** turns on JIT instructions collecting.

The following restrictions apply for non-root users running the **tprof** command:

- The **tprof** will not be able to verify that the running kernel is the same as the **/unix** file. This means that even if a warning message is displayed, in most cases the running kernel and **/unix** are the same, so the data should be accurate.
- When the **gensyms** command is run by a non-root user, the same warning as in restriction #1 (above) is given and the **gensyms** file is marked. If **tprof** is run in the offline mode, the file created with the **gensyms** command will flag **tprof** as to kernel that is not verified.
- The **tprof** will not be able to open and read symbols on files which do not have the read permission set. Some private, shared libraries do not have read permission, and some kernel extensions are not readable.

Time-Based versus Event-Based Profiling

By default, `tprof` is time-based and is driven by the decrementer interrupt. Another mode of profiling is event-based profiling, in which the interrupt is driven by either software-based events or by Performance Monitor events. With event-based profiling, both the sampling frequency and the profiling event can be varied on the command line.

The `-E` flag enables event-based profiling. The `-E` flag is one of the four software-based events (EMULATION, ALIGNMENT, ISLBMISS, DSLBMISS) or a Performance Monitor event (PM_*). By default, the profiling event is processor cycles. All Performance Monitor events are prefixed with PM_, such as PM_CYC for processor cycles or PM_INST_CMPL for instructions completed. The `pmlist` lists all Performance Monitor events that are supported on a processor. The chosen Performance Monitor event must be taken in a group where we can also find the `PM_RUN_INST_CMPL` Performance Monitor event. On POWER4 and later processors, profiling on marked events results in better accuracy. Marked events have the PM_MRK_ prefix.

If you specify the **-y** flag, only the specified program and its descendents are profiled. Use the **-y** flag only with the **-E** or **-a** flag.

The `-f` flag varies the sampling frequency for event-based profiling. For software-based events and processor cycles, supported frequencies range from 1 to 500 milliseconds, with a default of 10 milliseconds. For all other Performance Monitor events, the range is from 10000 to MAXINT occurrences of the event, with a default of 10000 events. If you specify the **-f** flag with the **-y** flag, the sampling frequency can range from 1 through the MAXINT occurrences for other Performance Monitor events, with a default of 10000 events.

Additional information is added to the `.prof` file to reflect the processor name, profiling event, and sampling frequency.

Java Applications Profiling

To profile Java applications, you must specify the **-j** flag, and start the applications with the **-Xrunjpa** API (for running on Java 5 and earlier JVMs) or the **-agentlib:jpa** (for running on Java 6 JVM) of the **java** command line option. When you specify this option, the JVM will automatically calls the **jpa** library whenever new classes and methods are loaded into memory. The library will in turn collect address to name mapping information for methods and classes in files named **/tmp/Java*PID*.syms**, where *PID* is the process ID of a process running a Java Virtual Machine. The **tprof** command will automatically look in that directory for such files.

When running in automated offline mode, or selecting the cooking flags, the **tprof** command will copy the information contained in **JavaPID.syms** files into the **RootString.syms** or **RootString.csyms** file. The corresponding files in **/tmp** can then be deleted. The directory content should be kept up to date by **tprof** command users. Whenever the JVM corresponding to a particular **JavaPID.syms** is stopped, the file should be deleted.

Profile Accuracy

The degree to which processor activity can be resolved is determined by the number of samples captured and the degree to which *hot spots* dominate. While a program with a few hot spots can be profiled with relatively few samples, less-frequently executed sections of the program are not visible in the profiling reports unless more samples are captured. In cases where user programs run less than a minute, there may be insufficient resolution to have a high degree of confidence in the estimates.

A simple solution is to repeatedly execute the user program or script until you achieve the degree of resolution you need. The longer a program is run, the finer the degree of resolution of the profile. If you doubt the accuracy of a profile, run the **tprof** command several times and compare the resulting profiles.

Information

The `-@` flag controls the addition of WPAR information to a `tprof` report. Sub-options specify what information is included to some of the report sections; these sub-options is in one of the following forms:

- The `-@` flag alone (that is, with no suboption) adds a summary of the processor usage WPAR name. Also, the WPAR name is shown for each process listed in the sections summarizing processor usage by process and by thread.
- The **ALL** suboption causes the `tprof` report to contain a process, thread, object file and subroutine-level profiling report for the overall system and for each running WPAR.
- A comma-separated list of WPAR names results in a process, thread, object file and subroutine-level profile section for each named WPAR in the `tprof` report.

Note: When a WPAR is used as a checkpoint and is restarted, some shared library areas might be local to the WPAR. In this case, the name of the WPAR is printed after the name of the area *myarea@mywpar*. In all other cases, the area is system-wide; thus the WPAR name is omitted.

XML Report Generating

The **-X** flag generates an XML report file named **RootString.etm**. This file can be shown in Visual Performance Analyzer. The XML report file contains four sections:

- Profile general information
- Symbol data
- Profile hierarchy
- Temporal data

The **-X** is used in automated offline mode to generate XML report directly.

The **-X** is also used in manual offline mode to generate XML report from the **RootString.syms** and **RootString.trc** files.

If the **-X** *timedata* is specified, the generated XML report will include the time data information. By default, the time data generating function is turned off.

To specify the bucket number for the time data, use the *buckets*=N argument. The default bucket number is 1800.

Large Page Analysis

The **tprof -a** command collects the profile trace from a representative application run, and produces performance projections. The projections map different portions of the data space of an application to different page sizes. The large page analysis uses the information in the trace to project translation buffer performance when the command maps any of the following application memory regions to a different page size:

- Static application data (data that is initialized or not initialized)
- Application heap (data that is dynamically allocated)
- Stack
- Application text

Performance projections are provided for each of the page sizes that the operating system supports. The first performance projection is a baseline projection that maps all of the memory regions to a default page size of 4 KB. Subsequent projections map one region at a time to a different page size. The following statistics are reported for each projection:

- Page size
- Number of pages needed to back all of the regions
- Translation miss score
- Cold translation miss score

The summary section lists the processes that are profiled and the statistics that are reported. It includes the following information:

- Number or percentage of memory reference
- Modeled memory reference
- Malloc calls
- Free calls

Data Profiling

The **tprof -b** command turns on basic data profiling and collects data access information. The summary section reports access information across the kernel data, library data, user global data, and the stack heap sections for each process.

If you specify the **-b** flag with the **-s**, **-u**, **-k**, and **-e** flags, the **tprof** command data profiling reports most used data structures (exported data symbols) in shared library, binary, kernel and kernel extensions. The **-b** flag also reports the functions that use those data structures.

Comparison of tprof Versus prof and gprof

The most significant differences between these three commands is that **tprof** collects data with no impact on the execution time of the programs being profiled, and works on optimized and stripped binaries without any need for recompilation, except to generate micro-profiling reports. Neither **gprof** nor **prof** have micro-profiling capabilities or work on optimized binaries, while they do require special compilation flags, and induce a slowdown in the execution time that can be significant. **prof** does not work on stripped binaries.

The **prof** and **gprof** tools are standard, supported profiling tools on many UNIX systems, including this operating system. Both **prof** and **gprof** provide subprogram profiling and exact counts of the number of times every subprogram is called. The **gprof** command also provides a very useful *call graph* showing the number of times each subprogram was called by a specific parent and the number of times each subprogram called a child. The **tprof** command provides neither subprogram call counts nor call graph information.

Like the **tprof** command, both the **prof** and **gprof** commands obtain their processor consumption estimates for each subprogram by sampling the program counter of the user program.

tprof collects processor usage information for the whole system, while **prof** and **gprof** collect only profiling information for a single program and only for the time spent in user mode.**tprof** also provides summary for all processes active during the execution of the profiled user program and fully support libraries and kernel mode profiling.

tprof support the profiling of Java applications, which **prof** and **gprof** do not.

Flags

Item	Description
-@ { **ALL** | *wparlist* }	Includes the WPAR information in the generated reports. The **ALL** option includes summaries for all of the WPAR. When this option is set, the report contains a 'SYSTEM' report and a report per WPAR traced. The *wparlist* option specifies a comma-separated list of WPAR. When the *wparlist* option is set, the **tprof** command produces a report for each WPAR specified.
-a	Turns on the large page analysis.
-A { **all** | *cpulist* }	Turns on automatic offline mode. No argument turns off per-processor tracing. **all** enables tracing of all processors. *cpulist* is a comma separated list of processor-ids to be traced.
-b	Turns on basic data profiling.
-B	Turns on basic data profiling with the information about the instruction address mapped function.
-c	Turns on generation of cooked files.
-C all | *cpulist*	Turns on the per-processor profiling. Specify **all** to generate profile reports for all processors. Processor numbers should be separated with a comma if you give a *cpulist* (for example, 0,1,2). **Note:** per-processor profiling is possible only if per-processor trace is either on (in automated offline mode), or has been used (in manual offline mode). It is not possible at all in online mode. This option is not supported if the number of CPUs traced is greater than 128.
-d	Turns on deferred tracing mode, that is defers data collection until **trcon** is called.
-D	Turns on detailed profiling which displays processor usage by instruction offset under each subroutine.
-e	Turns on kernel extension profiling.
-E [*mode*]	Enables event-based profiling. The possible modes are: **PM_*event*** Specifies the hardware event to profile. If no mode is specified for the **-E** flag, the default event is processor cycles (**PM_CYC**). **EMULATION** Enables the emulation profiling mode. **ALIGNMENT** Enables the alignment profiling mode. **ISLBMISS** Enables the Instruction Segment Lookaside Buffer miss profiling mode. **DSLBMISS** Enables the Data Segment Lookaside Buffer miss profiling mode.

Item	Description
-f *frequency*	Specifies the sampling frequency. The sampling frequency can be from 1 to 500 milliseconds for processor cycles and **EMULATION**, **ALIGNMENT**, **ISLBMISS**, and **DSLBMISS** events, and from 10000 to **MAXINT** event occurrences for other Performance Monitor events. If you specify the **-f** flag with the **-y** flag, the value of the sampling frequency ranges from 1 through the value of the MAXINT occurrences for other Performance Monitor events, with the default value of 10000 events.
-F	Overwrites cooked files if they exists. If used without the **-x** flag, this forces the manual offline mode.
-g	Does not translate symbol names into human-readable names.
-G	Sets trace processing start date and end date. The parameters are specified in the following format: `"start=mmddhhmmssyy,end=mmddhhmmssyy"` where `mmddhhmmssyy` is the month, day, hour, minute, second, and year respectively. This option can have the following values: **start** When set, trace processing starts from the specified start date string. **end** When set, trace processing stops at the specified end date string.
-I	Turns on binary instructions collecting. **Note:** The **-I** flag activates to gather binary instructions when generating symbol files or cooked symbol files in automated offline mode. However, in manual offline mode, the **-I** flag does not affect the report files.
-j	Turns on Java classes and methods profiling.
-k	Enables kernel profiling.
-l	Enables long names reporting. By default **tprof** truncates the subroutine, program and source file names if they do not fit into the available space in the profiling report. This flag disables truncation.
`-L` *objectlist*	Enables listing annotation for objects specified by the comma separated list, *objectlist*. Executables and shared libraries can have their listing files annotated. Specify the archive name for libraries. **Note:** 1. To enable listing annotation of programs, user mode profiling (`-u`) must be turned on. 2. To enable listing annotation of shared libraries, shared library profiling (`-s`) must be turned on. 3. To annotate a listing generated with IPA compilations, specify a `.lst` as the *objectlist*.

Item	Description
-m *objectslist*	Enables micro-profiling of objects specified by the comma separated list, *objectlist*. Executables, shared libraries, and kernel extensions can be micro-profiled. Specify the archive name for libraries and kernel extensions. **Note:** 1. To enable micro-profiling of programs, user mode profiling (**-u**) must be turned on. 2. To enable micro-profiling of shared libraries, shared library profiling (**-s**) must be turned on. 3. To enable micro-profiling of kernel extensions, kernel extension profiling (**-e**) must be turned on.
-M *PathList*	Specifies the source path list. The *PathList* is a colon separated list of paths that are searched for source files and **.lst** files that are required for micro-profiling and listing annotation. By default the source path list is the object search path list.
-n	Turns off postprocessing. If the **-n** flag is specified, the **-u**, **-s**, **-k**, **-e**, and **-j** flags are ignored. The data is collected, the **.trc** file and the **gensyms** files are generated, but the **.prof** file is not generated. This helps avoid overloading the system during a benchmark, for example. The **-A** flag must be used if the **-n** option is used.
-N	Turns on source line number info collecting. The **-N** flag activates to gather source line information when generating symbol files or cooked symbol files in automated offline mode. However, in manual offline mode, the **-N** flag does not affect the report files.
-O	This option can have the following values: **showaddrbytes=[on|off]** Turns on the `Address` and `Bytes` columns in subroutine reports. The default value is `off`. **wrapfname=[on|off]** Turns on the line wrap of the long function name. To wrap the function names on a line, set value as -l. The default value is `off`. **pdetails=[on|off]** Turns on the data consolidation process for the report. The report consolidates data for the specified `processlist` in the **kernel** and **sharedlib** segment of the Process Summary section in the report.
-p *processlist*	Enables process level profiling of the process names specified in the *processlist*. *processlist* is a comma separated list of process names Process level profiling is enabled only if at least one of the profiling modes (**-u**,**-s**,**-k**,**-e**, or **-j**) is turned on.
-P { **all** | *PIDList* }	Enables process level profiling of **all** processes encountered or for processes specified with *PIDList*. The *PIDList* is a comma separated list of process-IDs. Process level profiling is enabled only if at least one of the profiling modes (**-u**,**-s**,**-k**,**-e**, or **-j**) is turned on.

Item	Description
-r *rootstring*	Specifies the *rootstring*.**tprof** input and report files all have names in the form of *rootstring.suffix*. If you do not specify the **-r** flag, the *rootstring* parameter uses the default program name that the **-x** flag specifies.
`-R`	Specifies that the `tprof` command should use samples weighted by PURR increment values to calculate percentages. This is the preferred mode when running in either simultaneous multithreading or Micro-Partitioning environments. The `-R` flag cannot be used with either the `-z` flag or the `-Z` flag.
-s	Enables shared library profiling.
-S *PathList*	Specifies the object search *PathList*. The *PathList* is a colon separated list of paths that are searched for executables, shared libraries and kernel extensions. The default object search *PathList* is the environment path list (**$PATH**).
-t	Enables thread level profiling. If **-p** or **-P** are not specified with the **-t** flag, **-t** is equivalent to **-P all -t**. Otherwise, it enables thread level reporting for the selected processes. Thread level profiling is enabled only if at least one of the profiling modes (**-u**,**-s**,**-k**,**-e**, **-j**) is enabled.
-T *buffersize*	Specifies the trace *buffersize*. This flag has meaning only in real time or automated offline modes.
-u	Enables user mode profiling.
-v	Enables verbose mode.
-V *File*	Stores the verbose output in the specified *File*.
-x *program*	Specifies the program to be executed by **tprof**. Data collection stops when *program* completes or **trace** is manually stopped with either **trcoff** or **trcstop** The **-x** flag must be the last flag in the list of flags specified in **tprof**.
-X	Specifies the **tprof** command to call XML Generator when the **tprof** profiling is finished, and to generate the XML report directly from the **tprof trace** and **symlib** data. The **-X** option needs Java. Install the Java first, and make sure Java is in PATH.
-y	Turns on the event-based profiling for only the specified command and its descendents.
-z	Turns on ticks report. Enables compatibility mode with the previous version of **tprof**. By default processor usage is only reported in percentages. When **-z** is used, **tprof** also reports ticks. This flag also adds the Address and Bytes columns in subroutine reports. If you specify the **-z** flag with the **-a** flag, the process summary section in the report displays numbers rather than percentages.

Item	Description
-Z	Switches reports to use ticks instead of percentages (same as the -z flag), and splits annotated listing (when used with the -L flag) and annotated source files (when used with the -m flag) into multiple files, one per subroutine. This option turns on the **-g** flag.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. The following example shows the basic global program and thread-level summary:

   ```
   $tprof -x sleep 10
   ```

   An output that is similar to the following is displayed:

   ```
   Mon May 21 00:39:26 2012 System: AIX 6.1 Node: dreaming Machine: 000671894C00
   Starting Command sleep 10
   stopping trace collection.
   Generating sleep.prof
   ```

   The **sleep.prof** file that is generated only contains the summary report section.
2. The following example shows the global profiling with all options:

   ```
   $tprof -skeuj -x sleep 10
   ```

   An output that is similar to the following is displayed:

   ```
   Mon May 21 00:39:26 2012
   System: AIX 6.1 Node: drea
   ming Machine: 000671894C00
   Starting Command sleep 10
   stopping trace collection.
   Generating sleep.prof
   ```

   The `sleep.prof` file that is generated contains the summary report and global profile sections.
3. The following example shows the single process level profiling:

   ```
   $tprof -u -p workload -x workload
   ```

   An output that is similar to the following is displayed:

   ```
   Mon May 21 00:39:26 2012
   System: AIX 6.1 Node: drea
   ming Machine: 000671894C00
   Starting Command workload stopping trace collection.
   Generating workload.prof
   ```

   The **workload.prof** file that is generated contains the summary report, the global user mode profile sections, and one process level profile section for the process 'workload' that contains only a user mode profile subsection.
4. The following example shows the multiple process level profiling:

   ```
   $tprof -se -p send,receive -x startall
   ```

An output that is similar to the following is displayed:

```
Mon May 21 00:39:26 2012
System: AIX 6.1 Node: drea
ming Machine: 000671894C00
Starting Command startall
stopping trace collection.
Generating startall.prof
```

The **startall.prof** file that is generated contains the summary report, the global shared library mode profile, the global kernel extension profile sections, and two process level profile sections: one for the process 'send', and one for the process 'receive'. The process level sections each contain two subsections: one with shared library profiling information and one with kernel extensions profiling information.

5. The following example shows the micro-profiling and listing annotation:

```
$tprof -m ./tcalc -L ./tcalc -u -x ./tcalc
```

An output that is similar to the following is displayed:

```
Mon May 21 00:39:26 2012
System: AIX 6.1 Node: drea
ming Machine: 000671894C00
Starting Command ./tcalc
stopping trace collection.
Generating tcalc.prof
Generating tcalc.tcalc.c.mprof
Generating tcalc.tcalc.c.alst
```

The **tcalc.prof** file that is generated contains the summary report and the global user mode profile sections. The resulting **tcalc.tcalc.c.mprof** and `tcalc.tcalc.c.alst` files contain the micro-profiling report and the annotated listing.

6. For event-based profiling on processor cycles, sampling once every 100 milliseconds, enter the following command:

```
$tprof -E -f 100 -Askex sleep 10
```

The output is similar to the following display:

```
Starting Command sleep 10
stopping trace collection.
Tue Apr 26 14:44:02 2005
System: AIX 5.3 Node: bigdomino Machine: 00C0046A4C00
Generating sleep.trc
Generating sleep.prof
Generating sleep.syms
```

7. For event-based profiling on completed instructions, sampling once every 20,000 completed instructions, enter the following command:

```
$tprof -E PM_INST_CMPL -f 20000 -Askex sleep 10
```

The output is similar to the following display:

```
Starting Command sleep 10
stopping trace collection.
Tue Apr 26 14:42:44 2005
System: AIX 5.3 Node: bigdomino Machine: 00C0046A4C00
Generating sleep.trc
Generating sleep.prof
Generating sleep.syms
```

8. For event-based profiling on emulation interrupts, sampling once every 10000 events, enter the following command:

```
$tprof -E EMULATION -Askex sleep 10
```

The output is similar to the following display:

```
Starting Command sleep 10
stopping trace collection.
Tue Apr 26 14:41:44 2005
System: AIX 5.3 Node: bigdomino Machine: 00C0046A4C00
Generating sleep.trc
Generating sleep.prof
Generating sleep.syms
```

9. The following example shows the automated offline mode:

```
$tprof -c -A all -x sleep 10
```

The output is similar to the following display:

```
Starting Command sleep 10
stopping trace collection.
Mon May 21 00:39:26 2012
System: AIX 6.1 Node: drea
ming Machine: 000671894C00
Generating sleep.ctrc
Generating sleep.csyms
Generating sleep.prof
```

The **sleep.prof** file that is generated only has a summary report section, while the two cooked files are ready to be re-postprocessed.

10. The following example shows the automated offline mode that is enabling source line collecting:

```
$tprof -A -N -x sleep 10
```

The output is similar to the following display:

```
Starting Command sleep 10
stopping trace collection.
Wed Feb 8 15:12:41 2006
System: AIX 5.3 Node: aixperformance Machine: 000F9F3D4C00
Generating sleep.trc
Generating sleep.prof
Generating sleep.syms
```

The **sleep.prof** file that is generated only contains the summary report section, while **sleep.syms** contains the source line information.

11. The following example shows the automated offline mode that is enabling source line and instruction collecting:

```
$tprof -A -N -I -r RootString -x sleep 10
```

The output is similar to the following display:

```
Starting Command sleep 10
stopping  trace collection.
Wed Feb  8 15:16:37 2006
System: AIX 5.3 Node: aixperformance Machine: 000F9F3D4C00
Generating RootString.trc
Generating RootString.prof
Generating RootString.syms
```

The **rootstring.prof** file is generated. The **rootstring.syms** file contains the source line information and binary instructions.

12. To enable Java source line and instructions collecting for the application HelloAIX that is running on Java 5 JVM in realtime mode, enter the following command:

```
$tprof -N -I -x java -Xrunjpa:source=1,instructions=1 Hello AIX
```

The output is similar to the following display:

```
Thu Feb  9 13:30:38 2006
System: AIX 5.3 Node: perftdev Machine: 00CEBB4A4C00
Starting Command java -Xrunvpn_jpa:source=1,instructions=1 Hello AIX
Hello AIX!
stopping trace collection.
Generating java.prof
```

The **java.prof** file is generated. It contains the JIT source line information and the JIT instructions.

13. The following example shows the processor usage for the **vloop_lib_32** program without any shared library, thread-level profiling, per-processor tracing, or post processing:

```
$tprof -A -n -s -t -r test -x vloop_lib_32 5
```

The output is similar to the following display:

```
Starting Command vloop_lib_32 5
stopping trace collection.
Generating test.trc
Generating test.syms
```

14. The following is an example of the automated offline mode for XML report:

```
$tprof -A -X -r RootString -x sleep 10
Starting Command sleep 10
stopping trace collection.
Tue Apr 17 22:00:24 2007
System: AIX 5.3 Node: test105 Machine: 00CEBB4A4C00
Generating sleep.trc
Generating sleep.syms
Calling tprof2xml to generate XML report.
tprof2xml TraceReader Version 1.2.0
Tue Apr 17 22:00:24 2007
System: AIX 6.1 Node: test105 Machine: 00CEBB4A4C00
 ------------------0------------------
Record 0
Post-processing counters
Retrieving Disassembly
writing the XML
Writing symbol list
.
Writing process hierarchy
Finished writing sleep.etm
```

15. The following is an example of the automated offline mode enabling source line and instruction collecting:

```
$tprof -A -N -I -X -x sleep 10
Starting Command sleep 10
stopping trace collection.
Tue Apr 17 22:00:24 2007
System: AIX 5.3 Node: test105 Machine: 00CEBB4A4C00
Generating sleep.trc
Generating sleep.syms
Calling tprof2xml to generate XML report.
tprof2xml TraceReader Version 1.2.0
Tue Apr 17 22:00:24 2007
System: AIX 6.1 Node: test105 Machine: 00CEBB4A4C00
------------------0------------------
Record 0
Post-processing counters
Retrieving Disassembly
writing the XML
Writing symbol list
.
Writing process hierarchy
Finished writing sleep.etm
The symbol data elements in the xml report will have both bytes and
LineNumberList child elements.
```

16. The following is an example of the automated offline mode for XML report enabling timedata:

```
$tprof -A -X timedata,buckets=100 -x sleep 10
Starting Command sleep 10
stopping trace collection.
Tue Apr 17 22:18:06 2007
System: AIX 5.3 Node: test105 Machine: 00CEBB4A4C00
Generating RootString.trc
Generating RootString.syms
Calling tprof2xml to generate XML report.
tprof2xml TraceReader Version 1.2.0
Tue Apr 17 22:18:06 2007
System: AIX 5.3 Node: test105 Machine: 00CEBB4A4C00
Tue Apr 17 22:18:06 2007
System: AIX 5.3 Node: test105 Machine: 00CEBB4A4C00
------------------0------------------
Record 0
Post-processing counters
Retrieving Disassembly
writing the XML
Writing symbol list
.
Writing process hierarchy
Finished writing RootString.etm
The RootString.etm will have bucket elements in each object of the profile
hierachy.
```

17. The following is an example of the manual offline mode for XML report:

```
$tprof -A -x sleep 10
Starting Command sleep 10
stopping trace collection.
Tue Apr 17 22:28:01 2007
System: AIX 5.3 Node: test105 Machine: 00CEBB4A4C00
Generating sleep.trc
Generating sleep.prof
Generating sleep.syms
```

To run the **tprof** to use the **sleep.trc** and **sleep.syms** to generate XML report, enter the following to specify the **-r sleep** to generate XMLl report:

```
$tprof -X -r sleep
Calling tprof2xml to generate XML report.
tprof2xml TraceReader Version 1.2.0
Tue Apr 17 22:28:01 2007
System: AIX 6.1 Node: test105 Machine: 00CEBB4A4C00
------------------0------------------
Record 0
Post-processing counters
Retrieving Disassembly
writing the XML
Writing symbol list
.
Writing process hierarchy
Finished writing sleep.etm
```

18. For large page analysis of the workload and its descendants, enter the following command:

```
$tprof -a -y workload
```

The output is similar to the following display:

```
Starting Command workload
stopping trace collection.
Tue Apr 26 14:42:44 2005
System: AIX 5.3 Node: bigdomino Machine: 00C0046A4C00
Generating workload.trc
Generating workload.prof
Generating workload.syms
```

19. To profile only the specified program `workload` and its descendents, enter the following command:

```
$tprof -E PM_MRK_LSU_FIN -f 20000 –Aske –y workload
```

The output is similar to the following display:

```
Starting Command workload
stopping trace collection.
Tue Apr 26 16:42:44 2005
System: AIX 5.3 Node: bigdomino Machine: 00C0046A4C00
Generating workload.trc
Generating workload.prof
Generating workload.syms
```

20. To enable Java source line and instructions collecting for the application HelloAIX that is running on Java 6 JVM in realtime mode, enter the following command:

```
$ tprof -N -I -x java -agentlib:jpa=source=1,instructions=1 Hello AIX
```

Note: When a 64-bit JDK is used, enter the **-agentlib:jpa64** command instead of **-agentlib:jpa** in the following format:

```
$ tprof -N -I -x java -agentlib:jpa64=source=1,instructions=1 Hello AIX
```

The output is similar to the following display:

```
Fri May 30 04:16:27 2008
System: AIX 6.1 Node: toolbox2 Machine: 00CBA6FE4C00
Starting Command java -agentlib:jpa=source=1,instructions=1 Hello AIX
Hello AIX!
stopping trace collection.
Generating java.prof
```

The **java.prof** file is generated. It contains the JIT source line information and JIT instructions.

21. To displays the address bytes information in the report by using the **-O** *showaddrbytes=on* flag, enter the following command:

```
$ tprof  -O showaddrbytes=on -x sleep 5
```

A report similar to the following example is displayed:

```
Subroutine                      %   Source               Address  Bytes
==========                    ====== ======               =======  =====
h_cede_end_point               98.47 hcalls.s              111bfc     14

Sample report without -O showaddrbytes=on option

Subroutine                      %   Source
==========                    ====== ======
h_cede_end_point               98.47 hcalls.s
```

22. To display the process for trace data between 02/18/2016 02:30:30 and 02/18/2016 02:35:30 by using the **-G** option, enter the following command:

```
$tprof  -G “start=021802303016,end=021802353016” -r sleep
```

To process trace data starting from 02/18/2016 02:30:30 till the end, enter the following command:

```
$tprof -G “start= 021802303016” -r sleep
```

To process trace data from start and until 02/18/2016 02:35:30, enter the following command:

```
$tprof -G “end=021802303517” -r sleep
```

23. In the following example, the function name is

```
Test::abcdefghijklmnoprstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789
```

. To display how to line wrap long function names by using the **-O wrapfname=on** option, enter the following command:

```
$tprof -ukesl -O wrapfname=on -x sleep 5

The following is a sample report:
.Test::abcdefghijklmnoprstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVW
XYZ123456789                          215  19.40 test. C
```

The following is a sample report without using the **-O** *wrapfname=on* option:

```
Test::abcdefghijklmnoprstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ123456789_0abcdefghijk
```

```
lmnoprstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890(int,int)  215  19.40 test. C
```

Messages

If your system displays the following message:

```
/dev/systrace: device busy or trcon: TRCON:no such device
```

This means the **trace** facility is already in use. Stop your program and try again after typing `trcstop`, stops the trace.

tput Command

Purpose

Queries the **terminfo** database for terminal-dependent information.

Syntax

For Outputting Terminal Information

tput [**-T** *Type*] [*CapabilityName {clear, init, longname, reset}* [*Parameters*...]]

For Using stdin to Process Multiple Capabilities

tput [**-S**]

Description

The **tput** command uses the **terminfo** database to make terminal-dependent information available to the shell. The **tput** command outputs a string if the attribute *CapabilityName* is of type *string.* The output string is an integer if the attribute is of type *integer*. If the attribute is of type *Boolean*, the **tput** command sets the exit value (0 for TRUE, 1 for FALSE), and produces no other output.

XTERM DESCRIPTION LIMITATION

The xterm terminal description in the DEC.TI file on AIX Version 4 provides underline mode by using the SGR attribute. The SMUL and RMUL attributes are not currently defined in the XTERM terminal description on AIX Version 4. Use the more generic capability named SGR.

```
tput sgr x y
```

Where *x* is either a 1 or a 0 to turn standout mode on or off respectively, and *y* is either a 1 or a 0 to turn underline mode on or off respectively. See the article "**terminfo** file format" for more details on the SGR capability.

```
tput sgr 0 1     turn off standout; turn on underline
tput sgr 0 0     turn off standout; turn off underline
```

```
tput sgr 1 1     turn on standout; turn on underline
tput sgr 1 0     turn on standout; turn off underline
```

Flags

In addition to the capability names, the following strings are supported as arguments to the **tput** subroutine.

Item	Description
clear	Displays the clear screen sequence (this is also a capability name).
init	Displays the sequence that initializes the user's terminal in an implementation-dependent manner.
reset	Displays the sequence that will reset the user's terminal in an implementation-dependent manner.
longname	Displays the long name and the specified terminal (or current terminal if none specified).
-S	Uses stdin. This allow the tput to process multiple capabilities. When using the -S option, the capabilities cannot be entered on the command line. Enter ^D token finished.
-T*Type*	Indicates the type of terminal. If **-T** is not specified, the **TERM** environment variable is used for the terminal.

Exit Status

This command returns the following exit values:

Item	Description
0	The requested string was written successfully.
1	Unspecified.
2	Usage error.
3	No information is available about the specified terminal type.
4	The specified operand is invalid.
>4	An error occurred.

Examples

1. To clear the screen for the current terminal, enter:

   ```
   tput clear
   ```

2. To display the number of columns for the current terminals, enter:

   ```
   tput cols
   ```

3. To display the number of columns for the aixterm terminal, enter:

   ```
   tput  -Taixterm cols
   ```

4. To set the shell variable **bold** to the begin standout mode sequence and the shell variable **offbold** to the end standout mode sequence, enter:

   ```
   bold=`tput smso`
   offbold='tput rmso'
   ```

   Entering these commands might be followed by the following prompt:

```
echo "${bold}Name: ${offbold} \c"
```

5. To set the exit value to indicate if the current terminal is a hardcopy terminal, enter:

```
tput hc
```

6. To initialize the current terminal, enter:

```
tput init
```

Files

Item	Description
/usr/share/lib/terminfo/?/*	Contains the terminal descriptor files.
/usr/include/term.h	Contains the definition files.

tr Command

Purpose

Transforms characters or range of characters.

Syntax

To transform characters or sequence of characters:

tr [**-c** | **-cds** | **-cs** | **-C** | **-Cds** | **-Cs** | **-ds** | **-s**] [**-A**] *String1 String2*

To delete characters or sequence of characters:

tr { **-cd** | **-cs** | **-Cd** | **-Cs** | **-d** | **-s** } [**-A**] *String1*

To define a range of characters for a specific locale:

[LANG=*ll_RR*] **tr** **-L** {c1-cn C1-Cn} {c1c2c3c4c5c6...cn C1C2C3C4C5C6...Cn}

To delete a range of characters that is specified by the user:

[LANG=*ll_RR*] **tr** **-L** {c1-cn C1-Cn}

Description

The **tr** command deletes or substitutes characters from standard input and writes the result to standard output. The **tr** command also defines a range of characters for a specific locale. The **tr** command performs the following types of operations depending on the strings specified by the *String1* and *String2* variable and depending on the flags specified by the user.

Transforming Characters

If *String1* and *String2* are both specified and the **-d** flag is not specified, the **tr** command replaces each character contained in *String1* from the standard input with the character in the same position in *String2*.

Deleting Characters Using the -d Flag

If the **-d** flag is specified, the **tr** command deletes each character contained in *String1* from standard input.

Removing Sequences Using the -s Flag

If the **-s** flag is specified, the **tr** command removes all but the first character in any sequence of a character string represented in *String1* or *String2*. For each character represented in *String1*, the **tr** command removes all but the first occurrence of the character from standard output. For each character

represented in *String2*, the **tr** command removes all but the first occurrence in a sequence of occurrences of that character in the standard output.

Defining a character range in the current locale environment

You can specify a character range for specific locales, code sets, and users by using the **-L** flag of the **tr** command. The **tr** command retains and recognizes the character range that you specified. The **tr** command uses this information to transform the characters to the mapped characters in the range. The character range `c1-cn` that is specified with the **-L** flag is mapped to characters `c1c2c3c4c5c6...cn`, and the specified character range `C1-Cn` is mapped to characters `C1C2C3C4C5C6...Cn`.

Special Sequences for Expressing Strings

The strings contained in the *String1* and *String2* variables can be expressed using the following conventions:

Item	Description
C1-*C2*	Specifies the string of characters that collate between the character specified by *C1* and the character specified by *C2*, inclusive. The character specified by *C1* must collate before the character specified by *C2*. **Note:** The current locale has a significant effect on results when specifying subranges using this method. If the command is required to give consistent results irrespective of locale, the use of subranges should be avoided.
[*C******Number***]**	*Number* is an integer that specifies the number of repetitions of the character specified by *C*. *Number* is considered a decimal integer unless the first digit is a 0; then it is considered an octal integer.
[*C****]**	Fills out the string with the character specified by *C*. This option, used only at the end of the string contained within *String2*, forces the string within *String2* to have the same number of characters as the string specified by the *String1* variable. Any characters specified after the ***** (asterisk) are ignored.
[:*ClassName***:]**	Specifies all of the characters in the character class named by *ClassName* in the current locale. The class name can be any of the following names: `alnum  lower` `alpha  print` `blank  punct` `cntrl  space` `digit  upper` `graph  xdigit` Except for **[:**`lower`**:]** and **[:**`upper`**:]** conversion character classes, the characters specified by other character classes are placed in an array in an unspecified order. Because the order of the characters specified by character classes is undefined, the characters should be used only if the intent is to map several characters into one. An exception to this is the case of conversion character classes. For more information on character classes, see the **ctype** subroutines.
[=*C***=]**	Specifies all of the characters with the same equivalence class as the character specified by *C.*
*****Octal	Specifies the character whose encoding is represented by the octal value specified by *Octal*. An Octal value can be a one digit, two digit, or three digit octal integer. The NULL character can be expressed by using the `'\0'` expression, and is processed like any other character.

Item	Description
*****ControlCharacter*	Specifies the control character that corresponds to the value specified by *ControlCharacter*. The following values can be represented: **\a** Alert **\b** Backspace **\f** Form-feed **\n** New line **\r** Carriage return **\t** Tab **\v** Vertical tab
****	Specifies the \ (backslash) as itself, without any special meaning as an escape character.
\[	Specifies the [(left bracket) as itself, without any special meaning as the beginning of a special string sequence.
\-	Specifies the - (minus sign) as itself, without any special meaning as a range separator.

If a character is specified more than once in *String1*, the character is transformed into the character in *String2* that corresponds to the last occurrence of the character in *String1*.

If the strings specified by *String1* and *String2* are not the same length, the **tr** command ignores the extra characters in the longer string.

Flags

Item	Description
-A	Performs all operations on a byte-by-byte basis using the ASCII collation order for ranges and character classes, instead of the collation order for the current locale.
-C	Specifies that the value of *String1* be replaced by the *complement* of the string specified by *String1*. The complement of *String1* is all of the characters in the character set of the current locale, *except* the characters specified by *String1*. If the **-A** and **-c** flags are both specified, characters are complemented with respect to the set of all 8-bit character codes. If the **-c** and **-s** flags are both specified, the **-s** flag applies to characters in the complement of *String1*. If the **-d** option is not specified, the complements of the characters specified by *String1* will be placed in the array in ascending collation sequence as defined by the current setting of **LC_COLLATE**.

Item	Description
-c	Specifies that the value of *String1* be replaced by the *complement* of the string specified by *String1*. The complement of *String1* is all of the characters in the character set of the current locale, *except* the characters specified by *String1*. If the **-A** and **-c** flags are both specified, characters are complemented with respect to the set of all 8-bit character codes. If the **-c** and **-s** flags are both specified, the **-s** flag applies to characters in the complement of *String1*. If the **-d** option is not specified, the complement of the values specified by *String1* will be placed in the array in ascending order by binary value.
-d	Deletes each character from standard input that is contained in the string specified by *String1*. **Note:** 1. When the **-C** option is specified with the **-d** option, all characters except those specified by *String1* will be deleted. The contents of *String2* are ignored unless the **-s** option is also specified. 2. When the **-c** option is specified with the **-d** option, all values except those specified by *String1* will be deleted. The contents of *String2* are ignored unless the **-s** option is also specified.
-L	Adds a user-defined character range in the current locale environment to the $*HOME*`/.trregexrc/$`*CODESET* file. The character range `c1-cn` is mapped to `c1c2c3c4c5c6...cn`, and the character range `C1-Cn` is mapped to `C1C2C3C4C5C6...Cn`. The **-L** flag is user-specific (depends on the $*HOME* variable), code-set specific, and locale-specific (depends on the $*LANG* variable). It means that you must define the character range for specific users, code sets, and locales. If the $*HOME*`/.trregexrc/$`*CODESET* file does not exist for a specific user or locale, the file is automatically generated when you specify the **-L** flag.
-s	Removes all but the first character in a sequence of repeated characters. Character sequences specified by *String1* are removed from standard input before translation, and character sequences specified by *String2* are removed from standard output.
String1	Specifies a string of characters.
String2	Specifies a string of characters.

Exit Status

This command returns the following exit values:

0
All input was processed successfully.

>0
An error occurred.

Examples

1. To transform braces into parentheses, enter the following command:

   ```
   tr '{}' '()' < textfile > newfile
   ```

   This transforms each { (left brace) to ((left parenthesis) and each } (right brace) to) (right parenthesis). All other characters remain unchanged.
2. To transform braces into brackets, enter the following command:

```
tr '{}' '\[]' < textfile > newfile
```

This transforms each { (left brace) to [(left bracket) and each } (right brace) to] (right bracket). The left bracket must be entered with a \ (backslash) escape character.

3. To transform lowercase characters to uppercase, enter the following command:

```
tr 'a-z' 'A-Z' < textfile > newfile
```

4. To create a list of words in a file, enter the following command:

```
tr -cs '[:lower:][:upper:]' '[\n*]' < textfile > newfile
```

This transforms each sequence of characters other than lowercase letters and uppercase letters into a single newline character. The * (asterisk) causes the **tr** command to repeat the new line character enough times to make the second string as long as the first string.

5. To delete all NULL characters from a file, enter the following command:

```
tr -d '\0' < textfile > newfile
```

6. To replace every sequence of one or more new lines with a single new line, enter the following command:

```
tr -s '\n' < textfile > newfile
```

OR

```
tr -s '\012' < textfile > newfile
```

7. To replace every nonprinting character, other than valid control characters, with a ? (question mark), enter the following command:

```
tr -c '[:print:][:cntrl:]' '[?*]' < textfile > newfile
```

This scans a file created in a different locale to find characters that are not printable characters in the current locale.

8. To replace every sequence of characters in the <space> character class with a single # character, enter the following command:

```
tr -s '[:space:]' '[#*]'
```

9. To define a character range for a specific locale, enter the following command:

```
LANG=ES_ES tr -L a-z A-Z abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

This command defines the a-z and A-Z character ranges for the ES_ES (Spanish_Spain) locale.

10. To delete a character range for a specific locale, enter the following command:

```
LANG=ES_ES tr -L a-z A-Z
```

trace Daemon

Purpose

Records selected system events.

Syntax

trace [**-a** [**-g**]] [**-f** | **-l**] [**-b** | **-B**] [**-c**] [**-C** [*CPUList* | **all**]] [**-d**] [-e *string-cmd*] [**-h**] [**-j** *EventList*] [**-k** *EventgroupList*] [**-J** *EventgroupList*] [**-K** *EventgroupList*] [**-m** *Message*] [-M] [-N] [**-n**]

[**-o** *Name*] [**-o-**] [**-p**] [`-r` *reglist*] [**-s**] [**-A** *ProcessIDList*] [**-t** *ThreadIDList*] [**-x** *program-specification* | **-X** *program-specification*] [**-I**] [**-P** *trace-propagation*][**-L** *Size*] [**-T** *Size*] [**-W**] [**-@** *WparList*]

Description

The **trace** daemon configures a trace session and starts the collection of system events. The data collected by the trace function is recorded in the trace log. A report from the trace log can be generated with the **trcrpt** command.

When invoked with the **-a**, **-x**, or **-X** flags, the trace daemon is run asynchronously (for example, as a background task). Otherwise, it is run interactively and prompts you for subcommands.

To put the WPARconfigured ID (CID) in the trace hooks, use the **-W** flag.

To trace specific WPAR, use the **-@** flag with a list of WPAR names that you want to trace.

You can use the System Management Interface Tool (SMIT) to run the **trace** daemon. To use SMIT, enter:

```
smit trace
```

The following are modes of trace data collection:

Item	Description
Alternate (the default)	All trace events are captured in the trace log file.
Circular (**-l**)	The trace events wrap within the in-memory buffers and are not captured in the trace log file until the trace data collection is stopped.
Single (**-f**)	The collection of trace events stops when the in-memory trace buffer fills up and the contents of the buffer are captured in the trace log file.
Buffer Allocation	Trace buffers are allocated from either the kernel heap, or are put into separate segments. By default, buffers are allocated from the kernel heap unless the buffer size requested is too large for buffers to fit in the kernel heap, in which case they are allocated in separate segments. Allocating buffers from separate segments hinders trace performance somewhat. However, buffers in separate segments will not take up paging space, just pinned memory. The type of buffer allocation can be specified with the optional **-b** or **-B** flags.

You can elect to trace only selected processes or threads. You can also trace a single program. You can specify whether the trace is to be propagated or extended to newly created processes or threads. You can optionally include interrupt events in such traces. This is only valid for trace channel `0`.

Note:

1. Unless the trace is started before the process that is being traced, the process startup events are not captured. If the trace is started before the process that is being traced, some events from processes other than the process being traced will be captured as well.
2. When trace uses memory from the kernel heap which is the case for the `-B` option (32-bit kernel only), this memory remains part of kernel memory until the next reboot of the system. Thus, care should be taken when using large buffers.

Flags

Item	Description
-@ *WparList*	Traces the workload partitions that you specify in the *WparList* parameter. Multiple WPAR names can either be separated by commas or enclosed in quotation marks and separated by spaces. To include the current Global system in the trace, specify Global. You can only specify the **-@** flag in the Global system in a workload partition environment.
-a	Runs the **trace** daemon asynchronously (i.e. as a background task). Once **trace** has been started this way, you can use the **trcon**, **trcoff**, and **trcstop** commands to respectively start tracing, stop tracing, or exit the trace session. These commands are implemented as links to **trace.**
-A *ProcessIDList*	Traces only the processes and, optionally, their children specified with the *ProcessIDList*. A process ID is a decimal number. Multiple process IDs can either be separated by commas or enclosed in quotation marks and separated by spaces. The **-A** flag is only valid for trace channel 0; the **-A** and **-g** flags are incompatible. All threads existing for the specified processes when tracing is started are traced. By default, if after the trace starts, the processes being traced create additional threads or processes, these are not traced unless the **-P** flag is specified.
-b	Allocate buffers from the kernel heap. If the requested buffer space can not be obtained from the kernel heap, the command fails. **Restriction:** The **-b** flag is only valid with the 32–bit kernel.
-B	Allocate buffers in separate segments. **Restriction:** The **-B** flag is only valid with the 32–bit kernel.
-c	Saves the trace log file, adding **.old** to its name.

Item	Description
-C [*CPUList* | **all**]	Traces using one set of buffers per processor in the *CPUList*. The processors can be separated by commas, or enclosed in double quotation marks and separated by commas or blanks. To trace all processors, specify **all**. Since this flag uses one set of buffers per processor, and produces one file per processor, it can consume large amounts of memory and file space, and should be used with care. The files produced are named **trcfile**, **trcfile-0**, **trcfile-1**, etc., where **0**, **1**, etc. are the processor numbers. If **-T** or **-L** are specified, the sizes apply to each set of buffers and each file. On a uniprocessor system, you may specify `-C all`, but `-C` with a list of processor numbers is ignored.  **Attention:** The -C flag can only be used by the root user.
-d	Disables the automatic start of trace data collection. Delays starting of trace data collection. Normally, the collection of trace data starts automatically when you issue the **trace** daemon. Use the **trcon** command to start the collection of trace data.
-e *string-cmd*	Configures Component Trace by running `ctctrl` with *string-cmd* as an argument before the trace is started. In other words, it runs `ctctrl` *string-cmd*. Passing multiple `-e` options is allowed and is equivalent to successively running the `ctctrl` command with each *string-cmd* of arguments. This option can be used to configure the system trace mode (by setting the system trace mode to On, changing the level of trace, and so on) for some components just before starting to trace the system.
-f	Runs **trace** in a single mode. Causes the collection of trace data to stop as soon as the in-memory buffer is filled up. The trace data is then written to the trace log. Use the **trcon** command to restart trace data collection and capture another full buffer of data. If you issue the **trcoff** subcommand before the buffer is full, trace data collection is stopped and the current contents of the buffer are written to the trace log.
-g	Starts a trace session on a generic trace channel (channels 1 through 7). This flag works only when **trace** is run asynchronously (**-a**). The return code of the command is the channel number; the channel number must subsequently be used in the generic trace subroutine calls. To stop the generic trace session, use the command **trcstop -<**channel_number>.

Item	Description
-h	Omits the header record from the trace log. Normally, the **trace**daemon writes a header record with the date and time (from the **date** command) at the beginning of the trace log; the system name, version and release, the node identification, and the machine identification (from the **uname -a** command); and a user-defined message. At the beginning of the trace log, the information from the header record is included in the output of the **trcrpt** command.
-I	Trace interrupt events. When specified with **-A** or **-t**, the **-I** flag includes interrupt events along with the events for the processes or threads specified. When -I is specified, but neither **-A** nor **-t** is specified, only interrupt level events are traced. The **-I** flag is only valid for trace channel 0; the **-I** and **-g** flags are incompatible.
-j *EventList*	Specifies the user-defined events to collect trace data. The list items specified in the *EventList* parameter can either be separated by commas or enclosed in quotation marks and separated by commas or spaces. In AIX 6.1 and earlier releases, specifying a two-digit hook ID in the form **hh** specifies **hh00, hh10,...,hhF0**. Specifying a three-digit hook ID in the form **hhh** specifies **hhh0**. Specifying a four-digit hook ID in the form **hhhh** specifies **hhhh**. If any of these events is missing, the information reported by the **trcrpt** command will be incomplete. Consequently, when using the **-j** flag, include all these events in the *EventList*. If starting the trace with SMIT, or the **-J** flag, these events are in the **tidhk** group.
-J *EventgroupList*	Specifies the event groups to be included. The list items specified in the *EventgroupList* parameter can either be separated by commas or enclosed in quotation marks and separated by commas or spaces. The **-J** and **-K** flags work like **-j** and **-k**, except with event groups instead of individual hook IDs. You can specify each flag **-j**, **-J**, **-k**, and **-K** within the command.

Item	Description
-k *EventgroupList*	Specifies the user-defined events to exclude trace data. The list items specified in the *EventgroupList* parameter can either be separated by commas or enclosed in quotation marks and separated by commas or spaces. In AIX 6.1 and earlier releases, specifying a two-digit hook ID in the form **hh** specifies **hh00, hh10,...,hhF0**. Specifying a three-digit hook ID in the form **hhh** specifies **hhh0**. Specifying a four-digit hook ID in the form **hhhh** specifies **hhhh**. **Tip:** The following events are used to determine the **pid**, the **cpuid**, and the **exec** path name in the **trcrpt** report: `106 DISPATCH` `10C DISPATCH IDLE PROCESS` `134 EXEC SYSTEM CALL` `139 FORK SYSTEM CALL` `465 KTHREAD CREATE` If any of these events is missing, the information reported by the **trcrpt** command will be incomplete. When using the **-k** flag, do not include these events in the *EventgroupList* parameter. If starting the trace with SMIT, or the **-J** flag, these events are in the **tidhk** group.
-K *EventgroupList*	Specifies the event groups to be excluded. The list items specified in the *EventgroupList* parameter can either be separated by commas or enclosed in quotation marks and separated by commas or spaces. The **-J** and **-K** flags work like **-j** and **-k**, except with event groups instead of individual hook IDs. You can specify each flag **-j**, **-J**, **-k**, and **-K** within the command.
-l	Runs **trace** in a circular mode. The **trace** daemon writes the trace data to the trace log when the collection of trace data is stopped. Only the last buffer of trace data is captured. When you stop trace data collection using the **trcoff** command, restart it using the **trcon**command.
-L *Size*	Overrides the default trace log file size of 1 MB with the value stated. Specifying a file size of zero sets the trace log file size to the default size. **Note:** In the circular and the alternate modes, the trace log file size must be at least twice the size of the trace buffer. In the single mode, the trace log file must be at least the size of the buffer. See the **-T** flag for information on controlling the trace buffer size.
-m *Message*	Specifies text to be included in the message field of the trace log header record.

Item	Description
-M	Dumps the address map of running processes into the trace. The **-M** flag must be specified if the trace file is to be processed by the `tprof` command.
-n	Adds information to the trace log header: lock information, hardware information, and, for each loader entry, the symbol name, address, and type.
-N	Dump the address map of specified processes into the trace. The **-N** option is used in conjunction with **-M** option.
-o *Name*	Overrides the **/var/adm/ras/trcfile** default trace log file and writes trace data to a user-defined file.
-o -	Overrides the default trace log name and writes trace data to standard output. The **-c** flag is ignored when using this flag. An error is produced if **-o -** and **-C** are specified.
-p	Includes the cpuid of the current processor with each hook. This flag is only valid for 64-bit kernel traces. **Note:** The **trcrpt** command can report the cpuid whether or not this option is specified.
-P *propagation*	The propagation is specified with the letters p for propagation across process creation, `t` for propagation across thread creation, and n for no propagation. Propagation across process creation implies propagation across thread creation. For example, if **-A** is specified to trace a process, all threads for that process that exist at the time the trace was started are traced. The **-Pt** flags causes all threads subsequently created by that process to be traced as well. If **-Pp** is specified, all processes and threads subsequently created by that process are traced. If **-t all** was specified to trace all threads, **-P** is ignored. The **-P** flag is only valid for trace channel `0`; the **-P** and **-g** flags are incompatible.

Item	Description
`-r` *reglist*	Optional, and only valid for a `trace` run on a 64-bit kernel. *reglist* items are separated by commas, or enclosed in quotation marks and separated by blanks. Up to 8 registers may be specified. Valid *reglist* values are: **PURR - The PURR** Register for this processor **SPURR** The SPURR register for this processor **MCR0, MCR1, MCRA - the MCR** Registers, 0, 1, and A **PMC1, PMC2, ... PMC8 - PMC** Registers 1 through 8. **Restriction:** Not all registers are valid for all processors.
-s	Stops tracing when the trace log fills. The **trace** daemon normally wraps the trace log when it fills up and continues to collect trace data. During asynchronous operation, this flag causes the **trace** daemon to stop trace data collection. (During interactive operation, the **quit** subcommand must be used to stop trace.)
-t *ThreadIDList*	Traces only the threads specified with the *ThreadIDList* parameter. A thread ID is a decimal number. Multiple thread IDs can either be separated by commas or enclosed in quotation marks and separated by spaces. Also, the thread list can be `all` or `*`, indicating that all threads are to be traced. This is useful for tracing all thread-related events without tracing interrupt-related events. However, if **-t all** and **-I** are both specified, this is the same as specifying neither one; all events are traced. Another way to say this is that **trace** and **trace -It all** are identical. The **-t** flag is only valid for trace channel `0`, the **-t** and **-g** flags are incompatible.

Item	Description
-T *Size*	Overrides the default trace buffer size of 128 KB with the value stated. You must be root to request more than 1 MB of buffer space. The maximum possible size is 268435184 bytes, unless the **-f** flag is used, in which case it is 536870368 bytes. The smallest possible size is 8192 bytes, unless the **-f** flag is used, in which case it is 16392 bytes. Sizes between 8192 and 16392 will be accepted when using the **-f** flag; however, the actual size used will be 16392 bytes. **Note:** In the circular and the alternate modes, the trace buffer size must be one-half or less the size of the trace log file. In the single mode, the trace log file must be at least the size of the buffer. See the **-L** flag for information on controlling the trace log file size. Also note that trace buffers use pinned memory, which means they are not pageable. Therefore, the larger the trace buffers, the less physical memory is available to applications. Unless the **-b** or **-B** flags are specified, the system attempts to allocate the buffer space from the kernel heap. If this request can not be satisfied, the system then attempts to allocate the buffers as separate segments. The **-f** flag actually uses two buffers, which behave as a single buffer (except that a buffer wraparound trace hook will be recorded when the first buffer is filled).
-W	Use the **-W** flag to include the workload partitionconfigured ID (CID) for the current process with each hook. This flag is only valid in the Global system in a workload partition environment. **Tip:** The **trcrpt** command can report the workload partitionCID whether or not this option is specified.

Item	Description
-x *program-specification*	Traces the specified program. The *program-specification* specifies a program and parameters as they would be when running the program from the shell, except that the program specification must be in quotes if more than just the program's name is given. The trace is stopped automatically when the program exits, and returns the program's return code. By default, any processes and threads created by the program are also traced; as if **-Pp** was specified. To change this behavior, use **-Pn** to specify no trace propagation, or **-Pt** to propagate trace only to threads created by the program's original process. **Tip:** The **-x** flag implies asynchronous tracing, as if the **-a** flag had also been specified.
-X *program-specification*	The **-X** flag works like the -x flag, except that the trace is not automatically stopped when the program exits. This is useful for tracing programs which fork processes, and then terminate, and you want these new processes traced as well.

Subcommands

When run interactively, trace recognizes the following subcommands:

Item	Description
trcon	Starts the collection of trace data.
trcoff	Stops the collection of trace data.
q or **quit** [-serial | -dd]	Stops the collection of trace data and exits **trace**. If the **-s** option is specified then this serializes any pending I/O operations. If the **-d** option is specified, any pending I/O operation is discarded.
! *Command*	Runs the shell command specified by the *Command* parameter.
?	Displays the summary of **trace** subcommands.

Signals

The **INTERRUPT** signal acts as a toggle to start and stop the collection of trace data. Interruptions are set to **SIG_IGN** for the traced process.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

To run the **trace** command on channel 0, you must have the following additional authorizations, if channel 0 **trace** restriction is enabled in the **trcctl** command:

Item	Description
aix.ras.trace.tracech0	Required to run the **trace** command on channel 0.

Note: By default, the root and system group users are privileged users.

To perform all functionality of all commands including the **trace** command on channel 0, you must have the following additional authorizations:

Item	Description
aix.ras.trace.trace	Required to perform all trace operations.

Examples

1. To use trace interactively, enter `trace`, (the > prompt is displayed), then specify the subcommands you want. For example, to trace system events during the run of the *anycmd* command, enter:

   ```
   trace
   > !anycmd
   > q
   ```

2. To avoid delays when the command finishes, start trace asynchronously (**-a**), using only one command line, enter:

   ```
   trace -a; anycmd; trcstop
   ```

3. To trace the system itself for a period of 10 seconds, enter:

   ```
   trace -a; sleep 10; trcstop
   ```

4. To output trace data to a specific trace log file (instead of the **/var/adm/ras/trcfile** default trace log file), :

   ```
   trace -a -o /tmp/my_trace_log; anycmd; trcstop
   ```

5. To capture the execution of a **cp** command, excluding specific events from the collection process:

   ```
   trace -a -k "20e,20f" -x "cp /bin/track /tmp/junk"
   ```

   In the example above, the **-k** option suppresses the collection of events from the **lockl** and **unlockl** functions (20e and 20f events).

   Also notice that the **-x** flag was used, so only hooks associated with the **cp** command process will be traced, and no interrupt activity will be traced.

6. To trace hook 234 and the hooks that will allow you to see the process names, use:

   ```
   trace -a -j 234 -J tidhk
   ```

   This traces the hooks in the event-group "tidhk" plus hook 234.

7. To have trace use one set of buffers per processor, specify:

   ```
   trace -aC all
   ```

   The files produced are **/var/adm/ras/trcfile**, **/var/adm/ras/trcfile-0**, **/var/adm/ras/trcfile-1**, etc. up to **/var/adm/ras/trcfile-**(*n*-1), where *n* is the number of procssors in the system.

   Tip: `trace -aCall -o mylog` produces the files mylog, mylog-0, mylog-1, ...

8. To trace a program that starts a daemon process, and to continue tracing the daemon after the original program has finished, use

   ```
   trace -X "mydaemon"
   ```

The trace must be stopped with **trcstop**.

9. To trace *mydaemon*, which is currently running, use:

```
trace -A mydaemon-process-id -Pp
```

   Where *mydaemon-process-id* is the process for *mydaemon* as returned by the **ps** command. The **-Pp** flag tells trace to also trace any processes and threads created by *mydaemon* while the trace is running.

10. To capture the PURR, and PMC1 and PMC2, type:

```
trace -ar "PURR PMC1 PMC2"
```

11. To trace hooks 1A00,1A10,...,1AF0, DCA0 and 1AB1, enter:

```
trace -aj 1A,DCA,1AB1
```

Files

Item	Description
/usr/include/sys/trcmacros.h	Defines **trchook** and **utrchook** macros.
/var/adm/ras/trcfile	Contains the default trace log file.

traceauth Command

Purpose

Trace the authorizations that a command needs to run successfully.

Syntax

traceauth [-d] [-e] [-f] [-o *outputfile*] *Command* [args]

Description

The **traceauth** command records the authorizations that a command attempts to use when the command is run. There are two ways an authorization can be used. The first way is the **accessauths** attribute that grants access to run a specified program. The second way is the **checkauths** attribute that is checked in a program before performing a privileged operation. The **traceauth** command can trace and report both types of authorizations. The **traceauth** command is used either for command investigation when entries are added to the privileged command database or to identify which authorizations to use while creating a role. The **traceauth** command runs the command specified by the *Command* parameter, along with associated arguments for the *Command*.

Generally, run the **traceauth** command with the PV_ROOT privilege or by assuming a role that has **aix** authorization so that any attempt to use authorization would succeed. In this case, the traceauth command can keep track of all of the authorizations that the command specified in the *Command* parameter needs for a successful run without the PV_ROOT privilege or a special role. After the command specified in the *Command* parameter is run, the list of used **accessauths** and **checkauths** are written to the standard output (stdout) file.

Flags

Item	Description
-d	Display the output of the truss command with the authorizations that are required by the command.

Item	Description
-e	Follow the **exec** subroutine. If the command specified by the *Command* parameter runs an **exec** subroutine, the **traceauth** command reports the authorizations needed so far, and then proceeds with recording the authorizations associated with the new executable file. If the file run by the **exec** subroutine has its setuid bit set and is not owned by root, the **traceauth** command cannot properly trace the authorizations use of the file.
-f	Follow the **fork** subroutine. If the controlled process calls the **fork** subroutine, the traceauth command also reports the authorizations used by the new child process.
-o	Write the output to the specified file instead of the standard output (stdout) file.

Parameters

Item	Description
args	Specifies the arguments for the associated command in the *Command* parameter.
Command	Specifies the name of the command whose authorizations you want to trace.
outputfile	If you do not want to write the output to the standard output (stdout) file, use the **-o** flag. Then, specify the name of the output file to which you want to record the authorizations in the *outputfile* parameter.

tracepriv Command

Purpose

Traces the privileges that a command needs for a successful run.

Syntax

tracepriv [-**d**] [-**e**] [-**f**] [**-o** *outputfile*] *Command* [*args*]

Description

The **tracepriv** command records the privileges that a command attempts to use when the command is run. The **tracepriv** command is used for command investigation when entries are added to the privileged command database. The **tracepriv** command runs the command specified by the *Command* parameter with the specified arguments (with the *args* parameter). Generally, run the **tracepriv** command with the PV_ROOT privilege so that any attempt to use a privilege succeeds. In this case, the **tracepriv** command can keep track of all of the privileges that the *Command* needs for a successful run without the PV_ROOT privilege. After the *Command* is run or when an **exec** subroutine within the command occurs, the list of used privileges is written to standard output (**stdout**).

Flags

Item	Description
-d	Displays the output of the **truss** command with the privileges that is required by the command.

Item	Description
-e	Follows the **exec** subroutine. If the command specified by the *Command* parameter runs an **exec** subroutine, the **tracepriv** command reports the privileges needed so far (and set them if the **-a** flag is used), and then proceeds with recording (and setting) the privileges associated with the new executable file. If the file run by the **exec** subroutine has its setuid bit set and is not owned by root, the **tracepriv** command cannot properly trace the privilege use of the file.
-f	Follows the **fork** subroutine. If the controlled process calls the **fork** subroutine, the **tracepriv** command also reports the privileges used by the new child process.
-o	Writes the output to the specified file instead of the standard output (**stdout**).

Parameters

Item	Description
args	Specifies the arguments.
Command	Specifies the command.
outputfile	Specifies the file to record the output.

traceroute Command

Purpose

Prints the route that IP packets take to a network host.

Syntax

traceroute [**-m** *Max_ttl*] [**-n**] [**-p** *Port*] [**-q** *Nqueries*] [**-r**] [**-d**] [**-g** *gateway_addr*] [**-s** *SRC_Addr*] [**-t** *TypeOfService*] [**-f** *flow*] [**-v**] [**-w** *WaitTime*] *Host* [*PacketSize*]

Description

> **Attention:** The **traceroute** command is intended for use in network testing, measurement, and management. It should be used primarily for manual fault isolation. Because of the load it imposes on the network, the **traceroute** command should not be used during normal operations or from automated scripts.

The **traceroute** command attempts to trace the route an IP packet follows to an Internet host by launching UDP probe packets with a small maximum time-to-live (*Max_ttl* variable), then listening for an ICMP **TIME_EXCEEDED** response from gateways along the way. Probes are started with a *Max_ttl* value of one hop, which is increased one hop at a time until an ICMP **PORT_UNREACHABLE** message is returned. The ICMP **PORT_UNREACHABLE** message indicates either that the host has been located or the command has reached the maximum number of hops allowed for the trace.

The **traceroute** command sends three probes at each *Max_ttl* setting to record the following:

- *Max_ttl* value
- Address of the gateway
- Round-trip time of each successful probe

The number of probes sent can be increased by using the **-q** flag. If the probe answers come from different gateways, the command prints the address of each responding system. If there is no response from a probe within a 3-second time-out interval, an * (asterisk) is printed for that probe.

The **traceroute** command prints an ! (exclamation mark) after the round-trip time if the *Max_ttl* value is one hop or less. A maximum time-to-live value of one hop or less generally indicates an incompatibility in the way ICMP replies are handled by different network software. The incompatibility can usually be resolved by doubling the last *Max_ttl* value used and trying again.

Other possible annotations after the round-trip notation are:

Item	Description
!H	Host unreachable
!N	Network unreachable
!P	Protocol unreachable
!S	Source route failed
!F	Fragmentation needed

If the majority of probes result in an error, the **traceroute** command exits.

The only mandatory parameter for the **traceroute** command is the destination host name or IP number. The **traceroute** command will determine the length of the probe packet based on the Maximum Transmission Unit (MTU) of the outgoing interface. The UDP probe packets are set to an unlikely value so as to prevent processing by the destination host.

Flags

Item	Description
-d	Enables socket level debugging.
-f *flow*	Sets the flow label field in IPv6 packet header. The default value is 0.
-g *gateway_addr*	Routes the outgoing packets through a specified gateway with the IP source routing option. Before you use this flag, your router must enable IP source routing. This flag is only available for IP version 6 addresses.
-m *Max_ttl*	Sets the maximum time-to-live (maximum number of hops) used in outgoing probe packets. The default is 30 hops (the same default used for TCP connections).
-n	Prints hop addresses numerically rather than symbolically and numerically. This flag saves a name-server address-to-name lookup for each gateway found on the path.
-p *Port*	Sets the base UDP port number used in probes. The default is 33434. The **traceroute** command depends on an open UDP port range of *base* to *base* + *nhops* - 1 at the destination host. If a UDP port is not available, this option can be used to pick an unused port range.
-q *Nqueries*	Specifies the number of probes the **traceroute** command sends at each *Max_ttl* setting. The default is three probes.
-r	Bypasses the normal routing tables and sends the probe packet directly to a host on an attached network. If the specified host is not on a directly attached network, an error is returned. This option can be used to issue a **ping** command to a local host through an interface that is not registered in the **routed** daemon's routing table.

Item	Description
-s *SRC_Addr*	Uses the next IP address in numerical form as the source address in outgoing probe packets. On hosts with more than one IP address, the **-s** flag can be used to force the source address to be something other than the IP address of the interface on which the probe packet is sent. If the next IP address is not one of the machine's interface addresses, an error is returned and nothing is sent.
-t *TypeOfService*	Sets the *TypeOfService* variable in the probe packets to a decimal integer in the range of 0 to 255. The default is 0. This flag can be used to investigate whether different service types result in different paths. For more information, see TCP/IP Protocols in *Performance Tools Guide and Reference*. Useful values are **-t 16** (low delay) and **-t 8** (high throughput).
-v	Receives packets other than **TIME_EXCEEDED** and **PORT_UNREACHABLE** (verbose output).
-w *WaitTime*	Sets the time (in seconds) to wait for a response to a probe. The default is 3 seconds.

Parameters

Item	Description
Host	Specifies the destination host, either by host name or IP number. This parameter is required.
PacketSize	Specifies the probe datagram length. The default packet size is determined by the **traceroute** command based on the MTU of the outgoing interface.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. A sample use and output is:

```
[yak 71]% traceroute nis.nsf.net.
traceroute to nis.nsf.net (35.1.1.48), 30 hops max, 56 byte packet
 1 helios.ee.lbl.gov (128.3.112.1) 19 ms 19 ms 0 ms
 2 lilac-dmc.Berkeley.EDU (128.32.216.1) 39 ms 39 ms 19 ms
 3 lilac-dmc.Berkeley.EDU (128.32.216.1) 39 ms 39 ms 19 ms
 4 ccngw-ner-cc.Berkeley.EDU (128.32.136.23) 39 ms 40 ms 39 ms
 5 ccn-nerif22.Berkeley.EDU (128.32.168.22) 39 ms 39 ms 39 ms
 6 128.32.197.4 (128.32.197.4) 40 ms 59 ms 59 ms
 7 131.119.2.5 (131.119.2.5) 59 ms 59 ms 59 ms
 8 129.140.70.13 (129.140.70.13) 99 ms 99 ms 80 ms
 9 129.140.71.6 (129.140.71.6) 139 ms 239 ms 319 ms
10 129.140.81.7 (129.140.81.7) 220 ms 199 ms 199 ms
11 nic.merit.edu (35.1.1.48) 239 ms 239 ms 239 ms
```

   Lines 2 and 3 are the same due to a bug in the kernel on the second hop system (`lbl-csam.arpa`) that forwards packets with a zero time-to-live. Host names are not printed in lines 6 through 10 because the National Science Foundation Network (NSFNet, `129.140`) does not provide address-to-name translations for its nodes.

2. Another output example might be:

```
[yak 72]% traceroute rip.Berkeley.EDU (128.32.131.22)
traceroute to rip.Berkeley.EDU (128.32.131.22), 30 hops max
```

```
 1 helios.ee.lbl.gov (128.3.112.1) 0 ms 0 ms 0 ms
 2 lilac-dmc.Berkeley.EDU (128.32.216.1) 39 ms 19 ms 39 ms
 3 lilac-dmc.Berkeley.EDU (128.32.216.1) 19 ms 39 ms 19 ms
 4 ccngw-ner-cc.Berkeley.EDU (128.32.136.23) 39 ms 40 ms 19 ms
 5 ccn-nerif35.Berkeley.EDU (128.32.168.35) 39 ms 39 ms 39 ms
 6 csgw/Berkeley.EDU (128.32.133.254) 39 ms 59 ms 39 ms
 7 * * *
 8 * * *
 9 * * *
10 * * *
11 * * *
12 * * *
13 rip.Berkeley.EDU (128.32.131.22) 59 ms! 39 ms! 39 ms!
```

In this example, exactly half of the 12 gateway hops (13 is the final destination) are "missing." However, these hops were actually not gateways. The destination host, a Sun-3 workstation running Sun OS3.5, used the ttl from the arriving datagram as the ttl in its ICMP reply; thus, the reply timed out on the return path. Because ICMPs are not sent for ICMPs, no notice was received. The ! (exclamation mark) after each round-trip time indicates some type of software incompatibility problem. (The cause was diagnosed after the **traceroute** command issued a probe of twice the path length. The destination host was really only seven hops away.)

tracesoff Command

Purpose

Turns off tracing of a subsystem, a group of subsystems, or a subserver.

Syntax

Subsystem

tracesoff [**-h** *Host*] { **-g** *Group* | **-p** *SubsystemPID* | **-s** *Subsystem*}

Subserver

tracesoff [**-h** *Host*] **-t** *Type* [**-p** *SubsystemPID*] { **-o** *Object* | **-P** *SubserverPID* }

Description

The **tracesoff** command sends the System Resource Controller a subsystem request packet that is forwarded to the subsystem to turn tracing off. Tracing is unsuccessful if the communication method for the subsystems is signals.

Note: Tracing is subsystem dependent.

Flags

Item	Description
-g *Group*	Specifies a group of subsystems to turn tracing off. The command is unsuccessful if the *Group* name is not contained in the subsystem object class.
-h *Host*	Specifies the foreign host on which this trace action is requested. The local user must be running as root. The remote system must be configured to accept remote System Resource Controller requests. That is, the **srcmstr** daemon (see **/etc/inittab**) must be started with the **-r** flag and the **/etc/hosts.equiv** or **.rhosts** file must be configured to allow remote requests.
-o *Object*	Specifies that a subserver *Object* name is to be passed to the subsystem as a character string.

Item	Description
-p *SubsystemPID*	Specifies a particular instance of the subsystem to turn tracing off, or a particular instance of the subsystem to which the trace off subserver request is to be passed.
-P *SubserverPID*	Specifies that a *SubserverPID* is to be passed to the subsystem as a character string.
-s *Subsystem*	Specifies a subsystem to turn tracing off. The *Subsystem* name can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the *Subsystem* name is not contained in the subsystem object class.
-t *Type*	Specifies a subsystem subserver to turn tracing off. The command is unsuccessful if the *Type* is not contained in the subserver object class.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To turn off the tracing of a group, enter the following command:

   ```
   tracesoff  -g tcpip
   ```

   This turns the tracing off for the `tcpip` group.
2. To turn off tracing of the `sendmail` subsystem on a foreign host, enter the following command:

   ```
   tracesoff -h odin  -s sendmail
   ```

   This turns off the tracing for the `sendmail` subsystem on the `odin` foreign host.

Files

Item	Description
/usr/bin/tracesoff	Contains the **tracesoff** command.
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration Object Class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration Object Class.
/etc/services	Defines the sockets and protocols used for Internet services.
/dev/SRC	Specifies the **AF_UNIX** socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.

traceson Command

Purpose

Turns on tracing of a subsystem, a group of subsystems, or a subserver.

Syntax

Subsystem

traceson [**-h** *Host*] [**-l**] { **-g** *Group* | **-p** *SubsystemPID*| **-s** *Subsystem*}

Subserver

traceson [**-h** *Host*] [**-l**] **-t** *Type* [**-o** *Object*] [**-p** *SubsystemPID*] [**-P** *SubserverPID*]

Description

The **traceson** command sends the System Resource Controller a subsystem request packet that is forwarded to the subsystem to turn tracing on. Tracing is unsuccessful if the communication method for the subsystems is signals.

Note: Tracing is subsystem dependent.

Tracing may occur in either short or long form. When the **-l** flag is absent, the trace request is assumed to be a short trace.

Flags

Item	Description
-g *Group*	Specifies a group of subsystems to turn tracing on. The command is unsuccessful if the *Group* name is not contained in the subsystem object class.
-h *Host*	Specifies the foreign host on which this trace action is requested. The local user must be running as "root". The remote system must be configured to accept remote System Resource Controller requests. That is, the **srcmstr** daemon (see **/etc/inittab**) must be started with the **-r** flag and the **/etc/hosts.equiv** or **.rhosts** file must be configured to allow remote requests.
-l	Specifies that a long trace is requested.
-o *Object*	Specifies that a subserver object is to be passed to the subsystem as a character string.
-p *SubsystemPID*	Specifies a particular instance of the subsystem to turn tracing on, or a particular instance of the subsystem to which the trace subserver request is to be passed.
-P *SubserverPID*	Specifies that a subserver PID is to be passed to the subsystem as a character string.
-s *Subsystem*	Specifies the subsystem to turn tracing on. The *Subsystem* name can be either the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the *Subsystem* name is not contained in the subsystem object class.
-t *Type*	Specifies a subserver to turn tracing on. The command is unsuccessful if the *Type* is not contained in the subserver object class.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To turn on tracing of the `tcpip` group on a foreign host, enter the following command:

```
traceson -h odin  -g tcpip
```

This turns on the tracing for the `tcpip` group on the `odin` foreign host.

2. To turn on tracing of the `sendmail` subsystem on a foreign host, enter the following command:

```
traceson -h odin  -s sendmail
```

This turns on the tracing for the `sendmail` subsystem on the `odin` foreign host.

Files

Item	Description
/usr/bin/traceson	Contains the **traceson** command.
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration Object Class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration Object Class.
/etc/services	Defines the sockets and protocols used for Internet services.
/dev/SRC	Specifies the **AF_UNIX** socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.

trbsd Command

Purpose

Translates characters (BSD version).

Syntax

trbsd [**-c**] [**-d**] [**-s**] [**-A**] [*String1* [*String2*]]

Description

The **trbsd** command deletes or substitutes characters from standard input and then writes the result to standard output. The **trbsd** command is the BSD version of the **tr** command. The **trbsd** command performs three kinds of operations, depending on the character strings specified by the parameters and flags specified. The default value for either the *String1* or *String2* parameter is a null string.

Transforming Characters

If both the *String1* and *String2* parameters are specified and the **-d** flag is not specified, the **trbsd** command replaces each character from standard input that is specified by the *String1* parameter with the character in the same position in the *String2* parameter.

If the *String1* parameter specifies a character more than once, the character is translated into the character in the *String2* parameter that corresponds to the last occurrence of the character in the *String1* parameter.

Deleting Characters Using the -d Flag

If the **-d** flag is specified, the **trbsd** command deletes each character from standard input that is specified by the *String1* parameter.

Removing Sequences of Characters Using the -s Flag

If the **-s** flag is specified, the **trbsd** command deletes from standard input all but the first character in a sequence of two or more repetitions of any character specified by the *String2* parameter.

Both the *String1* and *String2* parameters must be specified when both the **-d** and **-s** flags are specified.

> **Note:** The **trbsd** command deletes all null characters from standard input before it begins processing.

Special Sequences for Expressing Strings

The strings contained in *String1* and *String2* parameters can be expressed using the following conventions:

Item	Description
C1-*C2*	Specifies the string of characters that collate between the character specified by the *C1* string and the character specified by the *C2* string, inclusive. The character specified by the *C1* string must collate before the character specified by the *C2* string.
*****Octal*	Specifies the character whose encoding is represented by the specified octal value. The octal value can be a one-, two-, or three-digit octal integer. Multibyte characters can be expressed by writing backslash-octal sequences for each byte.
\-	The \- (backslash, minus sign) specifies the minus sign character itself, without any special meaning as an escape character.

If the strings specified by the *String1* and *String2* parameters are not the same length, the **trbsd** command pads the shorter string to equal the length of the longer string. Padding is accomplished by duplicating the last character in the shorter string as many times as necessary.

Flags

Item	Description
-A	Performs all operations on a byte-by-byte basis using the ASCII collation order for ranges and character classes, instead of the collation order of the current locale.
-c	Specifies that the value of the *String1* parameter be replaced by the complement of that string. The complement is all of the characters in the character set of the current locale, except for the characters specified by the *String1* parameter. If the **-A** and **-c** flags are specified together, characters are complemented with respect to the set of all 8-bit character codes.
-d	Deletes each character from standard input that is contained in the *String1* parameter.
-s	Deletes from standard input all but the first character in a sequence of two or more repetitions of any character contained in the *String2* parameter.

Examples

1. To translate braces into parentheses, enter:

   ```
   trbsd '{}' '()' < textfile > newfile
   ```

   This translates each { (left brace) to ((left parenthesis) and each } (right brace) to) (right parenthesis). All other characters remain unchanged.
2. To interchange plus signs with minus signs, and slash characters with asterisks, enter:

   ```
   trbsd '+\-/*' '\-+*/' < textfile > newfile
   ```

   The minus sign must be entered with a backslash escape character.
3. To translate lowercase characters to uppercase, enter:

   ```
   trbsd 'a-z' 'A-Z' < textfile > newfile
   ```
4. To create a list of words in a file, enter:

   ```
   trbsd -cs 'a-zA-Z' '\012' < textfile > newfile
   ```

This translates each sequence of characters other than lowercase letters and uppercase letters into a single newline character. The octal value 012 is the code for the newline character.

5. To replace every sequence of one or more newlines with a single newline, enter:

```
trbsd -s '\012'  < textfile > newfile
```

Files

Item	Description
/usr/bin/trbsd	Contains the **trbsd** command.
/usr/ucb/tr	Contains a symbolic link to the **trbsd** command.

trcctl Command

Purpose

Changes and displays system trace parameters.

Syntax

`trcctl` [`-d` *Directory* `-l` `-L` *LogfileSize* `-M` *LMT_log_dir* `-N` *NonrootUserBufferMax* `-o` *Logfile* `-r` `-T` *BufferSize* `-R` *disable*|*enable* `-S` *disable*|*enable*]

Description

The `trcctl` command will display or change the system trace default parameters. If the `-l` option (or no parameter) is specified, `trcctl` will show the values as follows:

```
Default Buffer Size: 131072
Default Log File Size: 1310720
Default Log File: /var/adm/ras/trcfile
Non-root User Buffer Size Maximum: 1048576
Default Components Directory File: /var/adm/ras/trc_ct
Default LMT Log Dir: /var/adm/ras/mtrcdir
Restrict non-privileged users from using trace Channel-0:  disable
Restrict non-privileged users from using trcrpt Channel-0: disable
```

Note that the default buffer and log file sizes initially depend upon the kernel. However, once they are set using this command, the effected value is the same for both kernels. The other parameters allow these default values to be changed. To change a default value, the user must be a member of the system group. Many of the flags used with `trcctl` correspond to those used by the `trace` daemon.

Flags

Item	Description
`-d` *Directory*	Specifies the default Component Trace log directory path. The default value is `/var/adm/ras/trc_ct`.
`-l`	Lists the current values.
`-L` *Value*	Specifies the default log file size. The original default value is 1310720 bytes for the 32-bit kernel, and 2621440 bytes for the 64-bit kernel. If specified with `-L`, the default will apply to both kernels.
`-M` *LMT_log_dir*	Specifies the default Lightweight Memory Trace log directory path. The default value is `/var/adm/ras/mtrcdir`.
`-N` *Value*	Specifies the maximum buffer size a non-root user may specify. The default is 1 MB, 1048576 bytes.

Item	Description
-o *Path*	Specifies the default log file path. The default value is `/var/adm/ras/trcfile`.
-R *disable* | *enable*	Restricts the **trace** facility of channel 0 to only the privileged users. If the `-R` option is disabled, the **trace** facility of channel 0 is available to all users. The default value is *disable*. For more information, see the `Security` section of the **trcrpt** command.
-S *disable* | *enable*	Restricts the **trcrpt** facility of channel 0 to only the privileged users. If the `-S` option is disabled, the **trcrpt** facility of channel 0 is available to all users. The default value is *disable*. For more information, see the `Security` section of the **trace** command
-r	Restores original default value.
-T *Value*	Specifies the default trace buffer size. The original default values are 128 KB and 256 KB for a 32- or 64-bit kernel. If specified with `-T`, the default will apply to both kernels.

Parameters

If you use 'k', 'm', or '#k', '#m' as parameters for the `-N`, `-L`, and `-T` options, `trcctl` will translate these into their respective byte totals.

k = 1024
m = 1048576

Using only 'k' or 'm', `trcctl` assumes you mean 1 kilobyte or 1 megabyte respectively. This way a root user can execute :

```
trcctl -L 10m -N m -T 256k
```

Security

The user must be a member of the system group.

trcdead Command

Purpose

Extracts trace buffers from a system dump image or live dump image.

Syntax

trcdead [`-1 -2 -3 ... -7`] [`-c`] [`-M`] [**-o** *Name*] *DumpImage* [*UnixFile*]

Description

If the system halts while trace facilities are active, the contents of the internal trace buffers are captured in the system dump. Alternatively, a live dump can also capture partial or complete internal trace buffers if the appropriate pseudo-component. Use the **trcdead** command to extract the eight active system trace channels, all component trace buffers, and the lightweight memory trace buffers from the system dump or the live dump. The system trace channel *0* is extracted when you do not specify any flag. To trace a channel other than channel *0* is identified through a *-channelnum* flag. Use a **-c** flag to identify component trace buffers. Use the **-M** flag to identify lightweight memory trace buffers. You can extract only one type of trace buffer, or one specific system trace channel at one time.

The **-o** flag can be used to indicate that the extracted buffers should be written to a nondefault trace log file or directory. System trace channels are extracted to a trace log file. Component Trace buffers and

Lightweight Memory Trace buffers are extracted to a directory. If the **-o** flag is not chosen, the **trcdead** command writes to the default trace log file or directory. The default log file name and directory names can be viewed and modified using the `trcctl` command.

Use the **trcrpt** command to format a report from the trace log file or files.

Flags

Item	Description
–1, ..., -7	Retrieves the trace buffer entries for channel 1, 2, 3, 4, 5, 6, and 7. The default is channel 0.
-c	Extracts all buffers of all active Component Trace components.
-M	Extracts the Lightweight Memory Trace buffers.
-o*Name*	Specifies the file or directory (`-c`, `-M`) to which data is written.

Parameter

Item	Description
DumpImage	Specifies the dump image to operate on.
UNIX File	Specifies the UNIX file that is in use when the system dump or live dump is taken. This is not necessary if you are using the **trcdead** command on the same system that the dump originated from.

Examples

> **Note:** To determine which example is more appropriate for your system, use the **sysdumpdev**command to display the current dump device assignments.

1. To extract the system trace buffer to the file named `trace_extract` from a dump located at `/var/adm/ras/dumpfile`, enter:

```
trcdead -o trace_extract /var/adm/ras/dumpfile
```

2. To extract the system trace buffer from a dump image written to a device, enter:

```
trcdead /dev/hd7
```

3. To extract lightweight memory trace information from dump image `vmcore.0` and put it into the `/tmp` directory, enter:

```
trcdead -o /tmp -M vmcore.0
```

4. To extract the component trace buffers from the dump image `vmcore.3` that is produced by the **/tmp/unix_64**, enter:

```
trcdead -c vmcore.3 /tmp/unix_64
```

Files

Item	Description
/**usr**/**bin**/**trcdead**	Contains the **trcdead** command.
/var/adm/ras/dumpfile	Contains the default system dump file.
/var/adm/ras/trcfile	Contains the default system trace log.

Item	Description
/var/adm/ras/trc_ct	Contains the default component trace logs.
/var/adm/ras/mtrcdir	Contains the default lightweight memory trace logs.

trcevgrp Command

Purpose

Manipulates trace event groups.

Syntax

List event groups

trcevgrp -l [*event-group* [...]]

Remove event groups

trcevgrp -r [*event-group* [...]]

Add an event group

trcevgrp -a -d "*group-description*" **-h** "*hook-list*" *event-group*

Update an event group

trcevgrp -u [**-d** "*group-description*"] [**-h** "*hook-list*"] *event-group*]

Description

The **trcevgrp** command is used to maintain the trace event groups. You must be in the system group to add, delete, or change trace event groups. You *cannot* modify or delete event groups whose type is `reserved`.

In AIX version older than AIX 6.1, you can specify only three-digit hook IDs. In AIX 6.1 or later, you can specify four-digit hook IDs.

Flags

Item	Description
-a [**-d** *group-description* | **-h** *hook-list*]	Creates a new event group. Only one event group name can be specified. Both **-d** *description* and **-h** *hook-list* must be specified when using the **-a** flag. If either **-d** or **-h** is not specified, an error is produced.
-d *group-description*	Designates the hook description. A description is required for all new groups.
-h *hook-list*	The hook list consists of trace hook IDs. The **-h** flag is required when using the **-a** flag. When updating an event group (**-u** flag), the hook-list, if specified, must contain all hook IDs for the group. List parameter items can be separated by commas or enclosed in double quotation marks and separated by commas or blanks.

Item	Description
-l *event-group*	The specified groups are listed to standard output. If no event group is specified, all are listed. The format of the listing is as follows: *group name - group-description (type)* "hook list" The following examples shows the listing of the group: • `* -l tidhk - Hooks needed to display thread name (reserved) "106,10C,134,139,465"` • `* -l gka - GENERAL KERNEL ACTIVITY (files,execs,dispatches) (reserved) "106,10C,134,139,465,107,135,15b,12e,116,117,200,20E,20F"` • `* -l mydriver - My Driver (files,execs,dispatches) (reserved) "106,1AB1,0AC0"`
-r *event-group*	Removes the specified event-groups.
-u [**-d** "*group-description*" | **-h** "*hook-list*"] *event-group*	Used to update the information for an event-group. Either **-d** *description* or **-h** *hook-list* must be specified.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To get a listing of all event groups, enter the following command:

   ```
   trcevgrp -l
   ```

2. To add a new group, enter the following command:

   ```
   trcevgrp -a -d "my group description" -h "500,501,502" mygrp
   ```

   This will add the group called `mygrp`, give it the description `my group description`, and will have hooks of `500`, `501`, and `502`.

3. To add another hook to `mygrp`, enter the following command:

   ```
   trcevgrp -u -d "my group description" -h "500,501,502,503" mygrp
   ```

   Note: You must specify all of the hook IDs.

   Files the event groups are currently kept in the SWserveAt ODM database.

trcnm Command

Purpose

Generates a kernel name list.

Syntax

trcnm [**-a** [*FileName*]] | [*FileName*] | **-K***Symbol1* ...

Description

The **trcnm** command generates a kernel name list used by the **trcrpt** command. A kernel name list is composed of a symbol table and a loader symbol table of an object file. The **trcrpt** command uses the kernel name list file to interpret addresses when formatting a report from a trace log file. For more information, see the **trcrpt -n** command.

If the *FileName* parameter is not specified, the default *FileName* is /unix.

Flags

Item	Description
-a	Writes all loader symbols to standard output. The default is to write loader symbols only for system calls.
-K*Symbol...*	Obtains the value of all command line symbols through the **knlist** command system call.

Examples

1. To obtain the value of the symbols in /unix, enter:

   ```
   trcnm -K environ errno
   ```

   This command sequence displays the following:

   ```
   environ 2FF7FFF8
   errno 2FF7FFFC
   ```

2. To print a symbol table for system calls, enter:

   ```
   trcnm
   ```

   A list similar to the following is generated:

   ```
   pin_obj_start    00000000
   header_offset    00000008
   ram_disk_start   0000000C
   ram_disk_end     00000010
   dbg_avail        00000014
   base_conf_start  00000018
   base_conf_end    0000001C
   base_conf_disk   00000020
   pin_com_start    00000024
   start            00000028
   ipl_cb           00000028
   ...
   ```

Files

Item	Description
/var/adm/ras/trcfile	Contains the default log file.
/tlo-tvl2/trcnam	Contains the **trcnm** command.
/etc/trcfmt	Contains the trace format file.

trcrpt Command

Purpose

Formats a report from the trace log.

Syntax

trcrpt [**-c**] [**-C** [*processorList* | **all**]] [**-d** *List*] [**-D** *Event-group-list*] [**-e** *Date*] [**-G**] [**-h**] [**-j**] [**-k** *List*] [**-K** *Group-list*] [**-m**] [**-n** *Name*] [**-o** *File*] [**-p** *List*] [**-r**] [**-s** *Date*] [**-t** *File*] [**-T** *List*] [**-v**] [**-O** *Options*] [**-x**] [**-@** *WparList*] [**-M** **common** | **rare** | **all**[:*LMT_dir*]] [`-l` *ComponentList* | `all`[:*CT_dir*]] [*FileOrDirectory*]

Description

The **trcrpt** command reads the trace log specified by the **-M**, **-l** and *File* or *Directory* parameters, formats the trace entries, and writes a report to standard output. The default file from which the system generates a trace report is the **/var/adm/ras/trcfile** file, but you can specify an alternate log file using the **-M**, **-l** and *File* or *Directory* parameters. You can specify one or more files or directories. If you specify a file, it must be a valid trace log file, which is any file that is produced by a trace-related command. If you specify a directory, it must contain a component trace master file. If you specify the **-m** flag, all specified traces will be merged in chronological order.

To include trace entries in a report for the specified workload partition (WPAR), use the **-@** flag.

In AIX 6.1 and later, four-hex-digit hook IDs can be displayed. However, if a four-hex-digit hook ID has a digit of zero, the zero is removed to display only three hex digits. This occurs because four-hex-digit hook IDs in the form **hhh0** are equivalent to three-hex-digit hook IDs in the form **hhh**.

You can use the System Management Interface Tool (SMIT) to run the **trcrpt** command by entering the SMIT fast path:

```
smit trcrpt
```

Flags

Item	Description
-@ *WparList*	Generates a report containing events that occurred on the workload partitions that you specified. You can specify a list of WPAR configured IDs (CID) or a list of WPAR names with the *WparList* parameter. The list items can either be separated by commas or enclosed in quotation marks and separated by commas or spaces. Specify 0 or Global in the list to include the Global system in the report.
-c	Checks the template file for syntax errors.
-C [*processorList* | **all**]	Generates a report containing events that occur on the processors specified. The processors can be separated by commas, or enclosed in double quotation marks and separated by commas or blanks. To report on all processors, specify `trace -C all`. The **-C** flag is not necessary unless you want to see only a subset of the processors traced, or have the processor number show up in the report. If **-C** is not specified, and the trace is a multi-processor trace, **trcrpt** generates the trace report for all processors, but the processor number is not shown for each hook unless you specify `-0 cpuid=on`.

Item	Description
-d *List*	Limits the report to hook IDs specified with the *List* variable. The *List* parameter items can be separated by commas or enclosed in double quotation marks and separated by commas or blanks. In AIX 6.1 and later, four-hex-digit hook IDs can be displayed. However, if a four-hex-digit hook ID has a digit of zero, the zero is removed to display only three hex digits. This occurs because four-hex-digit hook IDs in the form **hhh0** are equivalent to three-hex-digit hook IDs in the form **hhh**.
-D *Event-group-list*	Limits the report to hook IDs in the *Event groups list*, plus any hook IDs specified with the **-d** flag. The list parameter items can be separated by commas or enclosed in double quotation marks and separated by commas or blanks. *Event groups* are described in **Debug and Performance Tracing**. The **-D** flag also reports the trace utility hook id for LMT Restart and LMT Suspend.
-e *Date*	Ends the report time with entries on, or before, the specified date. The *Date* variable has the form *mmddhhmmssyy* (month, day, hour, minute, second, and year). Date and time are recorded in the trace data only when trace data collection is started and stopped. If you stop and restart trace data collection multiple times during a trace session, date and time are recorded each time you start or stop a trace data collection. Use this flag in combination with the **-s** flag to limit the trace to data collected during a certain time interval. **Restriction:** The **-e** and **-s** flags are only valid for trace log files collected without the `trace -C` flag.
-G	Lists all event groups. The list of groups, the hook ids in each group, and each group's description is listed to standard output.
-h	Omits the header information from the trace report and writes only formatted trace entries to standard output.
-j	Displays the list of hook IDs. The **trcrpt -j** command can be used with the **trace -j** command that includes IDs of trace events or the **trace -k** command that excludes IDs of trace events.

Item	Description
-k *List*	Excludes from the report hook IDs specified with the *List* variable. The *List* parameter items can be separated by commas or enclosed in double quotation marks and separated by commas or blanks. In AIX 6.1 and above, specifying a two-digit hook ID in the hh form results in hh00, hh10,...,hhF0. Specifying a three-digit hook ID in the hhh form results in hhh0. Specifying a four-digit hook ID in the hhhh form results in hhhh.
-K *Event-group-list*	Excludes from the report hook IDs in the *event-groups* list, plus any hook IDs specified with the **-k** flag. List parameter items can be separated by commas or enclosed in double quotation marks and separated by commas or blanks. Event groups are described in **Debug and Performance Tracing**.
-l *ComponentList*	Generates a report for a multi-component trace with `ctctrl -D` or **trcdead**. The components can be separated by commas, or enclosed in double quotation marks and separated by commas or blanks. The `-l` flag is not necessary unless you want to see only a subset of the components traced. If `-l` is not specified, the command assumes the trace is a multi-component trace if a directory is given as input on the command line. Multi-component trace log files not in the default directory must either have their directory specified on the command line or with the *CT_dir* parameter in conjunction with the **-l** flag. The **-l all** option can be used to select all available components. Multiple **-l** flags can be used to specify components in different directories.
-m	Merges all specified trace files based on time stamps. Files merged from another partition, system or from two or more separate boots of the same system will produce unpredictable results. Without the **-m** flag, reports for each log file are appended to the specified output file.

Item	Description
-M common | **rare** | **all**[:*LMT_dir*]	Generates a report from the LMT log files obtained via the **mtrcsave** or **trcdead** command. Use the **common** keyword if you only want events from the common LMT buffers to be reported; use the **rare** keyword if you only want events from the rare LMT buffers to be reported; use the **all** keyword if you want common and rare events to be reported. This flag searches only the default LMT log directory unless the *LMT_dir* parameter is specified. With this parameter, the **trcrpt** command will search for the LMT files in the specified directory rather than the default LMT log directory. To merge common and rare buffers you must use the **all** keyword and the **-m** flag. The **-M** flag can only appear once.
-n *Name*	Specifies the kernel name list file to be used to interpret address for output. Usually, this flag is used when moving a trace log file to another system.
-o *File*	Writes the report to a file instead of to standard output.
-O *Options*	Specifies options that change the content and presentation of the **trcrpt** command. Arguments to the options must be separated by commas or enclosed in double quotation marks and separated by commas or spaces. Valid options are: **2line=[on|off]** Uses two lines per trace event in the report instead of one. The default value is **off**. **component=[on|off]** Displays the full component name in the trace report. The default value is **off**. **cpuid=[on|off]** Displays the physical processor number in the trace report. The default value is **off**. **cid=[on|off]** Displays the workload partition configured ID (CID) in the trace report. The default value is **off**. **endtime=Seconds** Displays the trace report data for events recorded before the seconds specified. Seconds can be given in either an integral or rational representation. If this option is used with the **starttime** option, a specific range can be displayed. **exec=[on|off]** Displays the exec path names in the trace report. The default value is **off**.

Item	Description
	filename=[on|off] Displays the file name from which an event was retrieved. The file name will be truncated from the left if it exceeds 40 characters. The default value is **off**.
	hist=[on|off] Logs the number of instances that each hook ID is encountered. This data can be used for generating histograms. The default value is **off**. This option cannot be run with any other option.
	ids=[on|off] Displays the trace hook identification numbers in the first column of the trace report. The default value is **on**.
	pagesize=*Number* Controls the number of lines per page in the trace report and is an integer within the range of 0 through 500. The column headings are included on each page. No page breaks are present when the default value of 0 is set.
	pid=[on|off] Displays the process IDs in the trace report. The default value is **off**.
	reportedprocessors=[on | off] Displays the number of processors remaining. This option is only meaningful for a multi-processor trace, `trace -C`. For example, if you are reading a report from a system with 4 processors, and the reported processor's value goes from 4 to 3, then you know that there are no more hooks to be reported for that processor.
	PURR=[on | off] Tells `trcrpt` to show the PURR along with any timestamps. The PURR is displayed following any timestamps. If the PURR is not valid for the processor traced, the elapsed time is shown instead of the PURR. If the PURR is valid, or the `cpuid` is unknown, but wasn't traced for a hook, the PURR field contains asterisks (*).

Item	Description
removedups=[on | off]	Enables duplicate event detection. A count in the DUPS column displays the number of events that each event in the report represents. If this option is set to **off**, duplicate event detection will be disabled. The default value is **on**. This option is only valid when merging log files via the **-m** flag. Duplicate entries can only be detected when the processor ID is known from the trace entry itself, not when it must be inferred. The processor ID can be obtained from the entry in the following cases: • A lightweight memory trace • A multi-processor system trace, where the **trace -C** command option was used • A 64-bit system trace initiated with the **-p** option • A 64-bit component trace.
wparname= [on | off]	Displays the workload partition names in the trace report. The default value is **off**.

Item	Description
	starttime=*Seconds* Displays trace report data for events recorded after the seconds specified. The specified seconds are from the beginning of the trace file. Seconds can be given in either an integral or rational representation. If this option is used with the **endtime** option, a specific range of seconds can be displayed.
	svc=[on|on_noblank|off] Displays the value of the system call in the trace report. The default value is **off**. This option can have following values: **on** Prints the name of the current system call in the trace report. **on_noblank** Prints the `----` string in the trace report when the `svc` option is not set. **off** Does not print any information that is related to the system call.
	tid=[on|off] Displays the thread ID in the trace report. The default value is **off**.
	timestamp=[0|1|2|3|4] Controls the reporting of the time stamp associated with an event in the trace report. The possible values are: **0** Time elapsed since the trace was started and delta time from the previous event. The elapsed time is in seconds and the delta time is in milliseconds. Both values are reported to the nearest nanosecond. This is the default. **1** Short elapsed time. Reports only the elapsed time (in seconds) from the start of the trace. Elapsed time is reported to the nearest microsecond. **2** Microsecond delta time. This is like 0, except the delta time is in microseconds, reported to the nearest microsecond. **3** No time stamp. **4** Raw timestamp from the trace event.

Item	Description
-p *List*	Reports the process IDs for each event specified by the *List* variable. The *List* variable may be a list of process IDs or a list of process names. List items that start with a numeric character are assumed to be process IDs. The list items can be separated by commas or enclosed in double quotation marks and separated by commas or blanks.
-r	Outputs unformatted (raw) trace entries and writes the contents of the trace log to standard output one entry at a time. Use the **-h** flag with the **-r** flag to exclude the heading. To get a raw report for processors in a multi-processors trace, use both the **-r** and **-C** flags.
-s *Date*	Starts the report time with entries on, or before, the specified date. The *Date* variable has the form *mmddhhmmssyy* (month, day, hour, minute, second, and year). Date and time are recorded in the trace data only when trace data collection is started and stopped. If you stop and restart trace data collection multiple times during a trace session, date and time are recorded each time you start or stop a trace data collection. Use this flag in combination with the **-e** flag to limit the trace to data collected during a certain time interval. **Restriction:** The **-e** and **-s** flags are only valid for trace log files collected without the `trace -C` flag.
-t *File*	Uses the file specified in the *File* variable as the template file. The default is the **/etc/trcfmt** file.
-T *List*	Limits the report to the kernel thread IDs specified by the *List* parameter. The list items are kernel thread IDs separated by commas or enclosed in double quotation marks and separated by commas or spaces. Starting the list with a kernel thread ID limits the report to all kernel thread IDs in the list. Starting the list with a **!** (exclamation point) followed by a kernel thread ID limits the report to all kernel thread IDs not in the list.
-v	Prints file names as the files are opened. Changes to verbose setting.
-x	Displays the exec path name and value of the system call.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

To run the **trcrpt** command on channel 0, you must have the following additional authorizations, if channel 0 **trcrpt** restriction is enabled in the **trcctl** command:

Item	Description
aix.ras.trace.trcrptch0	Required to run the **trcrpt** command on channel 0.

Note: By default, the root and system group users are privileged users.

To perform all functionality of all commands including the **trcrpt** command on channel 0, you must have the **aix.ras.trace** authorization.

Examples

1. To format the trace log file and print the result, enter:

   ```
   trcrpt | qprt
   ```

2. To send a trace report to the **/tmp/newfile** file, enter:

   ```
   trcrpt -o /tmp/newfile
   ```

3. To display process IDs and exec path names in the trace report, enter:

   ```
   trcrpt -O pid=on,exec=on
   ```

4. To create trace ID histogram data, enter:

   ```
   trcrpt -O hist=on
   ```

5. To produce a list of all event groups, enter:

   ```
   trcrpt -G
   ```

   The format of this report is shown under the **trcevgrp** command.

6. To generate back-to-back LMT reports from the common and rare buffers, enter:

   ```
   trcrpt -M all
   ```

7. If, in the above example, the LMT files reside at **/tmp/mydir**, and we want the LMT traces to be merged, enter:

   ```
   trcrpt -m -M all:/tmp/mydir
   ```

8. To merge the system trace with the **scdisk.hdisk0** component trace, enter:

   ```
   trcrpt -m -l scdisk.hdisk0 /var/adm/ras/trcfile
   ```

9. To merge LMT with the system trace while not eliminating duplicate events, enter:

   ```
   trcrpt -O removedups=off -m -M all /var/adm/ras/trcfile
   ```

10. To merge all component traces in **/tmp/mydir** with the LMT traces in the default LMT directory while showing the source file for each trace event, enter:

    ```
    trcrpt -O filename=on -m -M all /tmp/mydir
    ```

    Tip: This is equivalent to the following command:

    ```
    trcrpt -O filename=on -m -M all -l all:/tmp/mydir
    ```

    Tip: If the traces are from a 64-bit kernel, duplicate entries will be removed. However, on the 32-bit kernel, duplicate entries will not be removed since we do not know the processor IDs of the entries in the components traces.

Files

Item	Description
/usr/bin/trcrpt	Contains the **trcrpt** command.
/var/adm/ras/trcfile	Contains the default log file.
/var/adm/ras/mtrcdir	Location of the default LMT dump directory
/var/adm/ras/trc_ct	Location of the default CT dump directory.
/etc/trcfmt	Contains the trace format file.

trcstop Command

Purpose

Stops the trace function.

Syntax

trcstop [**-<channel>**][**-s** | **-d**]

Description

The **trcstop** command ends a trace session.

You can use the System Management Interface Tool (SMIT) to run the **trcstop** command. To use SMIT, enter:

```
smit trcstop
```

Flags

Item	Description
-<channel>	Specifies the channel which stop the trace. The valid value range is from 0-7. If unspecified, then the default value is 0.
-s	Enables the serialization of trace I/O from multiple processor buffers into the trace file during the *tracestop* operation. The **-s** flag is mutually exclusive with the **-d** flag. **Note:** The serial **-s** option is available for all modes (single, circular, and alternate). In previous releases, the **-s** option was available only for circular mode.
-d	Discards any captured trace buffers which are yet to be written into the file.

Example

To terminate the trace background process, enter:

```
trcstop
```

File

Item	Description
/usr/bin/trcstop	Contains the **trcstop** command.

trcupdate Command

Purpose

Adds, replaces, or deletes trace report format templates.

Syntax

trcupdate [**-o**] [**-t** *File*] [**-v**] [**-x** *IDList*] [*File*]

Description

The **trcupdate** command adds, replaces, or deletes trace report format templates in the **/etc/trcfmt** or the **/etc/trcfmt.Z** file. When the **/etc/trcfmt.Z** file is used, the **trcupdate** command uncompresses the file, updates it, and recompresses it. The **trcupdate** command creates an "undo" file named *File***.undo.trc** in the specified directory.

The **trcupdate** command adds the extension **.trc** to the file name and reads update commands from that file. The undo file is input to the **trcupdate** command if the **-o** (override) flag is specified. When the **-o** flag is specified, the **trcupdate** command undoes the changes previously made to the file.

The first field of each template contains an operator:

Item	Description
+	The plus sign indicates that a template is to be added or replaced. The field that follows this operator contains the template to be replaced.
-	The minus sign indicates that a template is to be deleted. The field after this operator contains the hook ID of the template to delete. Operations are performed in the order in which they appear.

The input to the **trcupdate** command must contain the following as the first line:

```
* /etc/trcfmt
```

The following is a sample trace file:

```
* /etc/trcfmt
+ 15A 1.0 new_fmt
- 1B3
- A14
```

When adding or replacing, the **trcupdate** command compares the version numbers of each input template with the version number of the template with the same hook ID. If the version number of the input template is greater than or equal to the version of the existing template, the **trcupdate** command replaces the old template with the input template. If a template does not exist, then the input template is added to the file.

The **trcupdate** command will not modify the **/etc/trcfmt** file if a syntax error is detected in the update file.

Flags

Item	Description
-o	Overrides the old template with the input template without verifying the version number of either template.
-t *File*	Specifies a file, instead of the **/etc/trcfmt** or the **/etc/trcfmt.Z** file, to be used as the template file.

Item	Description
-v	Prints the file names as each file is opened.
-x *IDList*	Extracts the templates specified in the *IDList* from the template file and writes them to standard output. The *IDList* parameter lists the hook IDs.

Security

Access Control: None, but you must have write authority to the template file you are changing. The default is **/etc/trcfmt**.

Examples

1. To add a template, enter the following command:

   ```
   trcupdate
   * /etc/trcfmt
   + 15A 1.0 new_fmt
   ```

   Tip: In AIX 6.1 and later versions, this is equivalent to:

   ```
   trcupdate
   * /etc/trcfmt
   + 15A0 1.0 new_fmt
   ```

2. To delete a template, type the following command:

   ```
   trcupdate
   * /etc/trcfmt
   - 15A 1.0 new_fmt
   ```

   Tip: In AIX 6.1 and later versions, this is equivalent to:

   ```
   trcupdate
   * /etc/trcfmt
   - 15A0 1.0 new_fmt
   ```

3. To replace a template, enter the following command:

   ```
   trcupdate
   * /etc/trcfmt
   + 15A 1.0 new_fmt
   ```

   Tip: In AIX 6.1 and later versions, this is equivalent to:

   ```
   trcupdate
   * /etc/trcfmt
   + 15A0 1.0 new_fmt
   ```

4. In AIX 6.1 and later versions, to add a template for hook ID 0AB0, enter the following command:

   ```
   trcupdate
   * /etc/trcfmt
   + 0AB0 1.0 new_fmt
   ```

   The above command is equivalent to the following command:

   ```
   trcupdate
   * /etc/trcfmt
   +0AB 1.0 new_fmt
   ```

5. In AIX 6.1 and above, to add a template for hook ID 1AB1, enter the following command:

```
trcupdate
* /etc/trcfmt
+ 1AB1 1.0 new_fmt
```

Files

Item	Description
/usr/bin/trcupdate	Contains the **trcupdate** command.
/etc/trcfmt	Contains the trace format file.
/usr/include/sys/trcmacros.h	Defines **trchook** and **utrchook** macros.

troff Command

Purpose

Formats text for printing on typesetting devices.

Syntax

troff [**-a**] [**-i**] [**-q**] [**-z**] [**-F** *Directory*] [**-n** *Number*] [**-o** *List*] [**-r** *ANumber*]
[**-s** *Number*] [**-T** *Name*] [**-mm** | **-me** | **-mptx** | **-ms** | **-man** | **-mv**] [**-M** *Media*] [*File* ... | **-**]

Description

The **troff** command reads one or more files and formats the text for printing on a phototypesetter or comparable device. A postprocessor is then required to post process the output of the **troff** command to the target device. See the accompanying example.

If no file is specified oroh.. the **-** (minus) flag is not the last parameter, standard input is read by default.

For the 3812, 3816, and Hewlett-Packard LaserJet Series II printer, the default fonts are the native fonts for the printer. Additional fonts also are available for these printers, which can be loaded through the use of the **troff .fp** directive. These fonts are stored on the host in the directory **/usr/lib/font/dev**Printer/**bitmaps**, and downloaded to the printer as necessary.

Typefaces

Three different typefaces are provided in four styles. The following chart shows the relationship between typeface, style, and the name that the **troff** command uses to access the font.

Note: The fonts in this set are based on the Computer Modern letter forms developed by Donald E Knuth. (Refer to Knuth, Donald: *Computer Modern Typefaces*. Addison-Wesley, 1986.)

Typeface	Regular	Italic	Bold	Italic
Roman	cr	cR	Cr	CR
Sans Serif	cs	cS	Cs	CS
Typewriter	ct	cT	Ct	CT
troff special	sp			

These fonts are all provided in the standard 15 troff sizes: 6, 7, 8, 9, 10, 11, 12, 14, 16, 28, 20, 22, 24, 28, and 36 points.

For example, `.fp 1 Cr` loads the Roman bold font into position 1.

Note: The **.tl** request cannot be used before the first break-producing request in the input to the **troff** command.

Flags

Item	Description
-a	Sends a printable ASCII approximation of the results to standard output.
-F*Directory*	Accesses font information from the Directory/**dev**Name directory instead of the default **/usr/lib/font/dev**Name directory (where *Name* is specified by the **-T** flag).
-i	Reads standard input after there are no more files.
-M *Media*	Specifies a paper size in order to determine the amount of imageable area on the paper. Valid values for the *Media* variable are: **A4** Specifies a paper size of 8.3 X 11.7 inches (210 X 297 mm). **A5** Specifies a paper size of 5.83 X 8.27 inches (148 X 210 mm). **B5** Specifies a paper size of 6.9 X 9.8 inches (176 X 250 mm). **EXEC** Specifies a paper size of 7.25 X 10.5 inches (184.2 X 266.7 mm). **LEGAL** Specifies a paper size of 8.5 X 14 inches (215.9 X 355.6 mm). **LETTER** Specifies a paper size of 8.5 X 11 inches (215.9 X 279.4 mm). This is the default value. **Note:** The *Media* variable is not case-sensitive.
-n*Number*	Numbers the first printed page with the value specified by the *Number* variable.
-o*List*	Prints only pages specified by the *List* variable, which consists of a comma-separated list of page numbers and ranges: • A range of *Start-Stop* means print pages *Start* through *Stop.* For example: `9-15` prints pages 9 through 15. • An initial *-Stop* means print from the beginning to page *Stop.* • A final *Start-* means print from page*Start* to the end. • A combination of page numbers and ranges prints the specified pages. For example: `-3,6-8,10,12-` prints from the beginning through page 3, pages 6 through 8, page 10, and page 12 to the end. **Note:** When this flag is used in a pipeline (for example, with one or more of the **pic**, **eqn,** or **tbl** commands), you might receive a `broken pipe` message if the last page in the document is not specified in the *List* variable. This broken pipe message is not an indication of any problem and can be ignored.
-q	Calls the simultaneous input and output mode of the **.rd** request.
-r*ANumber*	Sets the register specified by the *A* variable to the specified number. The *A* variable value must have a one-character ASCII name.
-s*Number*	Generates output to make the typesetter stop every specified number of pages.

Item	Description
-T*Name*	Prepares the output for the specified printing device. Phototypesetters or comparable printing devices use the following *Name* variables for operating system international extended characters. The default is **ibm3816**. **Note:** You get a message that reads `bad point size` if your device does not support the point size that you specified. The **troff** command uses the closest valid point size to continue formatting. **canonls** Canon Lasershot LBP-B406S/D/E,A404/E,A304E. **ibm3812** 3812 Pageprinter II. **ibm3816** 3816 Pageprinter. **hplj** Hewlett-Packard LaserJet II. **ibm5585H-T** 5585-H01 Traditional Chinese Language support. **ibm5587G** 5587-G01, 5584-H02, 5585-H01, 5587-H01, and 5589-H01 Kanji Printer multibyte language support. **psc** PostScript printer. **X100** AIXwindows display. **Note:** You also can set the **TYPESETTER** environment variable to one of the preceding values instead of using the **-T***Name* flag of the **troff** command.
-man	Selects the **man** macro processing package.
-me	Selects the **me** macro processing package.
-mm	Selects the **mm** macro processing package.
-mptx	Selects the **mptx** macro processing package.
-ms	Selects the **ms** macro processing package.
-mv	Selects the **mv** macro processing package.

See Macro Packages for Formatting Tools for more information on the macros.

Item	Description
-z	Prints only messages generated by **.tm** (workstation message) requests.
-	Forces input to be read from standard input.

Environment Variables

Item	Description
TYPESETTER	Contains information about a particular printing device.

Examples

The following is an example of the **troff** command:

```
troff -Tibm3812 File | ibm3812 | qprt
```

Macro Packages for Formatting Tools

The following macro packages are part of the Formatting Tools in the Text Formatting System and are described in more detail on the next pages:

Item	Description
man	Enables you to create your own manual pages from online manual pages.
me	Provides macros for formatting papers.
mm	Formats documents with **nroff** and **troff** formatters.
mptx	Formats a permuted index.
ms	Provides a formatting facility for various styles of articles, theses, and books.
mv	Typesets English-language view graphs and slides by using the **troff** command.

man Macro Package for the nroff and troff Commands

The **man** macro package is provided to enable users to create their own manual pages from online manual pages that have been processed with either the **nroff** command or **troff** command. The **man** macro package is used with either the **nroff** command or the **troff** command.

Special macros, strings, and number registers exist, internal to the **man** macro package, in addition to the following lists of format macros, strings, and registers. Except for the names predefined by the **troff** command and the **d**, **m**, and **y** number registers, all such internal names are of the form *SymbolAlpha*, where *Symbol* is one of **)**, **]**, or **}**, and *Alpha*is any alphanumeric character.

The **man** macro package uses only the Roman font. If the input text of an entry contains requests for other fonts (for example, the **.I** format macro, **.RB** request, or **\fI** request) the corresponding fonts must be mounted.

Format Macros

The following macros are used to alter the characteristics of manual pages that are formatted using the **man**macro package.

Type font and size are reset to default values before each paragraph and after processing font- and size-setting macros (for example, the **.I** format macro, **.SM** format macro, and **.B** format macro).

Tab stops are neither used nor set by any of the format macros except the **.DT** format macro and the **.TH** format macro.

.B [*Text*]
Makes text bold.

The *Text* variable represent up to six words; use “ ” (double quotation marks) to include character spaces in a word. If the variable is empty, this treatment is applied to the next input text line that contains text to be printed. For example, use the **.I** format macro to italicize an entire line, or use the **.SM** and **.B** format macros to produce an entire line of small-bold text. By default, hyphenation is turned off for the **nroff** command, but remains on for the **troff** command.

.DT
Restores default tab settings every 5 ens for the **nroff** command and every 7.2 ens for the **troff** command.

.HP [*Indent*]
Begins a paragraph with a hanging indent as specified by the *Indent*variable.

If the *Indent* variable is omitted, the previous *Indent* value is used. This value is set to its default (5 ens for the **nroff** command and 7.2 ens for the **troff** command) by the **.TH** format macro, **.P** format

macro, and **.RS** format macro, and restored by the **.RE** format macro. The default unit for *Indent* is ens.

.I [*Text*]
Makes text italic.

The *Text* variable represent up to six words; use " " (double quotation marks) to include character spaces in a word. If the variable is empty, this treatment is applied to the next input text line that contains text to be printed. For example, use the **.I** format macro to italicize an entire line, or use the **.SM** and **.B** format macros to produce an entire line of small-bold text. By default, hyphenation is turned off for the **nroff** command, but remains on for the **troff** command.

.IP [*Tag*] [*Indent*]
Same as the **.TP** *Indent* macro with the *Tag* variable; if the value of the *Tag* variable is **NULL**, begin indented paragraph. This macro is often used to get an indented paragraph without a tag.

If the *Indent* variable is omitted, the previous *Indent* value is used. This value is set to its default (5 ens for the **nroff** command and 7.2 ens for the **troff** command) by the **.TH** format macro, **.P** format macro, and **.RS** format macro, and restored by the **.RE** format macro. The default unit for *Indent* is ens.

.P
Begins paragraph with normal font, point size, and indent. The **.PP** macro is a synonym for the **mm** macro package **.P** macro.

.PD [*Number*]
Sets inter-paragraph distance the number of vertical spaces specified by the *Number* parameter. The default *Number* variable value is 0.4v for the **troff** command and 1v for the **nroff** command.

.PM [*Indicator*]
Sets proprietary marking as follows:

Indicator	**Marking**
P	PRIVATE
N	NOTICE
No *Indicator* specified	Turns off proprietary marking.

.RE [*Number*]
Ends relative indent (**.RS**) at indent level position specified by the *Number* variable. If the *Number* variable value is omitted, return to the most recent lower indent level.

.RI *Character1Character2...*
Concatenates the Roman *Character1* with the italic *Character2*; alternate these two fonts up to six sets of *Character1Character2*. Similar macros alternate between any two of Roman, italic, and bold: the **.IR**, **.RB**, **.BR**, **.IB**, and **.BI** macros.

.RS [*Indent*]
Increases relative indent (initially zero). Indent all output an extra number of units from the left margin as specified by the *Indent* variable.

If the *Indent* variable is omitted, the previous *Indent* value is used. This value is set to its default (5 ens for the **nroff** command and 7.2 ens for the **troff** command) by the **.TH** format macro, **.P** format macro, and **.RS** format macro, and restored by the **.RE** format macro. The default unit for *Indent* is ens.

.SH [*Text*]
Places subhead text.

The *Text* variable represent up to six words; use " " (double quotation marks) to include character spaces in a word. If the variable is empty, this treatment is applied to the next input text line that contains text to be printed. For example, use the **.I** format macro to italicize an entire line, or use

the **.SM** and **.B** format macros to produce an entire line of small-bold text. By default, hyphenation is turned off for the **nroff** command, but remains on for the **troff** command.

.SM [*Text*]
Makes text one point smaller than default point size.

The *Text* variable represent up to six words; use " " (double quotation marks) to include character spaces in a word. If the variable is empty, this treatment is applied to the next input text line that contains text to be printed. For example, use the **.I** format macro to italicize an entire line, or use the **.SM** and **.B** format macros to produce an entire line of small-bold text. By default, hyphenation is turned off for the **nroff** command, but remains on for the **troff** command.

.SS [*Text*]
Places sub-subhead text.

The *Text* variable represent up to six words; use " " (double quotation marks) to include character spaces in a word. If the variable is empty, this treatment is applied to the next input text line that contains text to be printed. For example, use the **.I** format macro to italicize an entire line, or use the **.SM** and **.B** format macros to produce an entire line of small-bold text. By default, hyphenation is turned off for the **nroff** command, but remains on for the **troff** command.

.TH [*Title*][*Section*][*Commentary*][*Name*]
Sets the title and entry heading. This macro calls the **.DT** format macro.

Variable	Marking
Title	Title
Section	Section number
Commentary	Extra commentary
Name	New manual name.

Note: If the **.TH** format macro values contain character spaces that are not enclosed in " " (double quotation marks), irregular dots are displayed on the output.

.TP [*Indent*]
Begins indented paragraph with hanging tag. The next input line that contains text is the tag. If the tag does not fit, it is printed on a separate line.

If the *Indent* variable is omitted, the previous *Indent* value is used. This value is set to its default (5 ens for the **nroff** command and 7.2 ens for the **troff** command) by the **.TH** format macro, **.P** format macro, and **.RS** format macro, and restored by the **.RE** format macro. The default unit for *Indent* is ens.

Strings

Item	Description
***R**	Adds trademark, (Reg.) for the **nroff** command and the registered trademark symbol for the **troff** command.
***S**	Changes to default type size.
***(Tm**	Adds trademark indicator.

Registers

Item	Description
IN	Indent left margin relative to subheads. The default is 7.2 ens for the **troff** command and 5 ens for the **nroff** command.
LL	Line length including the value specified by the **IN** register.

Item	Description
PD	Current inter-paragraph distance.

Flags

Item	Description
-rs1	Reduces default page size of 8.5 inches by 11 inches with a 6.5-inch by 10-inch text area to a 6-inch by 9-inch page size with a 4.75-inch by 8.375-inch text area. This flag also reduces the default type size from 10-point to 9-point and the vertical line spacing from 12-point to 10-point.

Examples

1. To process the file `your.book` and pipe the formatted output to the local line printer, `qprt`, type:

```
nroff -Tlp -man your.book | qprt -dp
```

2. To process the files `my.book` and `dept.book`, which contain tables, and pipe the formatted output to the local line printer, `qprt`, type:

```
tbl my.book dept.book | nroff -Tlp -man | col -Tlp | qprt -dp
```

   Note: Before the output is sent to `qprt`, it is first filtered through the **col** command to process reverse linefeeds used by the **tbl** command.

3. To process the file `group`, which contains pictures, graphs, and tables, and prepare the formatted output for processing on the IBM 3816 printer, enter:

```
grap group | pic | tbl | troff -Tibm3816 -man \
  | ibm3816 | qprt -dp
```

Note:

1. If manual pages created with the **man** macro package are intended for an online facility, components requiring the **troff** command, such as the **grap** or **pic** command, should be avoided.
2. The **grap** command precedes the **pic** command because it is a preprocessor to the **pic** command; the reverse does not format correctly.
3. The **col** command is not required as a filter to the **tbl** command; typeset documents do not require reverse linefeeds.

me Macro Package for the nroff and troff Commands

The **me** package of the **nroff** and **troff** command macro definitions provides a formatting facility for technical papers in various formats. The **col** command may be required to postprocess **nroff** output in certain cases.

The macro requests are defined in the following section, in **me Requests**. Many **nroff**/**troff** requests can have unpredictable results in conjunction with this package. However, the following requests can be used after the first **.pp** request:

Item	Description
.bp	Begins new page.
.br	Breaks output line here.
.ce [*Number*]	Centers next specified number of lines. Default is 1 (one).
.ls [*Number*]	Sets line spacing. Text is single-spaced if *Number* is set to 1 (one); double-spaced if the value is set to 2.
.na	Leaves right margin unjustified.

Item	Description
.sp [*Number*]	Inserts the specified number of spacing lines.
.sz [*+*]*Number*	Adds the specified number to point size.
.ul [*Number*]	Underlines next specified number of lines. Default is 1 (one).

Output of the **eqn**, **neqn**, **refer**, and **tbl** commands preprocessors for equations and tables can be used as input.

me Requests

The following list contains all macros, strings, and number registers available in the **me** macros. Selected **troff** commands, registers, and functions are included.

Item	Description
\(space)	Defines unpaddable space (**troff** command built-in function).
\“	Comments to end of line (**troff** command built-in function).
***#**	Indicates optional delayed text tag string.
\$*Number*	Interpolates the value specified by the *Number* variable (**troff** command built-in function).
\n($0	Defines section depth (number register).
.$0	Started after section title printed (user-definable macro).
\n($1	Defines first section number (number register).
.$1	Started before printing depth 1 (one) section (user-definable macro).
\n($2	Defines second section number (number register).
.$2	Started before printing depth 2 section (user-definable macro).
\n($3	Defines third section number (number register).
.$3	Started before printing depth 3 section (user-definable macro).
\n($4	Defines fourth section number (number register).
.$4	Started before printing depth 4 section (user-definable macro).
\n($5	Defines fifth section number (number register).
.$5	Started before printing depth 5 section (user-definable macro).
\n($6	Defines sixth section number (number register).
.$6	Started before printing depth 6 section (user-definable macro).
.$C	Called at beginning of chapter (user-definable macro).
.$H	Indicates text header (user-definable macro).
\n($R	Defines relative vertical spacing in displays (number register defined by default; changing is not recommended).
\n($c	Defines current column header (number register).
.$c	Prints chapter title (macro defined by default; changing is not recommended).
\n($d	Indicates delayed text number (number register).
\n($f	Indicates footnote number (number register).
.$f	Prints footer (macro defined by default; changing is not recommended).
.$h	Prints header (macro defined by default; changing is not recommended).

Item	Description
\n($i	Defines paragraph base indent (number register).
\n($l	Defines column width (number register).
\n($m	Indicates number of columns in effect (number register).
***($n**	Indicates section name (string).
\n($p	Defines numbered paragraph number (number register).
.$p	Prints section heading (macro defined by default; changing is not recommended).
\n($r	Defines relative vertical spacing in text (number register defined by default; changing is not recommended).
\n($s	Defines column indent (number register).
.$s	Separates footnotes from text (macro defined by default; changing is not recommended).
\n%	Defines current page number (number register defined by default; changing is not recommended).
\&	Indicates zero-width character; useful for hiding controls (**troff** command built-in function).
\(*XX*	Interpolates special character specified by the *XX* variable (**troff** command built-in function).
.(b	Begins block (macro).
.(c	Begins centered block (macro).

Item	Description
.(d	Begins delayed text (macro).
.(f	Begins footnote (macro).
.(l	Begins list (macro).
.(q	Begins quote (macro).
.(x*Index*	Begins indexed item in the specified index (macro).
.(z	Begins floating keep (macro).
.)b	Ends block (macro).
.)c	Ends centered block (macro).
.)d	Ends delayed text (macro).
.)f	Ends footnote (macro).
.)l	Ends list (macro).
.)q	Ends quote (macro).
.)x	Ends index entry (macro).
.)z	Ends floating keep (macro).
*********String***	Interpolates the value specified by the *String* variable (**troff** command built-in function).
*********String1String2***	Interpolates the value specified by the *String1String2* variable (**troff** command built-in function).

Item	Description
******	Indicates optional footnote tag string.
.++*mH*	Macro to define paper section. The value specified by the *m* variable defines the part of the paper. The *m* variable can have the following values: **C** Defines chapter. **A** Defines appendix. **P** Defines preliminary information, such as abstract and table of contents. **B** Defines bibliography. **RC** Defines chapters to be renumbered from page 1 (one) of each chapter. **RA** Defines appendix to be renumbered from page 1 (one). The *H* parameter defines the new header. If there are any spaces in it, the entire header must be quoted. If you want the header to have the chapter number in it, use the string **\\\n(ch**. For example, to number appendixes A.1, A.2, ..., type: `.++ RA ''\\n(ch.%'`. Each section (such as chapters and appendixes) should be preceded by the **.+c** request.
.+c*Title*	Begins chapter (or appendix, for instance, as set by the **.++** macro). The value specified by the *Title* variable is the chapter title (macro).
***,**	Indicates cedilla (string).
\-	Indicates minus sign (**troff** command built-in function).
***-**	Indicates 3/4 em dash (string).
\0	Defines unpaddable digit-width space (**troff** command built-in function).
.1c	Reverts to single-column output (macro).
.2c	Begins two-column output (macro).
***:**	Indicates umlaut (string).
***<**	Begins subscript (string).
***>**	Ends subscript (string).
.EN	Ends equation. Space after equation produced by the **eqn** command or **neqn** command (macro).
.EQ*XY*	Begins equation; breaks out and adds space. The value specified by the *Y* variable is the equation number. The optional *X* variable value might be any of the following: **I** Indents equation (default). **L** Left-adjusts equation. **C** Centers equation (macro).
\L'*Distance***'**	Indicates vertical line-drawing function for the specified distance (**troff** command built-in function).

Item	Description
.PE	Ends pic picture (macro).
.PF	Ends pic picture with flyback (macro).
.PS	Starts pic picture (macro).
.TE	Ends table (macro).
.TH	Ends header of table (macro).
.TS *X*	Begins table. If the value of the *X* variable is **H**, the table has a repeated heading (macro).
***[**	Begins superscript (string).
\n(.$	Defines number of options to macro (number register defined by default; changing is not recommended).
\n(.i	Indicates current indent (number register defined by default; changing is not recommended).
\n(.l	Indicates current line length (number register defined by default; changing is not recommended).

Item	Description
\n(.s	Indicates current point size (number register defined by default; changing is not recommended).
***(**4	Indicates acute accent (string).
***(`**	Indicates grave accent (string).
\(4	Indicates acute accent (**troff** command built-in function).
\(`	Indicates grave accent (**troff** command built-in function).
***]**	Ends superscript (string).
\^	Indicates 1/12 em narrow space (**troff** command built-in function).
***^**	Indicates caret (string).
.ac*AuthorNumber*	Sets up for ACM-style output. The *Author* variable specifies the author name or names. The *Number* variable specifies the total number of pages. Must be used before the first initialization (macro).
.ad	Sets text adjustment (macro).
.af	Assigns format to register (macro).
.am	Appends to macro (macro).
.ar	Sets page numbers in Arabic (macro).
.as	Appends to string (macro).
.b *X*	Prints in boldface the value specified by the *X* variable. If the *X* variable is omitted, boldface text follows (macro).
.ba +*Number*	Augments the base indent by the specified *Number* value. Sets the indent on regular text such as paragraphs (macro).
.bc	Begins new column (macro).
.bi *X*	Prints in bold italic the value specified by the *X* parameter, in no-fill mode only. If the *X* parameter is not used, bold italic text follows (macro).
\n(bi	Displays block indent (number register).

Item	Description
.bl	Requests blank lines, even at top of page (macro).
\n(bm	Sets bottom title margin (number register).
.bp	Begins page (macro).
.br	Sets break; starts new line (macro).
\n(bs	Displays block pre- or post-spacing (number register).
\n(bt	Blocks keep threshold (number register).
.bu	Begins bulleted paragraph (macro).
.bx *X*	Prints in no-fill mode only the value specified by the *X* variable in box (macro).
\c	Continues input (**troff** command built-in function).
.ce	Centers lines (macro).
\n(ch	Defines current chapter number (number register).
.de	Defines macro (macro).
\n(df	Displays font (number register).
.ds	Defines string (macro).
\n(dw	Defines current day of week (number register).
***(dw**	Defines current day of week (string).
\n(dy	Defines current day of month (number register).
\e	Indicates printable version of \ (backslash) (**troff** command built-in function).
.ef'*X***'***Y***'***Z***'**	Sets even-page footer to the values specified by the *XYZ* variables (macro).
.eh'*X***'***Y***'***Z***'**	Sets even-page header to the values specified by the *XYZ* variables (macro).
.el	Specifies the else part of an if/else conditional (macro).
.ep	Ends page (macro).

Item	Description
\n(es	Indicates equation pre- or post-space (number register).
\f*Font*	Sets inline font change to the specified *Font* variable value (**troff** command built-in function).
\f(*Fontf*	Sets inline font change to the specified *Fontf* variable value (**troff** command built-in function).
.fc	Sets field characters (macro).
\n(ff	Sets footnote font (number register).
.fi	Fills output lines (macro).
\n(fi	Indicates footnote indent, first line only (number register).
\n(fm	Sets footer margin (number register).
.fo '*X***'***Y***'***Z***'**	Sets footer to the values specified by the *XYZ* variables (macro).
\n(fp	Sets footnote point size (number register).
\n(fs	Sets footnote pre-space (number register).
\n(fu	Sets footnote indent from right margin (number register).

Item	Description
\h'*Distance***'**	Sets local horizontal motion for the specified distance (**troff** command built-in function).
.hc	Sets hyphenation character (macro).
.he '*X***'***Y***'***Z***'**	Sets header to the values specified by the *XYZ* variables (macro).
.hl	Draws horizontal line (macro).
\n(hm	Sets header margin (number register).
.hx	Suppresses headers and footers on next page (macro).
.hy	Sets hyphenation mode (macro).
.i *X*	Italicizes the value specified by the *X* variable. If the *X*variable is omitted, italic text follows (macro).
.ie	Specifies the else part of an if/else conditional (macro).
.if	Designates a conditional (macro).
\n(ii	Sets indented paragraph indent (number register).
.in	Indents (transient); use the **.ba** macro if pervasive (macro).
.ip *X Y*	Starts indented paragraph, with hanging tag specified by the *X* variable. Indentation is the en value specified by the *Y* variable. Default is 5 (macro).
.ix	Indents, no break (macro).
\l'*Distance***'**	Starts horizontal line-drawing function for the specified distance (**troff** command built-in function).
.lc	Sets leader repetition character (macro).
.lh	Interpolates local letterhead (macro).
.ll	Sets line length (macro).
.lo	Reads in a file of local macros of the form .*x. Must be used before initialization (macro).
.lp	Begins left-justified paragraph (macro).
***(lq**	Designates left quotation marks (string).
.ls	Sets multi-line spacing (macro).
.m1	Sets space from top of page to header (macro).
.m2	Sets space from header to text (macro).
.m3	Sets space from text to footer (macro).
.m4	Sets space from footer to bottom of page (macro).
.mc	Inserts margin character (macro).
.mk	Marks vertical position (macro).
\n(mo	Defines month of year (number register).

Item	Description
***(mo**	Defines current month (string).
\n*X*	Interpolates number register specified by the *X* variable value (number register).
\n(*XX*	Interpolates number register specified by the *XX* variable (number register).

Item	Description
.n1	Sets number lines in margin (macro).
.n2	Sets number lines in margin (macro).
.na	Turns off text adjustment (macro).
.ne*Number*	Sets the specified number of lines of vertical space (macro).
.nf	Leaves output lines unfilled (macro).
.nh	Turns off hyphenation (macro).
.np	Begins numbered paragraph (macro).
.nr	Sets number register (macro).
.ns	Indicates no-space mode (macro).
***o**	Indicates superscript circle (such as for Norse A; string).
.of'*X***'***Y***'***Z***'**	Sets odd footer to the values specified by the *XYZ* variables (macro).
.oh'*X***'***Y***'***Z***'**	Sets odd header to the values specified by the *XYZ* variables (macro).
.pa	Begins page (macro).
.pd	Prints delayed text (macro).
\n(pf	Indicates paragraph font (number register).
\n(pi	Indicates paragraph indent (number register).
.pl	Sets page length (macro).
.pn	Sets next page number (macro).
.po	Sets page offset (macro).
\n(po	Simulates page offset (number register).
.pp	Begins paragraph, first line indented (macro).
\n(pp	Sets paragraph point size (number register).
\n(ps	Sets paragraph pre-space (number register).
.q	Indicates quoted (macro).
***(qa**	For all (string).
***qe**	There exists (string).
\n(qi	Sets quotation indent; also shortens line (number register).
\n(qp	Sets quotation point size (number register).
\n(qs	Sets quotation pre- or post-space (number register).
.r	Sets Roman text to follow (macro).
.rb	Sets real bold font (macro).
.re	Resets tabs to default values (macro).
.rm	Removes macro or string (macro).
.rn	Renames macro or string (macro).
.ro	Sets page numbers in Roman (macro).
***(rq**	Indicates right quotation marks (string).
.rr	Removes register (macro).

Item	Description
.rs	Restores register (macro).

Item	Description
.rt	Returns to vertical position (macro).
\s*Size*	Changes inline size to specified size (**troff** command built-in function).
.sc	Reads in a file of special characters and diacritical marks. Must be used before initialization (macro).
\n(sf	Sets section title font (number register).
.sh*LevelTitle*	Indicates section head to follow; font automatically bold. The *Level* variable specifies the level of section. The *Title* variable specifies the title of section (macro).
\n(si	Sets relative base indent-per-section depth (number register).
.sk	Leaves the next page blank. Only one page is remembered ahead (macro).
.sm*X*	Sets, in a smaller point size, the value specified by the *X* variable (macro).
.so	Indicates source input file (macro).
\n(so	Sets additional section title offset (number register).
.sp	Indicates vertical space (macro).
\n(sp	Indicates section title point size (number register).
\n(ss	Indicates section prespace (number register).
.sx	Changes section depth (macro).
.sz +*Number*	Augments point size by the specified number of points (macro).
.ta	Sets tab stops (macro).
.tc	Sets tab repetition character (macro).
***(td**	Sets today's date (string).
n(tf	Indicates title font (number register).
.th	Produces paper in thesis format. Must be used before initialization (macro).
.ti	Indicates temporary indent, next line only (macro).
.tl	Indicates 3-part title (macro).
\n(tm	Sets top title margin (number register).
.tp	Begins title page (macro).
\n(tp	Sets title point size (number register).
.tr	Translates (macro).
.u *X*	Underlines the value specified by the *X* variable, even in the **troff** command. No-fill mode only (macro).
.uh	Sets section head to follow; font automatically bold. Similar to the **.sh** macro, but unnumbered (macro).
.ul	Underlines next line (macro).
\v'*Distance***'**	Local vertical motion for the specified distance (**troff** command built-in function).
***v**	Inverts v for Czech e (string).

Item	Description
\w'*String***'**	Returns width of the specified string (**troff** command built-in function).
.xl	Sets local line length (macro).
.xp*Index*	Prints the specified index (macro).
\n(xs	Sets index entry prespace (number register).
\n(xu	Sets index indent, from right margin (number register).
\n(yr	Indicates year, last two digits only (number register).
\n(zs	Sets floating keep pre- or post-space (number register).
\{	Begins conditional group (**troff** command built-in function).
|	1/6 em, narrow space (**troff** command built-in function).
\}	Ends conditional group (**troff** command built-in function).
***~**	Indicates tilde (string).

For further information, see the *-ME Reference Manual* by E. P. Allman.

mm Macro Package for the mm, mmt, nroff, and troff Commands

The **mm** macro package provides macros to format text in a wide variety of document forms, such as memos, letters, and reports. The manner in which you type and edit a document is essentially independent of whether the document is later formatted at a terminal or phototypeset.

The **col** command may be required to postprocess **nroff** output. See the **col** command for specific requirements.

The **mm** macros and additional information are summarized under the following headings:

- Beginning Macros for Formal Memoranda
- Business Letter Macros
- Ending Macros (Trailing Information)
- Paragraphs
- Section Headings
- Lists
- Displays, Tables, Equations, and Footnotes
- Page Headers and Footers
- Miscellaneous Macros
- **mm** Registers
- **mm** Strings
- String Names
- Reserved Names.

Beginning Macros for Formal Memoranda

Item	Description
.ND *Date*	Sets new date.
.TL [*ChgNumber*] [*FileNumber*]	Sets title information. Text on the following line is used as the title of the document.
.AF [*CompanyName*]	Specifies author's company name.

Item	Description
.AU *Name* [*Initials*] [*Loc*] [*Dept*] [*Ext*] [*Room*] [*Option...*]	Sets author information.
.AT *AuthorTitle* [*...*]	Specifies title to follow signer's name (up to nine options).
.TM [*Number*]	Sets technical memorandum number.
.AS [**0 | 1 | 2**] [*Indent*]	Starts abstract, for technical memorandum and released paper only: **0** Abstract on cover sheet and first page **1** Abstract only on cover sheet **2** Abstract only on memorandum for file cover sheet.
.AE	Ends abstract.
.NS	Starts notation, allowed on memorandum for file cover sheets following an **.AS 2/.AE** macro pair (see "Ending Macros").
.NE	Ends notation, allowed on memorandum for file cover sheets following an **.AS 2/.AE** macro pair (see "Ending Macros").
.OK [*Keyword ...*]	Specifies other keywords (up to nine options).
.MT [*type*] [*title*]	Sets document type: **""** No type. **0** No type (internal letter). **1** Memorandum for file. **2** Programmer's notes. **3** Engineer's notes. **4** Released paper. **5** External letter. ***"String"*** The specified string is printed.
Title	User-supplied text prefixed to page number

Business Letter Macros

Item	Description
.WA	Starts writer's address.

Item	Description
.WE	Ends writer's address.
.LO CN [*Notation*]	Specifies confidential notation.
.LO RN [*Notation*]	Specifies reference notation.
.IA	Starts inside (recipient's) address.
.IE	Ends inside (recipient's) address.
.LO AT [*Notation*]	Specifies attention line.
.LO SA [*Notation*]	Specifies salutation.
.LO SJ [*Notation*]	Specifies subject line.
.LT [{ none **BL SB FB SP**}]	Specifies business letter type: **none** Blocked **BL** Blocked **SB** Semiblocked **FB** Full-Blocked **SP** Simplified.

Ending Macros (Trailing Information)

Item	Description
.FC [*Closing*]	Prints formal closing.
.SG [*Initials*] [**1**]	Prints signature line.

Item	Description
.NS [{**" "0 1 2 3 4 5 6 7 8 9 10 11 12 13** *String*}]	Starts notation: **" "** Copy to **0** Copy to **1** Copy (with attachment) to **2** Copy (without attachment) to **3** Attachment **4** Attachments **5** Enclosure **6** Enclosures **7** Under Separate Cover **8** Letter to **9** Memorandum to **10** Copy (with attachments) to **11** Copy (without attachments) to **12** Abstract Only to **13** Complete Memorandum to ***String*** Copy (*String*) to.
.NE	Ends notation.
.AV *Name* [**1**]	Prints approval signature.
.CS [*Pgs*] [*Other*] [*Tot*] [*Figs*] [*Tbls*] [*Ref*]	Prints cover sheet.
.TX	Calls user exit for table-of-contents titles.
.TY	Calls user exit for table-of-contents header.
.TC [*Slev*] [*Spacing*] [*Tlev*] [*Tab*] [*H1*] [*H2*] [*H3*] [*H4*] [*H5*]	Prints table of contents.

Paragraphs

Item	Description
.P [{**0 1 2**}]	Starts paragraph: **0** Left-justified (default) **1** Indented **2** Indented except after **.H**, **.LE**, **.DE**.

Section Headings

Item	Description
.H {**1 2 3 4 5 6 7**} [*HeadingText*] [*FootnoteMark*]	Specifies numbered headings.
.HU *HeadingText*	Specifies unnumbered headings.
.HM {**1 0001 A a I i**}**...**	Specifies heading mark style: **1** Arabic **0001** Arabic with leading 0s (zeros) **A** Uppercase alphabetic **a** Lowercase alphabetic **I** Uppercase Roman **i** Lowercase Roman.
.HX [*Dlev*] [*Rlev*] [*HeadingText*]	Calls user-defined exit macro before headings.
.HY [*Dlev*] [*Rlev*] [*HeadingText*]	Calls user-defined exit macro in the middle of headings.
.HZ [*Dlev*] [*Rlev*] [*HeadingText*]	Calls user-defined exit macro after headings.

Lists

If the last option [**1**] is present in the list-start macros, there is no space between items.

Item	Description
.AL [{**1 A a I i**}] [*TextIndent*] [**1**]	Starts automatically incremented list (**1**).
.BL [*TextIndent*] [**1**]	Starts a bullet list.
.DL [*TextIndent*] [**1**]	Starts a dash list.
.ML *Mark* [*TextIndent*] [**1**]	Starts a list in which each list item is tagged with a specified mark. If the value of the *TextIndent* is **NULL** or omitted, it is set to [*Mark* - width + 1]. If the 3rd argument is specified, no blank lines separate items in the list.
.RL [*TextIndent*] [**1**]	Starts a reference list.
.VL *TextIndent* [*MarkIndent*] [**1**]	Starts a variable tag list.

Item	Description
.LI [*Mark*] [**1**]	Starts list item; **1** means that the *Mark* variable value is to be prefixed to the current mark.
.LE [**1**]	Ends list item; **1** means to output a blank line after list. The default is no blank line.
.LB *TextIndent MarkIndent Pad Type* [*Mark*] [{**0 1**}] [{**0 1**}]	Begins list: The value of the *Type* variable is: **1=. 2=) 3=() 4=[] 5=<> 6={}**. Sixth option: **0** No blank line before each list item. Seventh option: **0** No blank line before list.
.LC [*Level***]**	Clears list status up to the *Level* variable value.

Displays, Tables, Equations, and Footnotes

.DS [{**0 1 2 3** }] [{**0 1**}] [*Number*]

.DS [{L I C CB}] [{N F}] [*Number*]
Starts static display:

0 or L
No indent

1 or I
Indent from left

2 or C
Center each line

3 or CB
Center as a block

0 or N
No-fill

1 or F
Fill.

Number
Indent from right the number of spaces specified by the *Number* parameter.

.DF [{**0 1 2 3** }] [{**0 1**}] [*Number*]

.DF [{L I C CB}] [{N F}] [*Number*]
Starts floating display:

0 or L
No indent

1 or I
Indent from left

2 or C
Center each line

3 or CB
Center as a block

0 or N
No-fill

1 or F
Fill.

Number
Indent from right the number of spaces specified by the *Number* parameter.

.DE
Ends display.

.FG [*Title*] [*Override*] [0 1 2]
The value of the *Override* variable replaces or enhances the default numbering. Specifies figure caption:

0
Override value is used as a prefix.

1
Override value becomes a suffix.

2
Replace *Override* value becomes a replacement.

.TS [H]
Starts table:

H
Multipage table.

.TH [N]
Must be used when specifying option **H** to **.TS**:

N
Suppresses table headers unless on top of new page.

.TE
Ends table.

.TB [*Title*] [*Override*] [0 1 2]
The value of the *Override* variable replaces or enhances the default numbering. Specifies table caption:

0
Override value is used as a prefix.

1
Override value becomes a suffix.

2
Replace *Override* value becomes a replacement.

.EX [*Title*] [*Override*] [0 1 2]
The value of the *Override* variable replaces or enhances the default numbering. Specifies exhibit caption:

0
Override value is used as a prefix.

1
Override value becomes a suffix.

2
Replace *Override* value becomes a replacement.

.EQ [*Label*]
Starts equation display using the specified label.

.EN
Ends equation display.

.EC [*Title*] [*Override*] [0 1 2]
The value of the *Override* variable replaces or enhances the default numbering. Specifies equation caption:

0
Override value is used as a prefix.

1
Override value becomes a suffix.

2
Replace *Override* value becomes a replacement.

.FS [*Label*]
Starts footnote using the specified label as an indicator. Default is numbered footnote.

.FE
Ends footnote.

.FD [{0 1 2 3 4 ... 11}] [1]
Sets footnote format:

First option:

Set up formatting style for footnote text. Default is 0 for **mmt** command. Default is 10 for **mm** command. See the following table for the value.

Second option:

Reset footnote counter on first-level heading.

.FD Arg.	Format
0	**Hyphens** .nh **Adjusted** .ad **Text Indented** Yes **Label Justified** Left
1	**Hyphens** .hy **Adjusted** .ad **Text Indented** Yes **Label Justified** Left

.FD Arg.	Format
2	**Hyphens** .nh **Adjusted** .na **Text Indented** Yes **Label Justified** Left
3	**Hyphens** .hy **Adjusted** .na **Text Indented** Yes **Label Justified** Left
4	**Hyphens** .nh **Adjusted** .ad **Text Indented** No **Label Justified** Left
5	**Hyphens** .hy **Adjusted** .ad **Text Indented** No **Label Justified** Left
6	**Hyphens** .nh **Adjusted** .na **Text Indented** No **Label Justified** Left

.FD Arg.	Format
7	**Hyphens** .hy **Adjusted** .na **Text Indented** No **Label Justified** Left
8	**Hyphens** .nh **Adjusted** .ad **Text Indented** Yes **Label Justified** Right
9	**Hyphens** .hy **Adjusted** .ad **Text Indented** Yes **Label Justified** Right
10	**Hyphens** .nh **Adjusted** .na **Text Indented** Yes **Label Justified** Right
11	**Hyphens** .hy **Adjusted** .na **Text Indented** Yes **Label Justified** Right

Page Headers and Footers

Item	Description
.PH "'*Left***'***Center***'***Right***'"**	Specifies page header.

Item	Description
.OH ”'*Left*'*Center*'*Right*'“**	Specifies odd-page header.
.EH ”'*Left*'*Center*'*Right*'“**	Specifies even-page header.
.PF ”'*Left*'*Center*'*Right*'“**	Specifies page footer.
.OF ”'*Left*'*Center*'*Right*'“**	Specifies odd-page footer.
.EF ”'*Left*'*Center*'*Right*'“**	Specifies even-page footer.
.BS	Starts bottom-block.
.BE	Ends bottom-block.
.PX	Calls user exit for page-header.
.TP	Calls top of page macro.

Miscellaneous Macros

Item	Description
.B [*Option*] [*Prev-Font-option*]	Prints in bold (up to six options).
.I [*Option*] [*Prev-Font-option*]	Prints in italics (up to six options); underlines with the **nroff** command.
.R	Returns to Roman font.
.PM [*Option*]	Sets proprietary marking. If you do not give the **.PM** macro an option, you turn off proprietary markings. The **/usr/lib/macros/string.mm** file contains some proprietary markings. This file should be edited to meet the user's needs.
.RD [*Prompt*] [*Diversion*] [*String*]	Stops code macro. The *Prompt* variable should be a user-defined string without spaces. The *Diversion* variable allows the typed-in text to be saved. The *String* variable contains the first line typed following the prompt.
.RP [{**0 1** }] [{**0 1 2 3**}]	Produces reference page: First option: **0** Resets reference counter (default). **1** Does not reset reference counter. Second option: **0** Causes an **.SK** macro after (default). **1** Does not cause an **.SK** macro after. **2** Does not cause an **.SK** macro before. **3** Does not cause an **.SK** macro before or after.
.RS/.RF	Numbers references automatically.

Item	Description
.WC [{**N WF -WF FF -FF WD -WD FB -FB**}]	Controls width for footnotes and displays when using two columns: **N** Normal mode (**-WF**, **-FF**, **-WD**). **WF** Footnotes always wide. **-WF** Footnotes follow page style. **FF** First footnote determines width of remaining footnotes on that page. **-FF** Footnotes follow setting of **WF** or **-WF** option. **WD** Always wide displays. **-WD** Displays follow page style. **FB** Floating display causes page break (default). **-FB** Floating display does not cause page break.
.SP [*Lines*]	Skips lines down.
.SK [*Number*]	Skips the specified number of pages. (The default is 1.)
.OP	Breaks to an odd page.
.2C	Prints output in two columns.
.1C	Prints output in one column (normal line width restored).
.SA [*Option*]	Sets right-margin justification Options: **0** Sets default to **off** (default for the **nroff** command). **1** Sets default to **on** (default for the **troff** command). If no option is specified, macro reverts to current default.
.SM *String1* [*String2*] [*String3*]	Reduces size of the *String1* variable value by 1 point if the *String3* variable value is omitted; otherwise, reduces size of the *String2* variable value by one point.
.HC *Character*	Sets hyphenation character to the *Character* variable value.

Item	Description
.S [*PointSize*] [*VerticalSpacing*]	Sets point size and vertical spacing (the **troff** command only). Defaults: Point size = **10p** Vertical spacing = **12p** Options 1 and 2: ***Number*** New value. ***+/-Number*** Increment to current value. **D** Default. **C** Current value. **P** Previous value.
.VM [*Top*] [*Bottom*]	Sets variable vertical margins.
.nP	Starts double-line indent on paragraph.

The following macros are for alternating fonts and all take one to six options:

Item	Description
.IB	Alternates italics (underlines for **nroff**) and bold.
.BI	Alternates bold and italics.
.RI	Alternates Roman and italics.
.IR	Alternates italics (underlines for **nroff**) and Roman.
.RB	Alternates Roman and bold.
.BR	Alternates bold and Roman.

mm Registers

If an * (asterisk) follows a register name, that register can be set one of two ways: from the command line (see the example in the **mm** command) or before the formatter reads **mm** macro definitions. In the following list, the number shown in parentheses is the default value.

Item	Description
A *	Handle preprinted forms.
Au	Inhibit author information on first page (1).
C *	Copy type (such as Original and Draft) (0).
Cl	Contents level (2).
Cp	Placement of figures, tables, equations, and exhibits (1).
D *	Debug flag (0). If set to 1, the **mm** command continues even if it encounters an error that is usually fatal.
De	Eject page after floating displays (0).
Df	If set to 1, format register for floating displays (5).

Item	Description
Ds	Static display pre- and post-space (1).
E *	Control font of the Subject/Date/From fields (0): 0 = bold; 1 = Roman. **0** Bold (0) **1** Roman.
Ec	Equation counter.
Ej	Page-ejection flag for headings (0).
Eq	Equation label placement (0).
Ex	Exhibit counter.
Fg	Figure counter.
Fs	Vertical footnote separation (1).
H1...H7	Heading counters.
Hb	Heading break level (after **.H** and **.HU**) (2).
Hc	Heading centering level for **.H** and **.HU** (0).
Hi	Heading temporary indent (after **.H** and **.HU**) (1).
Hs	Heading space level (after **.H** and **.HU**) (2).
Ht	Heading type: **0** Concatenated numbers (0) **1** Single numbers (0).
Hu	Heading level for unnumbered heading (2).
Hy	Hyphenation control: **0** No hyphenation (0) **1** Enable hyphenation.
L *	Length of page (66v).
Le	List of equations following table of contents (0): **0** Do not print **1** Print.
Lf	List of figures following table of contents (0): **0** Do not print **1** Print.
Li	List indent (5, **troff** command); (6, **nroff** command).
Ls	List level down to which there is spacing between items (6).

Item	Description
Lt	List of tables following table of contents (0): **0** Do not print **1** Print
Lx	List of exhibits following table of contents (1): **0** Do not print **1** Print.

Item	Description
N *	Numbering style (0).
Np	Numbered paragraphs: **0** Unnumbered **1** Numbered (0).
O *	Offset of page.
Oc	Page numbering style for table of contents: **0** Lowercase Roman **1** Arabic (0).
Of	Figure caption style (0).
P	Page number; managed by the **mm** command (0). The register accepts a value of 0, or positive integers.
Pi	Paragraph indent (5).
Ps	Paragraph spacing (1).
Pt	Paragraph type (0).
Pv	PRIVATE header: **0** Do not print PRIVATE **1** On first page only
2	On all pages (0).
Rf	Reference counter; used by **.RS** macro.
S *	The **troff** command's default point size (10).
Si	Display indent (5).
T *	Type of the **nroff** command output device (0).
Tb	Table counter.
U *	Underlining style (the **nroff** command) for **.H** and **.HU** (0).

Item	Description
W *	Width of page (line and title length).

mm Strings

Print special strings by using the following escape sequences:

Item	Description
***x**	For strings with single-character names (**x**)
***(xx**	For strings with two-character names (**xx**).

String Names

Item	Description
BU	Bullet.
Ci	Indent of heading levels in the table of contents.
DT	Current date. The locale-specific date format specified by the locale setting for the **LC_TIME** category is used as the default setting. This corresponds to the **%x** format specifier of the **strftime** subroutine. Use the **.ND** macro to change the current date.
EM	Em dash.
F	Footnote numbering.
HF	Heading level font string: **1** Roman **2** italics **3** Bold (2 2 2 2 2 2 2).
HP	Point sizes of the various heading levels.
Le	Title of the list of equations.
Lf	Title of the list of figures.
Lt	Title of the list of tables.
Lx	Title of the list of exhibits.
RE	SCCS SID of **mm** macros.
Rf	Reference numberer.
Rp	Title of the reference page.
Tm	Trademark.
`	Grave accent.
'	Acute accent.
^	Circumflex.
~	Tilde.
:	Lowercase umlaut.
;	Uppercase umlaut.
,	Cedilla.

Reserved Names

If you define your own strings, macros, and registers, use only names that consist of either a single lowercase letter, or a lowercase letter followed by any character other than a lowercase letter. The names **c2** and **nP** are exceptions to this; they are reserved.

mptx Macro Package for the nroff and troff Commands

The **mptx** macro package provides a definition for the **.xx** macro that is used for formatting a permuted index produced by the **ptx** command. The **mptx** macro package does not provide any other formatting capabilities, such as headers and footers. Use the **mptx** macro package in conjunction with the **mm** macro package if such capabilities are required. In this case, call the **-mptx** option after the **-mm** call, as follows:

nroff -mm -mptx *File...* | *Printer*

ms Macro Package for the nroff and troff Commands

The ms macro package of **nroff** and **troff** command macro definitions provides a formatting facility for various styles of articles, theses, and books. In certain cases, the **col** command may be required to postprocess output.

The macro requests are defined in the **ms** Requests section. Many **nroff** and **troff** command requests can have unpredictable results in conjunction with this package. However, the first 4 requests in the following list can be used after initialization, and the last 2 requests can be used before initialization.

Item	Description
.bp	Begins new page.
.br	Breaks output line.
.ce [*Number*]	Centers the next specified number of lines.
.ls [*Number*]	Sets line spacing. Set the value of the *Number* variable to 1 (one) to single-space text; and to 2 to double-space text.
.na	Turns off alignment of right margin.
.sp [*Number*]	Inserts the specified number of spacing lines.

Font and point-size changes with the**\f** and **\s** macros are also allowed. For example, `\fIword\fR` italicizes `word`. Output of the **tbl**, **eqn**, and **refer** command preprocessors for equations, tables, and references is acceptable as input.

Formatting distances can be controlled in **ms** macros by means of built-in number registers. For example, the following number register sets the line length to 6.5 inches:

```
.nr LL 6.5i
```

For more information on **ms** macro registers, see **ms** Registers.

ms Requests

Following are external **ms** macro requests:

Item	Description
.AB [**X**]	Begins abstract. If **X** is no, do not label abstract. Initial Value: **-** Break: **yes**

Item	Description
.AE	Ends abstract. Break: **yes**Initial Value: **-** Break: **yes**
.AI*Name*	Author's institution. Initial Value: **-** Break: **yes**
.AM	Sets accent mark definitions. Initial Value: **-** Break: **no**
.AU*Name*	Sets author's name. Initial Value: **-** Break: **yes**
.B [**X**]	Puts **X** in boldface. If no **X**, switches to boldface. Initial Value**: -** Break: **no**
.B1	Begins text to be enclosed in a box. Initial Value: **-** Break: **yes**
.B2	Ends boxed text and prints it. Initial Value: **-** Break: **yes**
.BT	Prints bottom title at foot of page. Initial Value: **date** Break: **no**
.BX X	Prints word **X** in a box. Initial Value: **-** Break: **no**
.CM	Cuts mark between pages. Initial Value: **if t** Break: **no**
.CT	Indicates chapter title; page number moved to CF (TM). Initial Value: **-** Break: **yes** Reset: **yes**

Item	Description
.DA [**X**]	Forces date **X** at bottom of page. If no **X**, date is today. Initial Value: **if n** Break: **no**
.DE	Ends display (unfilled text) of any kind. Initial Value: **-** Break: **yes**
.DS X Y	Begins display with keep. **X**=I, L, C, B; **Y**=indent. Initial Value: **I** Break: **yes**
.ID Y	Indents display with no keep; **Y**=indent. Initial Value: **8n, .5i** Break: **yes**
.LD	Sets left display with no keep. Initial Value: **-** Break: **yes**
.CD	Centers display with no keep. Initial Value: **-** Break: **yes**
.BD	Block display; centers entire block. Initial Value: **-** Break: **yes**
.EF X	Sets even page footer **X** (3 part as for **troff** command, **.tl** request). Initial Value: **-** Break: **no**
.EH X	Sets even page header **X** (3 part as for **troff** command, **.tl** request). Initial Value: **-** Break: **no**
.EN	Ends displayed equation produced by **eqn** command. Initial Value: **-** Break: **yes**
.EQ [**X**] [**Y**]	Breaks out equation. **X**=L, I, C; **Y** is equation number. Initial Value: **-** Break: **yes**

Item	Description
.FE	Ends footnote to be placed at bottom of page. Initial Value: **-** Break**: no**
.FP	Numbers footnote paragraph; can be redefined. Initial Value: **-** Break: **no**
FS [**X**]	Starts footnote; **X** is optional footnote label. Initial Value: **-** Break: **no**
.HD	Sets optional page header below header margin. Initial Value: **undef** Break: **no**
.I [**X**]	Italicizes **X**. If no **X**, equivalent to italics font **.ft 2**. Initial Value: **-** Break: **no**
.IP X Y	Indents paragraph, with hanging tag **X**. **Y** specifies spaces to indent. Initial Value: **-** Break: **yes** Reset: **yes**
.IX X Y	Indexes words such as **X** and **Y**, up to five levels. Initial Value: **-** Break: **yes**
.KE	Ends keep of any kind. Initial Value: **-** Break: **no**
.KF	Begins floating keep; text fills remainder. Initial Value: **-** Break: **no**
.KS	Begins keep; keeps unit together on a single page. Initial Value: **-** Break: **yes**

Item	Description
.LG	Sets larger type size; increases point size by 2. Valid only for the **troff** command. Initial Value: **-** Break: **no**

Item	Description
.LP	Begins left block paragraph. Initial Value: **-** Break: **yes** Reset: **yes**
.MC X	Sets multiple columns. **X** is column width. Initial Value: **-** Break: **yes** Reset: **yes**
.ND [**X**]	Indicates no date in page footer; **X** is date on cover. Initial Value: **if t** Break: **no**
.NH X Y	Sets numbered header: **X**=level; **X**=0, resets; **X**=S, sets to **Y**. Initial Value: **-** Break: **yes** Reset: **yes**
.NL	Sets point size back to default. Valid for the **troff** command only. Initial Value: **10p** Break: **no**
.OF X	Sets odd page footer **X** (3 part as for **me** macro, **.tl** request). Initial Value: **-** Break: **no**
.OH X	Sets odd page header **X** (3 part as for **me** macro, **.tl** request). Initial Value: **-** Break: **no**
.P1	Prints header on first page. Initial Value: **if TM** Break: **no**
.PP	Indents first line of paragraph. Initial Value: **-** Break: **yes** Reset: **yes**
.PT	Prints page title at head of page. Initial Value: **%** Break: **no**

Item	Description
.PX X	Prints index (table of contents); **X**=do not suppress title. Initial Value: **-** Break: **yes**
.QP	Quotes paragraph (indented and shorter). Initial Value: **-** Break: **yes** Reset: **yes**
.R [**X**]	Returns to Roman font. Prints in Roman font. If **X** is missing, equivalent to font **.ft1**. Initial Value: **on** Break**: no**
.RE	Retreats (end level of relative indentation). Used with the **.RS** request. Initial Value: **5n** Break: **yes** Reset: **yes**
.RP [**X**]	Prints title page in released paper format; **X**=no, stops title on first page. Initial Value: **-** Break: **no**
.RS	Right-shifts in one indentation level (start level of relative indentation). Used with the **.IP** request. Initial Value: **5n** Break: **yes** Reset: **yes**
.SG	Sets signature line.
.SH	Sets unnumbered section header (in boldface). Initial Value: **-** Break: **yes** Reset: **yes**
.SM	Sets smaller type size; decrease point size by 2. Valid for the **troff** command only. Initial Value: **-** Break: **no**
.TA	Sets tabs to 8n, 16n, ... (**nroff**); 5n, 10n, ... (**troff**). Initial Value: **8n**, **5n** Break: **no**
.TC X	Prints table of contents at end; **X**=do not suppress title. Initial Value: **-** Break: **yes**

Item	Description
.TE	Ends table processed by **tbl** command. Initial Value: **-** Break: **yes**
.TH	Ends multipage header of table. Must be used with the **.TS** H request. Initial Value: **-** Break: **yes**
.TL	Sets title line (in boldface and 2 points larger). Initial Value: **-** Break: **yes**
.TM	Sets UC Berkeley thesis mode. Initial Value: **off** Break: **no**
.TS X	Begins table. If **X** is H, table prints header on all pages. Initial Value: **-** Break: **yes** Reset: **yes**
.UL X	Underlines **X**, even for the **troff** command. Initial Value: **-** Break: **no**
.UX X	Sets UNIX; trademark message first time; **X** appended. Initial Value: **-** Break: **no**
.XA X Y	Sets another index entry; **X**=page; **X**=no, for none. Initial Value: **-** Break: **yes**
.XE	Ends index entry or series of **.IX** request entries. Initial Value: **-** Break: **yes**
.XP	Exdents first line of paragraph; others indented. Initial Value: **-** Break: **yes** Reset: **yes**
.XS X Y	Begins index entry; **X**=page; **X**=no, for none; **Y**=indent. Initial Value: **-** Break: **yes**

Item	Description
.1C	Begins one-column format, on a new page. Initial Value: **on** Break: **yes** Reset: **yes**
.2C	Begins two-column format. Initial Value: **-** Break: **yes** Reset: **yes**
.]-	Sets beginning of **refer** command reference. Initial Value: **-** Break: **no**
.[0	Sets end of unclassifiable type of reference. Initial Value: **-** Break: **no**
.[N	For journal article, **N**=1 (one). For book, **N**=2. For book article, **N**=3. Initial Value: **-** Break: **no**

ms Registers

Following is a list of number registers and their default values:

Item	Description
PS	Sets point size. Takes effect for paragraph. Default is 10.
VS	Sets vertical spacing. Takes effect for paragraph. Default is 12.
LL	Sets line length. Takes effect for paragraph. Default is 6i.
LT	Sets title length. Takes effect on next page. Defaults to the **LL** register value.
FL	Sets footnote length. Takes effect at next **.FS** request. Default is 5.5i.
PD	Sets paragraph distance. Takes effect for paragraph. Default is 1v (in **nroff**), .3v (in **troff**).
DD	Sets display distance. Takes effect for displays. Default is 1v (in **nroff**), .5v (in **troff**).
PI	Sets paragraph indent. Takes effect for paragraph. Default is 5n.
QI	Sets quotation indent. Takes effect at next **.QP** request. Default is 5n.
FI	Sets footnote indent. Takes effect at next **.FS** request. Default is 2n.
PO	Sets page offset. Takes effect on next page. Default is 0 (zero) (in **nroff**), 1i (in **troff**).
HM	Sets header margin. Takes effect on next page. Default is 1i.
FM	Sets footer margin. Takes effect on next page. Default is 1i.
FF	Sets footnote format. Takes effect at next **.FS** request. Default is 0 (zero) (1, 2, 3 available).

When resetting number register values, make sure to specify the appropriate units. Set the line length to 7i instead of just 7, which would result in output with one character per line. Setting the **FF** register to

1 (one) suppresses footnote superscripting. Setting it to 2 also suppresses indentation of the first line. Setting the **FF** register to 3 produces a footnote paragraph like the **.IP** request.

Following is a list of string registers available in the **ms** macros. These string registers can be used anywhere in the text.

Item	Description
***Q**	Open quotation marks (**”** in **nroff**; **`` `` ** in **troff**)
***U**	Close quotation marks (**“** in **nroff**; **'' ''** in**troff**)
***-**	Dash (**—** in **nroff**; **-** in **troff**)
***(MO**	Month of year
***(DY**	Day (current date)
******	Automatically numbered footnote
***'**	Acute accent (before letter)
***`**	Grave accent (before letter)
***^**	Circumflex accent (before letter)
***,**	Cedilla (before letter)
***:**	Umlaut (before letter)
***~**	Tilde (before letter).

When using the extended accent mark definitions available with the **.AM** request, these strings should come after, rather than before, the letter to be accented.

Note:

1. It is important to note that floating keeps and regular keeps are diverted to the same space, so they cannot be mixed.
2. The date format is restricted to U.S. English format.

mv Macro Package for the mvt and troff Commands

This package simplifies the typesetting of view graphs and projection slides in a variety of sizes. Although a few macros accomplish most of the formatting tasks needed in making transparencies, the entire facilities of the **troff**, **tbl**, **pic**, and **grap** commands are available for more difficult tasks.

The output can be previewed on most terminals, in particular the Tektronix 4014. For this device, specify the **-rX1** flag (which is automatically specified by the **mvt** command when that command is called with the **-D4014** flag). To preview output on other terminals, specify the **-a** flag.

The **mv** macros are summarized under the following headings:

- Foil-Start Macros
- Level Macros
- Text-Control Macros
- Default-Setting Macros.

Foil-Start Macros

For the following nine macros, the first character of the name (**V** or **S**) distinguishes between view graphs and slides, respectively, while the second character indicates whether the foil is square (**S**), small wide (**w**), small high (**h**), big wide (**W**), or big high (**H**). Slides are narrower than the corresponding view graphs. The ratio of the longer dimension to the shorter one is larger for slides than for view graphs. As a result, slide foils can be used for view graphs, but view graphs cannot be used for slide foils. On the other hand, view graphs can accommodate a bit more text.

Item	Description
.VS [*FoilNumber*] [*FoilID*] [*Date*]	Starts a square view graph. Foil size is to be 7 inches by 7 inches. The foil-start macro resets all variables (such as indent and point size) to initial default values, except for the values of the *FoilID* and *Date* variables inherited from a previous foil-start macro. The **.VS**macro also calls the **.A** macro.
.Vw, **.Vh**,**.VW**, **.VH**, **.Sw**, **.Sh**, **.SW** , **.SH**	Same as the **.VS** macro, except that these macros start view graphs (**V**) or slides (**S**) that are small wide (**w**), small high (**h**), large wide (**W**), or large high (**H**). The following macros are recommended: • **.VS** for square view graphs and slides • **.Sw** (and, if necessary, **.Sh**) for 35mm slides.
.Vw [*FoilNumber*] [*FoilID*] [*Date*]	Same as the **.VS** macro, except that foil size is 7 inches wide by 5 inches high.
.Vh [*FoilNumber*] [*FoilID*] [*Date*]	Same as the **.VS** macro, except that foil size is 5 inches wide by 7 inches high.
.VW [*FoilNumber*] [*FoilID*] [*Date*]	Same as the **.VS** macro, except that foil size is 7 inches wide by 5.4 inches high.
.VH [*FoilNumber*] [*FoilID*] [*Date*]	Same as the **.VS** macro, except that foil size is 7 inches wide by 9 inches high.
.Sw [*FoilNumber*] [*FoilID*] [*Date*]	Same as the **.VS** macro, except that foil size is 7 inches wide by 5 inches high.
.Sh [*FoilNumber*] [*FoilID*] [*Date*]	Same as the **.VS** macro, except that foil size is 5 inches wide by 7 inches high.
.SW [*FoilNumber*] [*FoilID*] [*Date*]	Same as the **.VS** macro, except that foil size is 7 inches wide by 5.4 inches high.
.SH [*FoilNumber*] [*FoilID*] [*Date*]	Same as the **.VS** macro, except that foil size is 7 inches wide by 9 inches high.

Note: The **.VW** and **.SW** foils are meant to be 9 inches wide by 7 inches high. However, because the typesetter paper is generally only 8 inches wide, **.VW** and **.SW** foils are printed 7 inches wide by 5.4 inches high and have to be enlarged by a factor of 9/7 before use as view graphs.

Level Macros

Item	Description
.A [*X*]	Places text that follows at the first indentation level (left margin). The presence of the *X* variable suppresses the half-line spacing from the preceding text.
.B [*Mark* [*Size*]]	Places text that follows at the second indentation level. Text is preceded by a specified mark (default is a large bullet). The *Size* variable is the increment or decrement to the point size of the mark with respect to the *prevailing* point size (default is 0). A value of 100 for the *Size* variable makes the point size of the mark equal to the default value of the *Mark* variable.
.C [*Mark* [*Size*]]	Same as the **.B** macro, but for the third indentation level. The default value of the *Mark* variable is an em dash.
.D [*Mark* [*Size*]]	Same as the **.B** macro, but for the fourth indentation level. The default value of the *Mark* variable is a small bullet.

Text-Control Macros

Item	Description
.I [+/-] [*Indentation*] [A[X]]	Changes the current text indent (does not affect titles). The specified indentation is in inches unless dimensioned. The default is 0. If the *Indentation* variable is signed, it is an increment or decrement. The presence of the *A* variable calls the **.A** macro and passes the *X* variable (if any) to it.
.S [*Size*] [*Length*]	Sets the point size and the line length. The value specified in the *Size* variable is the point size (default is previous). If the *Size* variable value is 100, the point size reverts to the *initial* default for the current foil-start macro. If the *Size* variable is signed, it is an increment or decrement (default is 18 for the **.VS**, **.VH**, and **.SH** macros, and 14 for the other foil-start macros). The *Length*variable specifies the line length (in inches unless dimensioned; the default is 4.2 inches for the **.Vh** macro, 3.8 inches for the **.Sh** macro, 5 inches for the **.SH** macro, and 6 inches for the other foil-start macros).
.T *String*	Prints the *String* variable value as a centered, enlarged title.
.U *String1*[*String2*]	Underlines the *String1* variable value and concatenates the *String2 variable* value (if any) to it. Using this operation is not recommended.

Default-Setting Macros

Item	Description
.DF [*Number Name*]**...**	Sets font positions. It cannot be displayed within foil input text; that is, it must follow the input text for a foil, but it must precede the next foil-start macro. The specified number is the position of the font specified by the *Name* variable. The **.DF** macro takes up to four pairs of *Number Name* variables, such as 1 H. The first *Name* variable specifies the prevailing font. For example: **.DF 1 H 2 I 3 B 4 S**.
.DV [*A*] [*B*] [*C*] [*D*]	Alters the vertical spacing between indentation levels. The value specified by the *A*, *B*, *C*, or *D* variable is the spacing for the **.A**, **.B**, **.C**, or **.D** macro, respectively. All non-null parameters must be dimensioned. Null parameters leave the corresponding spacing unaffected. The default setting is: **.DV .5v .5v .5v 0v**.

The **.S**, **.DF**, **.DV**, and **.U** macros do not cause a break. The **.I** macro causes a break only if it is called with more than one variable. All the other macros cause a break.

The **mv** macro package also recognizes the following uppercase synonyms for the following corresponding lowercase **troff** command requests:

- **.AD**
- **.BR**
- **.CE**
- **.FI**
- **.HY**
- **.NA**
- **.NF**
- **.NH**
- **.NX**
- **.SO**
- **.SP**
- **.TA**
- **.TI**

The **Tm** string produces the trademark symbol.

Environment Variable

Item	Description
LANG	Determines the locale's equivalent of y for yes or no queries. The allowed affirmative responses are defined in the locale variable **YESSTR**. If **LANG** is not set, or if it is set to an empty string, the **YESSTR** from the default C locale is used.

nroff and troff Requests for the nroff and troff Commands

The following **nroff** and **troff** requests are included in a specified working file or in standard input. The **nroff** and **troff** requests control the characteristics of the formatted output when the file or standard input is processed with the **nroff** or **troff** commands. The **nroff** and **troff** requests are grouped by function, in the following sections:

- Numerical Parameter Input
- Font and Character Size Control
- Page Control
- Text Filling, Adjusting, and Centering
- Vertical Spacing
- Line Length and Indenting
- Macros, Strings, Diversions, and Position Traps
- Number Registers
- Tabs, Leaders, and Fields
- Input and Output Conventions and Character Translations
- Hyphenation
- Three-Part Titles
- Output Line Numbering
- Conditional Acceptance of Input
- Environment Switching
- Insertions from Standard Input
- Input and Output File Switching
- Miscellaneous

For number variables written as *+Number*, the variable can be expressed as follows:

- The *Number* variable by itself is an absolute value.
- The *+Number* variable increases the currently set value.
- The *-Number* variable decreases the variable relative to its current value.

Note: For all numeric parameters, numbers are expressed using ASCII Arabic numerals only.

The notes at the end of this command are referenced in the specific requests where applicable.

Numerical Parameter Input

Both **nroff** and **troff** requests accept numerical input with the appended scale indicators shown in the following table, where *S* is the current type size in points, *V* is the current vertical line spacing in basic units, and *C* is a nominal character width in basic units.

Indicator	Meaning	Number of Basic nroff Units
`i`	Inch (machine-dependent for troff)	240

Indicator	Meaning	Number of Basic nroff Units
c	Centimeter	240x50/127
P	Pica = 1/6 inch	240/6
m	Em = S points	*C*
n	En = Em/2	*C (same as Em)*
p	Point = 1/72 inch	240/72
u	Basic unit	1
v	Vertical line space	*V*
k	Width single-width kana	*C*
K	Width double-width kanji	Two *C*s
none	Default	

Note:

1. If a non-kanji output device is selected, an en-width is used instead.
2. If a non-kanji output device is selected, an em-width is used instead.

In the **nroff** request, both the em and the en are taken to be equal to the *C*, which is output-device dependent; frequent values are 1/10 and 1/12 inch. Actual character widths in the **nroff** request need not be all the same, and characters constructed with predefined strings such as **- >** are often extra wide.

Japanese Language Support: In the output from the **nroff** command, all double-width Japanese characters such as all kanji and some katakana characters have a fixed width equal to two *C*s. All single-width Japanese characters such as some katakana characters have a fixed width equal to *C*.

The scaling for horizontally-oriented control characters, vertically-oriented control characters, and the requests **.nr**, **.if**, and **.ie** are as follows:

Orientation	Default Measure	Request or Function
Horizontal	Em (**m**)	**.ll, .in, .ta, .lt, .po, .mc, \h, \l**
Vertical	Vertical line space (**v**)	**.pl, .wh, .ch, .dt, .sp, .sv, .ne, .rt, \v \x, \L**
Register-oriented or Conditional	Basic unit (**u**)	**.nr, .if. .ie**
Miscellaneous	Point (**p**)	**.ps, .vs, \H, \s**

All other requests ignore scale indicators. When a number register containing an already appropriately scaled number is interpreted to provide numerical input, the unit scale indicator **u** might need to be appended to prevent an additional inappropriate default scaling. The *Number* might be specified in decimal-fraction form, but the parameter that is finally stored is rounded to an integer number of basic units.

Font and Character Size Control

Item	Description
.bd *Font Number*	Makes the characters in the specified font artificially bold by overstriking them the specified number of times when using **nroff**, or by printing each character twice separated by *Number* -1 basic units when using **troff**. If the *Number* variable is not specified, the bold mode is turned off. The *Font* value must be an ASCII font name or font position. For the **nroff** command, the default setting of the **.bd** request is 3 3, specifying that characters on the font mounted at position 3 (usually bold) are to be overstruck 3 times (that is, printed in place a total of 4 times). The font name itself can be substituted for the font position; for example, **.bd I 3**. The *Number* variable is functionally identical to the **-u** flag of the **nroff** command. (The bold mode must be in effect when the characters are physically printed.) This request can affect the contents of the **.b** general-number register. The bold mode still must be in effect, or restarted at the time of physical output. You cannot turn off the bold mode in the **nroff** command if it is being controlled locally by the printing device as with, for example, a DASI 300. Initial Value: Off If No Value Specified: **-**
.bd S *Font Number*	Makes the characters in the special font bold whenever the specified font is the current font. The mode must be in effect when the characters are physically printed. The *Font* value must be an ASCII font name or font position. The mode still must be in effect, or again so, at the time of physical output. Initial Value: Off If No Value Specified: **-**
.cs *Font Number M*	Sets constant character space (width) mode to the *Font* variable value (if mounted). The width of every character is taken to be the value specified in the *Number* variable divided by 36 ems. If the *M* variable is not specified, the em width is that of the character's point size; if the *M* variable is given, the width is the value specified by the *M* variable minus points. All affected characters are centered in this space, including those with an actual width larger than this space. Special font characters occurring while the specified font is the current font are also so treated. The *Font* value must be an ASCII font name or font position. If the *Number* variable is absent, the mode is turned off. The mode must be in effect when the characters are physically printed. This request is ignored by the **nroff** command. Relevant values are part of the current environment. The mode still must be in effect, or again so, at the time of physical output. Initial Value: Off If No Value Specified: **-**

Item	Description
.fp *Font Number*[*File*]	Specifies the font position. This is a statement that the specified font is mounted on the position specified by the *Number* variable. The *Font* variable must be a one- or two-character ASCII font name.  **Attention:** It is an irrecoverable error if the *Font* variable is not specified. The **.fp** request accepts a third optional variable, the *File* variable, which is the actual path name of the file containing the specified font. The *File* variable value can be any legal file name and can contain extended characters. Japanese Language Support**:** The *File*value can be any legal file name. Values are typesetter- or printer-dependent. Initial Value: **-** If No Value Specified: Ignored
.ft *Font*	Changes the font style to the specified font, or if *Font* value is numeric, to the font mounted on that position. Alternatively, embed **\f***Font* command. The font name **P** is reserved to mean the previous font. The *Font* variable value must be an ASCII font name or font position. If using a font name consisting of two characters, use the alternative form of **.ft**, **\f**. Relevant values are part of the current environment. Values are typesetter or printer-dependent. Initial Value: Roman If No Value Specified: Previous
.ps [+/-][*Number*]	Sets the point size to that specified by the +/-*Number* variable. Although any positive size value can be requested, an invalid size results in the nearest valid size being used. Size 0 refers to the previous size. Alternatively, **\s***Number* or **\s**+/-*Number*; if the *Number* value is two digits, use **\s(***Number* or **\s**+/-**(***Number*. For compatibility with older versions of the **troff** command, the form is valid for two-digit values of *n* = **10**, **11**, **12**, **14**, **16**, **18**, **20**, **22**, **24**, **28**, and **36**. This request is ignored by the **nroff** command. Relevant values are part of the current environment. Initial Value: 10 point If No Value Specified: Previous
.ss *Number*	Sets space-character size to the specified number divided by 36 ems. This size is the minimum word spacing in adjusted text. This request is ignored by the **nroff** command. Relevant values are part of the current environment. Initial Value: 12/36 em If No Value Specified: Ignored

Page Control

Item	Description
.bp [+/-][*Number*]	Specifies a break page. The current page is ejected and a new page is begun. If the +/-*Number* variable is specified, its value becomes the new page number. Also refer to the **.ns** request. This request usually causes a line break similar to the **.br** request. Calling this request with the control character ” **'** “ (instead of ”**.**“) suppresses that break function. Initial Value: *Number*=**1** If No Value Specified: **-**
.mk *Register*	Marks the current vertical place (or a place in the current diversion) in an internal register (associated with the current diversion level) or in the specified register, if given. The *Register* variable is the ASCII name of a number register. Mode or relevant values are associated with the current diversion level. For more information, refer to the **.rt** request. Initial Value: None If No Value Specified: Internal
.ne *Number D*	Indicates a need for the specified vertical space. If the page space needed (*Number*) is greater than the distance to the next trap (*D*), a forward vertical space of size *D* occurs, which springs the trap. If there are no remaining traps on the page, the size specified by the *D* variable is the distance to the bottom of the page. If the distance to the next trap (*D*) is less than one vertical line space (**v**), another line could still be output before the trap is sprung. In a diversion, the size specified by *D* is the distance to the diversion trap, if any, or is very large. The value of *D* is also usually contained in the **.t** *Number* register. Mode or relevant values are associated with the current diversion level. Initial Value: *Number*=**1***V* If No Value Specified: **-**
.pl [+/-][*Number*]	Sets page length to the +/-*Number* variable value. The internal limitation is approximately 136 inches in the **nroff** command, but varies with the device type in the **troff** command. A good working maximum for the **troff** command is 75 inches. The current page length is available in the **.p** register. Initial Value: 11 inches If No Value Specified: 11 inches
.pn [+/-][*Number*]	Specifies that the next page (when it occurs) has the page number specified by the +/-*Number* variable. A **.pn** request must occur either before text is initially printed or before a break occurs to affect the page number of the first page. The current page number is in the **%** register. Initial Value: *Number*=**1** If No Value Specified: Ignored
.po [+/-][*Number*]	Specifies a page offset. The current left margin is set to the +/-*Number*variable value. The initial **troff** command value provides 1 inch of left margin. For more information, refer to ”Line Length and Indenting“. The current page offset is available in the **.o** register. Initial Value: **0** for the **nroff** command; **1** for the **troff** command. If No Value Specified: Previous

Item	Description
.rt [+/-][*Number*]	Returns upward only to a marked vertical place in the current diversion. If the +/-*Number* variable value (relative to the current place) is given, the place is the value specified by the +/-*Number* variable from the top of the page or diversion. If the *Number* variable is not specified, the place is marked by a previous **.mk** request. Mode or relevant values are associated with the current diversion level. The **.sp** request can be used in all cases, instead of the **.rt** request, by spacing to the absolute place stored in an explicit register as, for example, when using the sequence **.mk** *Register* . . . **.sp**|**n**R**u**. Initial Value: None If No Value Specified: Internal

Text Filling, Adjusting, and Centering

Item	Description
.ad *Indicator*	Begins line adjustment. If the fill mode is not on, adjustment is deferred until the fill mode is back on. If the *Indicator* variable is present, the adjustment type is changed as shown in the following list: **Indicator** **Adjustment Type** ***l*** Adjust left margin only. ***r*** Adjust right margin only. ***c*** Center. ***b* or *n*** Adjust both margins. **blank** Unchanged. The adjustment indicator can also be a number obtained from the **.j** register.

Japanese Language Support:

Indicator	Adjustment Type
k	Turn on kinsoku shori processing (turned off with **.ad n**, **.ad b**, or **.ad l**). Usually, lines of Japanese text are filled to the margins without regard for the characters beginning or ending lines. When kinsoku shori processing is enabled, lines are prevented from ending with an open bracket character or from beginning with a close bracket or punctuation character. If a line ends with an open bracket, the line is left short and the bracket begins the next line. If a line begins with a close bracket or punctuation character, the preceding line is extended and the character ends the preceding line. Requesting Japanese kinsoku shori processing on an output device that does not support kanji characters has no effect. Relevant values are part of the current environment. Initial Value: Adjust, both If No Value Specified: Adjust

Item	Description
.br	Specifies a break. The filling of the line currently being collected is stopped and the line is output without adjustment. Text lines beginning with space characters and empty text lines (blank lines) also cause a break. Initial Value: **-** If No Value Specified: **-**
.ce [*Number*]	Centers the next specified number of input text lines within the current line length, minus indent. If the *Number* variable equals 0, any residual count is cleared. A break occurs after each of the *Number* variable input lines. If the input line is too long, it is left adjusted. Relevant values are part of the current environment. This request usually causes a line break similar to the **.br** request. Calling this request with the control character ” **'** “ (instead of ”**.**“) suppresses that break function. Initial Value: Off If No Value Specified: *Number*=**1**
.fi	Fills subsequent output lines. The **.u** register has a value of 1 (one) in fill mode and a value of 0 (zero) in no-fill mode. Relevant values are part of the current environment. This request usually causes a line break similar to the **.br** request. Calling this request with the control character ” **'** “ (instead of ”**.**“) suppresses that break function. Initial Value: Fill If No Value Specified: **-**
.na	Specifies no-adjust mode. Adjustment is turned off; the right margin is ragged. The adjustment type for the **.ad** request is not changed. Output-line filling still occurs if the fill mode is on. Relevant values are part of the current environment. Initial Value: None If No Value Specified: **-**
.nf	Specifies no-fill mode. Subsequent output lines are neither filled nor adjusted. Input-text lines are copied directly to output lines without regard for the current line length. Relevant values are part of the current environment. This request usually causes a line break similar to the **.br** request. Calling this request with the control character ” ' “ (instead of ”**.**“) suppresses that break function. Initial Value: Fill If No Value Specified: **-**

Vertical Spacing

Item	Description
Blank text line	Causes a break and outputs a blank line exactly like an **.sp 1** request.
.ls *Number*	Sets line spacing to the value specified by the +/-*Number* variable. The *Number* **-1** *V***s** (blank lines) variable values are appended to each output-text line. Appended blank lines are omitted if the text or previous appended blank line reached a trap position. Relevant values are part of the current environment. Initial Value: **1** If No Value Specified: Previous

Item	Description
.ns	Turns on no-space mode. When on, the no-space mode inhibits **.sp** and **.bp** requests without a next page number. The no-space mode is turned off when a line of output occurs or with the **.rs** request. This request usually causes a break. Initial Value: Space If No Value Specified: **-**
.os	Outputs saved vertical space. The no-space mode has no effect. Used to output a block of vertical space requested by the previous **.sv** request. Initial Value: **-** If No Value Specified: **-**
.rs	Restores spacing. The no-space mode is turned off. This request usually causes a break. Initial Value: None If No Value Specified: **-**
.sp *Number*	Spaces vertically in either direction. If the *Number* variable value is negative, the motion is backward (upward) and is limited to the distance to the top of the page. Forward (downward) motion is truncated to the distance to the nearest trap. If the no-space mode is on, no spacing occurs. Refer to the **.ns** and **.rs** requests. This request usually causes a line break similar to the **.br** request. Calling this request with the control character "'" (instead of ".") suppresses that break function. Initial Value: **-** If No Value Specified: **1***V*
.sv *Number*	Saves a contiguous vertical block of the specified size. If the distance to the next trap is greater than the *Number* variable value, the specified vertical space is output. The no-space mode has no effect. If this distance is less than the specified vertical space, no vertical space is immediately output, but is remembered for later output (refer to the **.os** request). Subsequent **.sv** requests overwrite any still-remembered *Number* variable value. Initial Value: **-** If No Value Specified: *Number*=**1***V*
.vs *Number*	Sets vertical base-line spacing size *V* to the *Number* variable. Transient extra vertical space can be specified by **\x** *N*. Relevant values are part of the current environment. Initial Value: The *Number* variable equals 1/16 inch for the **nroff** command and 12 points for the **troff** command. If No Value Specified: Previous

Line Length and Indenting

Item	Description
.in [+/-]*Number*	Sets indent to the +/-*Number* variable value. The indent is prepended to each output line. Relevant values are part of the current environment. This request usually causes a line break similar to the **.br** request. Calling this request with the control character " **'** " (instead of "**.**") suppresses that break function. Initial Value: *Number*=**0** If No Value Specified: Previous

Item	Description
.ll [+/-]*Number*	Sets line length to the +/-*Number* variable value. In the **troff** command, the maximum line length plus page offset is device-dependent. Relevant values are part of the current environment. Initial Value: 6.5 inches If No Value Specified: Previous
.ti [+/-]*Number*	Specifies a temporary indent. The next output text line is indented a distance of the value specified by the +/-*Number* variable with respect to the current indent. A negative value for the *Number* variable can result in spacing backward over the current indent, so that the resulting total indent can be a value of 0 (zero) (equal to current page offset), but cannot be less than the current page offset. The temporary indent applies only for the one output line following the request; the value of the current indent, which is stored in the **.i** register, is not changed. Relevant values are part of the current environment. This request usually causes a line break similar to the **.br** request. Calling this request with the control character ” **'** “ (instead of ”**.**“) suppresses that break function. Initial Value: **-** If No Value Specified: Ignored

Macros, Strings, Diversions, and Position Traps

Item	Description
.am *Macro1* [*Macro2*]	Appends to *Macro 1*; appends version of the **.de** request. Both the *Macro1* and *Macro2* variables must be either one or two ASCII characters. *Macro2* is a termination sequence to end the diversion. Initial Value: **-** If No Value Specified: **.***Macro2***=..**
.as *StringName String*	Appends the specified string to the value specified by the *StringName* variable; appended version of the **.ds** request. The *StringName* variable value must be one or two ASCII characters. Initial Value: **-** If No Value Specified: Ignored
.ch *Macro* [*Number*]	Changes the trap position for the specified macro to the value specified by the *Number* variable. In the absence of the *Number* variable, the trap, if any, is removed. The *Macro* variable value must be one or two ASCII characters. Initial Value: **-** If No Value Specified: **-**
.da [*Macro*]	Diverts, appending to the specified macro and appends version of the **.di** request. The *Macro* variable must be one or two ASCII characters. Mode or relevant values are associated with the current diversion level. Initial Value: - If No Value Specified: End current diversion

Item	Description
.de *Macro1* [*Macro2*]	Defines or redefines the value specified by the *Macro1* variable. The contents of the macro begins on the next input line. Input lines are copied in copy mode until the definition is stopped by a line beginning with *.Macro2*. In the absence of the *Macro2* variable, the definition is stopped by a line beginning with "**..**". A macro can contain **.de** requests, provided the stopping macros differ or the contained definition terminator is concealed. The "**..**" can be concealed as "**** . ", which copies as "**\...**" and is reread as "**..**". The *Macro1* and *Macro2*variables must each be one or two ASCII characters. Initial Value: **-** If No Value Specified: **.***Macro2***=..**
.di [*Macro*]	Diverts output to the specified macro. Normal text processing occurs during diversion except that page offsetting is not performed. The diversion ends when the **.di** or **.da** request is encountered without a variable. Extraneous requests of this type should not be displayed when nested diversions are being used. The *Macro* variable must be one or two ASCII characters. Mode or relevant values are associated with the current diversion level. Initial Value: **-** If No Value Specified: End
.ds *StringName String*	Defines a string specified by the *StringName* variable to contain the value specified by the *String* variable. Any initial double-quote in *String* is stripped off to permit initial blanks. The *StringName* variable must be one or two ASCII characters.

Item	Description
.ds *StringName* ^A <*SetNumber*> <*MessageNumber*> [^A"<*DefaultMessage*> "] [^A<*Argument*> ^B<*Argument*> ^B <*Argument*>...]	Provides an alternate **.ds** syntax that allows the use of a message catalog for language-independent string definitions. Based on the message *SetNumber* and the *MessageNumber* within the locale-specific catalog, the message catalog is read in copy mode and the corresponding message is placed into the *StringName* variable. The initial sequence specifying the message set and message number can be omitted for backward compatibility. The ASCII code Control-A (**^A**) delimits message identification, default message and optional argument list. The ASCII code Control-B (**^B**) delimits an individual optional argument list. In the following example, `.ds {c ^A2 41^A"ERROR: (%1$s) input line \` `%2$s" ^A\n(.F^B\n(.c` 2 is the message set number. 41 is the message number. text within quotes (`"..."`) is the default message. `\n(.F` is the name of the current input file. `\n(.c` is the number of lines read from the input file. If you assume the **troff** command runs with these conditions: • The message at set 2 and number 41 matches the default message • The current input file is `paper.doc` • The **.ds** directive is on line 124 in the input file. then the string `{c` would be defined as: `ERROR: (paper.doc)input line 123` Other examples are: `.ds {c ^A2 41` `/* Without optional default message */` `.ds {c ^A2 41^A"ERROR: (%1$s) input file \` `%2$s" /* Without optional arguments */`

Item	Description
	If both the set number and the message number are set to zero, then the current date is returned in the current local's format. A user defined date format string can be defined in the default message field. The user defined format string must conform to the conversion specifications outlined by the **strftime** function in *Technical Reference: Base Operating System and Extensions*. In the following examples: `.ds DT^A0 0` If the current date were July 10, 1991, in an English U.S. locale, DT would be defined as 7/10/91. `.ds DT^A0 0^A"Today is %B %d, %Y"` If the current date were July 10, 1991, in an English U.S. locale, DT would be defined as `Today is July 10, 1991`. The second syntax method is not intended for general use. It is used in the **nroff** and **troff** macro files supplied with the system to facilitate internationalization of internally generated messages. Initial Value: **-** If No Value Specified: Ignored
.dt *Number Macro*	Installs a diversion trap at the position specified by the *Number* variable in the current diversion to start the specified macro. Another **.dt** request redefines the diversion trap. If no variables are given, the diversion trap is removed. The *Macro* variable must be one or two ASCII characters. Mode or relevant values are associated with the current diversion level. Initial Value: **-** If No Value Specified: Off
.em *Macro*	Calls the specified macro when all input has ended. The effect is the same as if the contents of the specified macro had been at the end of the last file processed. The specified macro must be one or two ASCII characters. Initial Value: None If No Value Specified: None
.it *Number Macro*	Sets an input-line-count trap to call the specified macro after the number of lines of text input specified by the *Number* variable have been read (control or request lines are not counted). The text can be inline text or text provided by macros called explicitly (through inline calls) or implicitly (through traps). The *Macro* variable must be one or two ASCII characters. Relevant values are part of the current environment. Initial Value: **-** If No Value Specified: Off
.rm *Name*	Removes the specified request, macro, or string. The *Name* variable value is removed from the name list and any related storage space is freed. Subsequent references have no effect. The *Name* variable must be one or two ASCII characters. Initial Value: **-** If No Value Specified: Ignored

Item	Description
.rn *Name1 Name2*	Renames the request, macro, or string value specified by the *Name1*variable to the value specified by the *Name2* variable. The *Name1*and *Name2* variable values must each be one or two ASCII characters. Initial Value: Ignored If No Value Specified: **-**
.wh *Number Macro*	Installs a trap to call the specified macro at the page position specified by the *Number* variable. A negative *Number* variable value is interpreted with respect to the page bottom. Any macro previously planted at the page position specified by the *Number* variable is replaced by the *Macro* variable value. A *Number* variable value of 0 refers to the top of a page. In the absence of the *Macro* variable, the first trap found at the page position specified by the *Number* variable, if any, is removed. The *Macro* variable must be one or two ASCII characters. Initial Value: **-** If No Value Specified: **-**

Number Registers

Item	Description
.af *Register Indicator*	Assigns the format as specified by the *Indicator* variable to the specified register. The *Register* variable must be one or two ASCII characters. The available format *Indicator* variable values are as follows: **Indicator** Numbering Sequence **1** 0,1,2,3,4,5, . . . **001** 000,001,002,003,004,005, . . . **i** 0,i,ii,iii,iv,v, . . . **I** 0,I,II,III,IV,V, . . . **a** 0,a,b,c, . . . ,z,aa,ab, . . . ,zz,aaa, . . . **A** 0,A,B,C, . . . ,Z,AA,AB, . . . ,ZZ,AAA, . . . An Arabic format indicator having *N* digits (for example, 000000001) indicates a field width of *N* digits. The read-only registers and the width function are always Arabic. Japanese Language Support**:** The following value specifies the character width for formatting Japanese numeric output in kanji: **k** The number is formatted as a kanji string. If this is requested when a non-kanji codeset is specified, a warning message is printed and the **1** format is used. Initial Value: Arabic If No Value Specified: **-**

Item	Description
.nr *Register* +/-*Number1 Number2*	Assigns the specified register the value specified by the +/-*Number* variable with respect to the previous value, if any. The increment for auto-incrementing is set to the *Number2* variable value. The *Register* variable must be one or two ASCII characters. Initial Value: **-** If No Value Specified: **-**
.rr *Register*	Removes the specified register. If many registers are being created dynamically, it can become necessary to remove registers that are not needed to recapture internal storage space for new registers. The *Register* variable must be one or two ASCII characters. Initial Value: **-** If No Value Specified: **-**

Tabs, Leaders, and Fields

Item	Description
.fc *Delimiter Indicator*	Sets the field delimiter to the specified delimiter; the padding indicator is set to the space character or to the specified indicator. In the absence of variables, the field mechanism is turned off. The *Delimiter* variable value and the *Indicator* variable value must be ASCII characters. Initial Value: Off If No Value Specified: Off
.lc *Character*	Sets the leader repetition character to the specified character, or removes specifying motion. The *Character* variable value must be an ASCII character. Relevant values are part of the current environment. Initial Value: **.** If No Value Specified: None
.ta *Stop* [*Type*]...	Sets tab stops. Default tab stops are set at every eight characters for the **nroff** command and every half inch for the **troff** command. Multiple *StopType* pairs can be specified by separating them with spaces; a value preceded by **+** (plus sign) is treated as an increment to the previous stop value. The specified type determines how the text is adjusted at the tab stops. The *Type* variable values are as follows: **Type** **Adjustment** **R** Right-adjusting **C** Centering **blank** Left-adjusting Relevant values are part of the current environment. Initial Value: 8 ens for the **nroff** command and 0.5 inch for the **troff** command If No Value Specified: None

Item	Description
.tc *Character*	Sets the tab repetition character to the specified character, or removes specifying motion. The *Character* variable value must be an ASCII character. Relevant values are part of the current environment. Initial Value: None If No Value Specified: None

Input/Output Conventions and Character Translations

Item	Description
.cc *Character*	Sets the basic control character to the specified character, or resets to ”**.**“. The *Character* variable value must be an ASCII character. Relevant values are part of the current environment. Initial Value: **.** If No Value Specified: **.**
.cu [*Number*]	A variant of the **.ul** request that causes every character to be underlined and causes no line breaks to occur in the affected input lines. That is, each output space following a **.cu** request is similar to an unpaddable space. The **.cu** request is identical to the **.ul** request in the **troff** command. Relevant values are part of the current environment. Initial Value: Off If No Value Specified: *Number*=**1**
.c2 *Character*	Sets the no-break control character to the specified character or resets to ” **'** “. The *Character* variable value must be an ASCII character. Relevant values are part of the current environment. Initial Value: **'** If No Value Specified: **'**
.ec *Character*	Sets the escape character to **** (backslash) or to the value specified by the *Character* variable, if given. The *Character* variable value must be an ASCII character. Initial Value: **** If No Value Specified: ****
.eo	Turns off the escape mechanism. Initial Value: On If No Value Specified: **-**
.lg [*Number*]	Turns on the ligature mode if the *Number* variable value is absent or nonzero; turns off ligature mode if the *Number* variable value is 0. If the *Number* variable value is 2, only the two-character ligatures are automatically called. The ligature mode is inhibited for request, macro, string, register, or file names, and in the copy mode. This request has no effect in the **nroff** command. Initial Value: On, for the **troff** command If No Value Specified: On

Item	Description
.tr *Character1 Character2 Character3 Character4*	Translates, among other things, the character value specified by the *Character1* variable into the *Character2* variable value, the character value specified by the *Character3* variable into the *Character4*variable value. If an odd number of characters is given, the last one is mapped into the space character. To be consistent, a particular translation must stay in effect from input to output time. All specified characters must be ASCII characters. To reset the **.tr** request, follow the request with previous variables given in duplicate. For example, the following **.tr** request: `.tr aAbBc<C,>` can be reset by entering: `.tr aabbcc` It must stay in effect until logical output. Initial Value: None If No Value Specified: **-**
.ul [*Number*]	Underlines in the **nroff** command (or italicizes in the **troff**command) the number of input-text lines specified by the *Number*variable. Actually switches to underline font, saving the current font for later restoration. Other font changes within the span of a **.ul** request take effect, but the restoration undoes the last change. Output generated by the **.tl** request is affected by the font change, but does not decrement the *Number* variable value. For more information, refer to the section "Three-Part Titles". If the specified number is greater than 1, there is the risk that a trap-called macro can provide text lines within the span; environment switching can prevent this. Relevant values are part of the current environment. Initial Value: Off If No Value Specified: *Number*=**1**
.uf *Font*	Underlines the font set to the value specified by the *Font* variable. In the **nroff** command, the *Font* variable cannot be on position 1 (initially Times Roman). The *Font* variable value must be an ASCII font name. Initial Value: Italic If No Value Specified: Italic

Hyphenation

Item	Description
.hc *Character*	Sets the hyphenation indicator character to the value specified by the *Character* variable or to the default. The indicator is not displayed in the output. The *Character* variable value must be an ASCII character. Relevant values are part of the current environment. Initial Value: **\%** If No Value Specified: **\%**

Item	Description
.hw *Word1...*	Specifies hyphenation points in words with embedded minus signs. Versions of a word with a terminal s are implied; that is, *dig-it* implies *dig-its*. This list is examined initially and after each suffix stripping. The space available is 1024 characters, or about 50 to 100 words. Initial Value: If No Value Specified: Ignored
.hy *Number*	Turns on automatic hyphenation if the specified number is equal to or greater than 1; turns it off if the specified number is equal to 0 (equal to the **.nh** request). If the specified number is 2, the last lines (ones that cause a trap) are not hyphenated. If the specified number is 4 or 8, the last or first two characters, respectively, of a word are not split off. These values are additive; for example, a value of 14 calls all three restrictions (number equal to 2, number equal to 4, and number equal to 8). Relevant values are part of the current environment. Initial Value: No hyphenation If No Value Specified: Hyphenate
.nh	Turns off automatic hyphenation. Relevant values are part of the current environment. Initial Value: No hyphenation If No Value Specified: **-**

Three-Part Titles

Item	Description
.lt [+/-][*Number*]	Sets the length of title value specified by the +/-*Number* variable. The line length and the title length are independent. Indents do not apply to titles, although page offsets do. Relevant values are part of the current environment. Initial Value: 6.5 inches If No Value Specified: Previous
.pc *Character*	Sets the page number character to the specified character or removes it. The page-number register remains %. The *Character* variable value must be an ASCII character. Initial Value: **%** If No Value Specified: Off
.tl *'Left'Center'Right'*	The strings represented by the *Left*, *Center*, and *Right* variables, respectively, are left-adjusted, centered, and right-adjusted in the current title length. Any of the strings can be empty, and overlapping is permitted. If the page-number character (initially **%**) is found within any of the fields, it is replaced by the current page number having the format assigned to the **%** register. Any ASCII character that is not displayed in the strings can be used as the string delimiter. Initial Value: **-** If No Value Specified: **-**

Output-Line Numbering

Item	Description
.nm [+/-] [*Number*] [*M*] [*S*] [*I*]	Turns on line-number mode. If the *M* variable is specified, only those line numbers that are multiples of the *M* variable value are to be printed. Every line number is printed if the *M* variable is absent (default is *M*=**1**). When line-number mode is in effect, a three-digit Arabic number plus a digit space are prepended to output text lines. The text lines are thus offset by four digit spaces, but otherwise retain their line length. If the *S* variable is given, it specifies the number of digit spaces to be displayed between the line number and the text (default is *S*=**1**). If the *I* variable is given, it specifies the number of digit spaces to indent before the line number (default is *I*=**0**). Relevant values are part of the current environment. Initial Value: **-** If No Value Specified: Off
.nn *Number*	Suspends line numbering. The specified number of lines are not numbered. Relevant values are part of the current environment. Initial Value: **-** If No Value Specified: *Number*=**1**

Conditional Acceptance of Input

The *Condition* variable specifies one of the following one-character names:

Item	Description
o	If the current page number is odd.
e	If the current page number is even.
t	If the formatter is the **troff** command.
n	If the formatter is the **nroff** command.
.if *Condition Anything*	If the value specified by the *Condition* variable is true, accepts the value specified by the *Anything* variable as input; in multiline case, uses **\{***Anything***\}**.
.if *!Condition Anything*	If the value specified by the *Condition* variable is false, accepts the value specified by the *Anything* variable as input.
.if *Number Anything*	If the expression states that the *Number* variable value is greater than 0, accept the value specified by the *Anything* variable as input.
.if *!Number Anything*	If the expression states that the *Number* variable value is less than or equal to 0, accepts the value specified by the *Anything* variable as input.
.if *'String1'String2' Anything*	If the *String1* variable value is identical to the *String2* variable value, accepts the value specified by the *Anything* variable as input. Any nonblank ASCII character not in the *String1* and *String2* variables can be used as the delimiter.
.if *!'String1'String2' Anything*	If the *String1* variable value is not identical to the *String2*variable value, accepts the value specified by the *Anything* variable as input. Any nonblank ASCII character not in the *String1* and *String2* variables can be used as the delimiter.
.el *Anything*	Specifies the **else** portion of an **if/else** conditional.

Item	Description
.ie *Condition Anything*	Specifies the **if** portion of an **if/else** conditional dependent on the value of the *Condition* variable. Can be used with any of the preceding forms of the **.if** request.

Environment Switching

Item	Description
.ev *Environment*	Switches to the specified environment. The value specified by the *Environment* variable must be 0, 1, or 2. Switching is done in push-down fashion so that restoring a previous environment must be performed with the **.ev** request rather than with a specific reference. Initial Value: *Environment*=0 If No Value Specified: Previous

Insertions from Standard Input

Item	Description
.ex	Exits from the **nroff** command or **troff** command. Text processing is stopped exactly as if all input had ended. Initial Value: **-** If No Value Specified: **-**
.rd *Prompt*	Reads insertion from standard input until two newline characters in a row are found. If the standard input is the user's keyboard, the specified prompt (or the ASCII BEL character) is written onto the user's terminal. The **.rd** request behaves like a macro, and additional variables can be placed after the *Prompt* variable. Initial Value: **-** If No Value Specified: *Prompt*=the ASCII BEL character

Input and Output File Switching

Item	Description
.cf *File*	Copies the contents of the specified file, uninterrupted, into the **troff** command output file at this point. Problems occur unless the motions in the file restore the current horizontal and vertical position. Initial Value: **-** If No Value Specified: **-**
.lf *Number File*	Corrects the **troff** command interpretation of the current line number (as specified by the *Number* variable) and the current file (as specified by the *File* variable) for use in error messages. Initial Value: **-** If No Value Specified: **-**
.nx *File*	Uses the specified file as the input file. The current file is considered ended and the input is immediately switched to the specified file. Initial Value: **-** If No Value Specified: End of file

Item	Description
.pi *Program*	Pipes output to the specified program. This request must occur before any printing occurs. No variables are transmitted to the specified program. Initial Value: **-** If No Value Specified: **-**
.so *File*	Switches the source file. The top input (file-reading) level is switched to the specified file. When this file ends, input is again taken from the original file. The **.so** request can be nested. When a **.so** request is encountered, the processing of the specified file is immediate. Processing of the original file (for example, a macro that is still active) is suspended. A file should be preprocessed, if necessary, before being called by the **.so** request. The **eqn**, **tbl**, **pic**, and **grap** commands do not reach through a **.so** request to process an object file. Initial Value: **-** If No Value Specified: **-**

Miscellaneous

Item	Description
.ab *Text*	Prints the value specified by the *Text* variable to the diagnostic output (usually the terminal) and ends without further processing. If text is missing, the message `User Abort` is printed and the output buffer is flushed. This request is used in interactive debugging to force output.

Item	Description
.ab ^A<*SetNumber*> <*MessageNumber*> [^A"<*Default*> "] [^A<*Argument*> ^B<*Argument*> ^B<*Argument*>...]	Provides alternate syntax to allow use of a message catalog for language-independent abort messages. Prints the appropriate message specified by the parameter on the diagnostic output (usually the terminal) and ends without further processing. If there are no parameters, the message catalog equivalent to the following: `troff: User Abort, line no. file filename` is output. The output buffer is flushed. This request is used in interactive debugging to force output. Based on the message *SetNumber* and the *MessageNumber* variables within the locale-specific catalog, the message catalog is read in copy mode and the corresponding message is written to the user's terminal. The initial sequence specifying the message set and message number can be omitted for backward compatibility. The ASCII code Control-A (^A) delimits message identification, default message, and optional argument list. The ASCII code Control-B (^B) delimits individual optional argument list. In the following example: `.ab ^A2 42^A"Processing has been terminated \` `at line %1$s."^A\n(c.` 2 is the message set number. 42 is the message number. Text within quotes **"...“** is the default message. `\n(c.` is the number of lines read from the input file. If you assume the **troff** command runs with the following conditions: • The message at set 2 and number 42matches the default message. • The **.ab** directive is on line 124in the input file. then the following would be displayed on the user's terminal: `Processing has been terminated at line 123.` Initial Value: **-** If No Value Specified: User cancel

Item	Description
.Dt *Parameter*	Defines the format for returning the date within the **nroff** or **troff** request. By default, without the optional *Parameter*, the locale-specific date format specified by the current locale setting for the **LC_TIME** category is used. This corresponds to the "%x" format specifier of **strftime**. *Parameter* is a format string identical to the format string used with the **strftime** function in *Technical Reference: Base Operating System and Extensions*. Reference this function for a complete list of the format specifiers. For example, `.Dt "%A, %B %d, %Y (%T)"` provides the following output for an English-speaking locale: `Thursday, January 31, 1991 (10:40:00)` The %A format is replaced by the locale-specific weekday name. The %B format is replaced by the locale-specific month name. The %d format is replaced by the day of the month in a two-digit format. The %Y format is replaced by the year with the century as a decimal number. The %T format is replaced by the time in hours (24-hour clock), minutes, and seconds in decimal numbers. This format provides for leap seconds and double leap seconds.
.fl	Flushes output buffer. This request usually causes a line break similar to the **.br** request. Calling this request with the control character " **'** " (instead of "**.**") suppresses that break function. Initial Value: **-** If No Value Specified: **-**
.ig *Macro*	Ignores input lines. The **.ig** request works exactly like the **.de** request, except that the input is discarded. For more information, refer to "Macros, Strings, Diversions, and Position Traps". The input is read in copy mode, and any auto-incremented registers are affected. The *Macro* variable must be one or two ASCII characters. Initial Value: **-** If No Value Specified: **.***Macro*=**..**
.mc [*Character*] [*N*]	Uses the specified character as the margin character to display the specified distance (*N*) to the right of the margin after each non-empty text line (except those produced by the **.tl** request). If the output line is too long (as can happen in no-fill mode), the character is appended to the line. If the *N* variable is not given, the previous *N* variable is used. The first *N* variable is 0.2 inches in the **nroff** command and 1 em in the **troff** command. Relevant values are part of the current environment. Initial Value: .2 inches in **nroff**; 1 em in **troff** If No Value Specified: Off
.pm [*Character*]	Prints macros. The names and sizes of all of the defined macros and strings are printed on the user's terminal. If any ASCII alphanumeric character is given as a variable, only the total of the sizes is printed. The size is given in blocks of 128 characters. Initial Value: **-** If No Value Specified: All

Item	Description
.sy *Command* [*Flags*]	The specified command is run but its output is not captured at this point. The standard input for the specified command is closed. Output must be explicitly saved in an output file for later processing. Often the **.sy** directive is followed by a subsequent **.so** directive to include the results of the previous command. For example: `.sy date > /tmp/today` `Today is` `.so /tmp/today` Initial Value: **-** If No Value Specified: **-**
.tm *String*	The specified string is written to the user's terminal.
.tm ^A<*SetNumber*> <*MessageNumber*> [^A"<*DefaultMessage*> "] [^A<*Argument*> ^B <*Argument*> ^B<*Argument*> ...]	Based on the message set number and the message number within the locale-specific catalog, the message catalog is read in copy mode and the corresponding message is written to the user's terminal. The initial sequence specifying the message set and message number can be omitted for backward compatibility. The ASCII code Control-A ^A delimits message identification, default message, and optional argument list. The ASCII code Control-B ^B delimits individual optional argument list. In the following example: `.tm ^A2 23^A"The typesetter is %1$s.On line` `%2$s."^A\*(.T^B\n(c.` 2 is the message set number. 23 is the message number. Text within quotes **"...**" is the default message. `\*(.T` is the first argument in troff for value of **-T**. `\n(c.` is the number of lines read from the input file. If you assume the **troff** command runs with the following conditions: • The message at set 2 and number 23 matches the default message. • The command line has **troff** using the **-T** option with device PSC. • The **.tm** directive is on line 539 in the input file. Then the following would be displayed on the user's terminal: `The typesetter is psc. On line 538.` The locale-specific message catalog is found in **/usr/lib/nls/msg/$LANG/macros.cat**. Initial Value: **-** If No Value Specified: Newline

Note:

The following notes apply to the **nroff** and **troff** requests. They are referenced by number in the requests where they apply.

1. The **.L** string register contains the current program locale value of all the categories.
2. The **.m** string register contains the locale value of the **LC_MESSAGES** category.

3. The **.t** string register contains the locale value for the **LC_TIME** category.
4. While the **.L**, **.t**, and **.m**string registers provide access to some environment values, a more general technique can be used to access any other environment variable. For example, if the **TED** environment variable is exported, the following **troff** commands:

```
.sy echo .ds z $TED >x
.so x
.sy rm x
```

set the *z* string register to contain the value of **$TED**.

Environment Variables

Item	Description
LC_ALL	Specifies the locale to be used for all the locale categories. It overrides any setting of the other locale environment variables.
LC_MESSAGES	Specifies the locale value for the **LC_MESSAGES** category. This is used if the **LC_ALL** environment variable is not set.
LC_TIME	Specifies the locale value for the **LC_TIME** category. This is used if the **LC_ALL** environment variable is not set.
LANG	Specifies the locale value to be used for all the locale categories. This is used if none of the above environment variables are set. This is the most often used environment variable to specify the locale.

Files

Item	Description
/usr/share/lib/tmac/tmac.*	Contains the pointers to standard macro files.
/usr/share/lib/macros/*	Denotes standard macro files.
/usr/share/lib/tmac/tmac.an	Contains the pointer to the **man** macro package.
/usr/share/lib/macros/an	Contains the **man** macro package.
/usr/share/lib/tmac/tmac.e file	Contains the **me** macro definition file.
/usr/share/lib/me directory	Contains the macro definition files.
/usr/share/lib/tmac/tmac.m	Contains the pointer to the **mm** macro package.
/usr/share/lib/macros/mmn	Contains the **mm** macro package.
/usr/share/lib/macros/mmt	Contains the **mm** macro package.
/usr/share/lib/tmac/tmac.ptx	Points to the macro package.
/usr/share/lib/macros/ptx	Contains the macro package.
/usr/share/lib/tmac/tmac.x	Contains the macro definition files.
/usr/share/lib/ms	Contains the **ms** macro definitions.
/usr/share/lib/tmac/tmac.v	Contains macro definitions.
/usr/share/lib/macros/vmca	Contains macro definitions.
/usr/lib/nls/msg/$LANG/macros.cat	Contains locale-specific message catalog for the **mm**, **me**, **ms**, and **mv** macro packages.
/usr/lib/font/dev*/*	Contains the font width tables.
/var/tmp/trtmp*	Denotes a temporary file.

trpt Command

Purpose

Performs protocol tracing on TCP sockets.

Syntax

trpt [**-a**] [**-f**] [**-j**] [**-p***Address*]... [**-s**] [**-t**]

Description

The **trpt** command queries the buffer for Transmission Control Protocol (TCP) trace records. This buffer is created when a socket is marked for debugging with the **setsockopt** subroutine. The **trpt** command then prints a description of these trace records.

> **Note:** You can use the **traceson** command to turn on socket level debugging for daemons.

When you specify no options, the **trpt** command prints all the trace records found in the system and groups them according to their TCP/IP connection protocol control block (PCB).

Before you can use the **trpt** command, you must:

1. Isolate the problem and mark for debugging the socket or sockets involved in the connection.
2. Find the address of the protocol control blocks associated with these sockets by using the **netstat -aA** command.
3. Then you can run the **trpt** command, using the **-p** flag to supply the associated protocol control block addresses. You can specify multiple **-p***Address* flags with a single **trpt** command.

The **-f** flag can be used to follow the trace log once it is located. The **-j** flag can be used to check the presence of trace records for the socket in question.

If the system image does not contain the proper symbols to find the trace buffer, the **trpt** command cannot succeed.

Output Fields

The information put out by the **trpt** command varies with the flag you use. Definitions of the fields contained in the various types of output follow:

Item	Description
Protocol Control Block identifier	Identifies the protocol block to be traced, as shown in the following example: `4c500c:`
Timestamp	Specifies the time at which the connection is attempted, as shown in the following example: `500`

Item	Description
Connection State	Specifies the state of the connection with the protocol control block: **CLOSED** Connection is closed. **LISTEN** Listening for a connection. **SYN_SENT** Active; have sent SYN. Represents waiting for a matching connection request after having sent a connection request. **SYN_RCVD** Have sent and received SYN. Represents waiting for a confirming connection request acknowledgment after having both received and sent connection requests. **ESTABLISHED** Connection established. **CLOSE_WAIT** Have received FIN; waiting to receive CLOSE. **LAST_ACK** Have received FIN and CLOSE; awaiting FIN ACK. **FIN_WAIT_1** Have closed; sent FIN. **CLOSING** Closed; exchanged FIN; awaiting FIN. **FIN_WAIT_2** Have closed; FIN is acknowledged; awaiting FIN. **TIME_WAIT** In 2MSL (twice the maximum segment length) quiet wait after close.
Action	Specifies the current status of the packet trace connection. The output of the command changes depending on the action. **Input** Receiving input packets. The syntax of the output is: `input (SourceAddress, Port, DestinationAddress, Port) <Sequence Number of the First Data Octet> @ AcknowledgementNumber` as in the following example: `input (src=129.353173176,23, dst=129.35.17.140, 1795) fb9f5461@fb9e4c68` **Output** Transmitting packets. The syntax of the output is: `output (SourceAddress, Port, DestinationAddress, Port) <Sequence Number Of The First Data Octet>.. <Sequence Number of the Last Data Octet>@ AcknowledgementNumber)` as in the following example: `output (src=129.35.17.140,1795, dst=129.35.17.176, 23) fb9e4c68@fb9f5462` **Window Size** Specifies the size of the window sending or receiving packets, as shown in the following example: `(win=1000)`

Item	Description
User	Specifies user request. The following is an example of a user request: `SLOWTIMO<KEEP>` Types of user requests and their definitions follow: **PRU_ATTACH** Attach protocol to up. **PRU-DETACH** Detach protocol from up. **PRU_BIND** Bind socket to address. **PRU_LISTEN** Listen for connection. **PRU_CONNECT** Establish connection to peer. **PRU_ACCEPT** Accept connection from peer. **PRU_DISCONNECT** Disconnect from peer. **PRU_SHUTDOWN** Will not send any more data. **PRU_RCVD** Have taken data; more room now. **PRU_SEND** Send this data. **PRU_ABORT** Abort (fast DISCONNECT, DETACH). **PRU_CONTROL** Control operations on protocol. **PRU_SENSE** Return status into m. **PRU_RCVOOB** Retrieve out of band data. **PRU_SENDOOB** Send out of band data. **PRU_SOCKADDR** Fetch socket's address. **PRU_PEERADDR** Fetch peer's address. **PRU_CONNECT2** Connect two sockets. **PRU_FASTTIMO** 200 milliseconds timeout. **PRU_SLOTIMO** 500 milliseconds timeout. **PRU_PROTORCV** Receive from below. **PRU_PROTOSEND** Send to below.
Drop	Specifies that data was in preceding segment; data is dropped.

Item	Description
Window and Sequence Variables	Types of window and sequence variables follow: ***rcv_nxt*** Next sequence number expected on incoming segments. ***rcv_wnd*** Size of receive window. ***snd_una*** Oldest unacknowledged sequence number. ***snd_nxt*** Next sequence number to be sent. ***snd_max*** Highest sequence number sent. ***snd_sl1*** Window update segment sequence number. ***snd_wl1*** Window update segment ack number. ***snd_wnd*** Send window.

Flags

Item	Description
-a	Prints the values of the source and destination addresses for each packet recorded, in addition to the normal output.
-f	Follows the trace as it occurs, waiting briefly for additional records each time the end of the log is reached.
-j	Lists just the protocol control block addresses for which trace records exist.
-p*Address*	Shows only trace records associated with the protocol control block specified in hexadecimal by the *Address* variable. You must repeat the **-p** flag with each *Address* variable specified.
-s	Prints a detailed description of the packet-sequencing information, in addition to the normal output.
-t	Prints the values for all timers at each point in the trace, in addition to the normal output.

Examples

1. To print trace information as well as the source and destination addresses for each packet recorded, enter:

   ```
   $ trpt -a
   ```

   This might display the following output:

   ```
   124b0c:
   900 ESTABLISHED:input (src=192.9.201.3,4257, dst=192.9.201.2,102
   5)2326e6e5@ad938c02(win=200)<ACK,FIN,PUSH> -> CLOSE_WAIT
   900 CLOSE_WAIT:output (src=192.9.201.2,1025, dst=192.9.201.3,425
   7)ad938c02@2326e6e6(win=4000)<ACK> -> CLOSE_WAIT
   900 LAST_ACK:output (src=192.9.201.2,1025, dst=192.9.201.3,4257)
   ad938c02@2326e6e6(win=4000)<ACK,FIN> -> LAST_ACK
   900 CLOSE_WAIT:user DISCONNECT -> LAST_ACK
   900 LAST_ACK:user DETACH -> LAST_ACK 12500c:
   800 ESTABLISHED:output (src=192.9.201.2,1024, dst=192.9.201.3,51
   2)ad8eaa13@2326e6e5(win=4000)<ACK> -> ESTABLISHED
   800 ESTABLISHED:input (src=192.9.201.3,512, \
   dst=192.9.201.2,1024)
   [2326e6e5..2326e727)@ad8eaa13(win=1ef)<ACK,PUSH> -> ESTABLISHED
   800 ESTABLISHED:user RCVD -> ESTABLISHED
   900 ESTABLISHED:output (src=192.9.201.2,1024, dst=192.9.201.3,51
   2)ad8eaa13@2326e727(win=4000)<ACK> -> ESTABLISHED
   900 ESTABLISHED:input (src=192.9.201.3,512, \
   dst=192.9.201.2,1024)
   [2326e727..2326e82f)@ad8eaa13(win=1ef)<ACK,PUSH> -> ESTABLISHED
   900 ESTABLISHED:user RCVD -> ESTABLISHED
   900 ESTABLISHED:output (src=192.9.201.2,1024, dst=192.9.201.3,51
   2)ad8eaa13@2326e82f(win=4000)<ACK> -> ESTABLISHED
   900 ESTABLISHED:input (src=192.9.201.3,512, \
   dst=192.9.201.2,1024)
   2326e82f@ad8eaa13(win=1ef)<ACK,FIN,PUSH> -> CLOSE_WAIT
   900 CLOSE_WAIT:output (src=192.9.201.2,1024, \
   dst=192.9.201.3,512)
   ```

```
ad8eaa13@2326e830(win=4000)<ACK> -> CLOSE_WAIT
900 LAST_ACK:output (src=192.9.201.2,1024, dst=192.9.201.3,512)a
d8eaa13@2326e830(win=4000)<ACK,FIN> -> LAST_ACK
900 CLOSE_WAIT:user DISCONNECT -> LAST_ACK
900 LAST_ACK:user DETACH -> LAST_ACK
$ _
```

2. To list the protocol control blocks that have trace records, enter:

```
trpt -j
```

This might display the following output:

```
124b0c, 12500c
```

3. To print the trace records associated with a single protocol control block, enter:

```
trpt -p 12500c
```

This might display the following output:

```
800 ESTABLISHED:output ad8eaa13@2326e6e5(win=4000)<ACK> ->
ESTABLISHED
800 ESTABLISHED:input [2326e6e5..2326e727)@ad8eaa13(win=1ef)
<ACK,PUSH> -> ESTABLISHED
800 ESTABLISHED:user RCVD -> ESTABLISHED
900 ESTABLISHED:output ad8eaa13@2326e727(win=4000)<ACK> -> ESTABLISHED
900 ESTABLISHED:input [2326e727..2326e82f)@ad8eaa13(win=1ef) <ACK,PUSH> -> ESTABLISHED
900 ESTABLISHED:user RCVD -> ESTABLISHED
900 ESTABLISHED:output ad8eaa13@2326e82f(win=4000)<ACK> -> ESTABLISHED
900 ESTABLISHED:input 2326e82f@ad8eaa13(win=1ef)<ACK,FIN,PUSH> -> CLOSE_WAIT
900 CLOSE_WAIT:output ad8eaa13@2326e830(win=4000)<ACK> -> CLOSE_WAIT
900 LAST_ACK:output ad8eaa13@2326e830(win=4000)<ACK,FIN> -> LAST_ACK
900 CLOSE_WAIT:user DISCONNECT -> LAST_ACK
900 LAST_ACK:user DETACH -> LAST_ACK
$ _
```

true or false Command

Purpose

Returns an exit value of zero (true) or a nonzero exit value (false).

Syntax

true

false

Description

The **true** command returns a zero exit value. The **false** command returns a nonzero exit value. These commands are most often used as part of a shell script.

Examples

To construct a loop that displays the date and time once each minute, use the following code in a shell script:

```
while true
do
   date
   sleep 60
done
```

truss Command

Purpose

Traces a process's system calls, dynamically loaded user level function calls, received signals, and incurred machine faults.

Syntax

truss [**-f**] [**-c**] [**-a**] [**-l**] [**-d**] [**-D**] [**-e**] [**-i**] [{ **-t** | **-x**} [**!**] *Syscall* [...]] [**-s** [**!**] *Signal* [...]] [{ **-m** }[**!**] *Fault* [...]] [{ **-u-r** | **-w**} [**!**] *FileDescriptor* [...]] [{ } [**!**]*LibraryName* [...]:: [**!**]*FunctionName* [...]] [**-o** *Outfile*] {*Command*| **-p** *pid* [. . .]}[**-X**]

Description

The **truss** command executes a specified command, or attaches to listed process IDs, and produces a trace of the system calls, received signals, and machine faults a process incurs. Each line of the trace output reports either the *Fault* or *Signal* name, or the *Syscall* name with parameters and return values. The subroutines defined in system libraries are not necessarily the exact system calls made to the kernel. The **truss** command does not report these subroutines, but rather, the underlying system calls they make. When possible, system call parameters are displayed symbolically using definitions from relevant system header files. For path name pointer parameters, **truss** displays the string being pointed to. By default, undefined system calls are displayed with their name, all eight possible argments and the return value in hexadecimal format.

When the **-o** flag is used with **truss**, or if standard error is redirected to a non-terminal file, **truss** ignores the hangup, interrupt, and signals processes. This facilitates the tracing of interactive programs which catch **interrupt** and **quit** signals from the terminal.

If the trace output remains directed to the terminal, or if existing processes are traced (using the **-p** flag), then **truss** responds to **hangup**, **interrupt**, and **quit** signals by releasing all traced processes and exiting. This enables the user to terminate excessive trace output and to release previously existing processes. Released processes continue to function normally.

For those options which take a list argument, the name **all** can be used as a shorthand to specify all possible members of the list. If the list begins with a **!**, the meaning of the option is negated (for example, exclude rather than trace). Multiple occurrences of the same option may be specified. For the same name in a list, subsequent options (those to the right) override previous ones (those to the left).

Every machine fault, with the exception of a page fault, results in posting a signal to the process which incurred the fault. A report of a received signal immediately follows each report of a machine fault, unless that signal is being blocked by the process.

To avoid collisions with other controlling processes, **truss** does not trace a process which it detects is being controlled by another process with the **/proc** interface.

The trace output for multiple processes is not produced in strict time order. For example, a read on a pipe may be reported before the corresponding write. However, for each process the output is strictly time-ordered. The trace output contains tab characters and standard tab stops are set at every eight positions.

The system may run out of per-user process slots when tracing children. This is because when tracing more than one process, **truss** runs as one controlling process for each process being traced, doubling the number of process slots being used for any given process. The usual system-imposed limit of 25 processes per user should be taken into account prior to running a trace on multiple processes

The operating system enforces certain security restrictions on the tracing of processes. You must have access privileges to the commands you are tracing. The **set-uid** and **set-gid** processes can only be traced by a privileged user. The **truss** command loses control of any process which performs an execution of a set-id or unreadable object file, unless it is run by a privileged user. These untraced processes continue normally and independently of truss from the point of the execution.

The lightweight processes (LWP) mentioned in truss output are really kernel threads. The option **-l** displays the LWP id (i.e. the thread id) on each line of the trace output.

User library functions in AIX libraries have both static and dynamic loaded function calls. The tracing with option **-u** is done for dynamically loaded function calls only.

User level function call tracing for dynamically loaded function calls is provided with **-u** option. This option will produce an entry/exit trace of the function calls.

Flags

Item	Description
-a	Displays the parameter strings which are passed in each exec system call.
-c	Counts traced system calls, faults, and signals rather than displaying trace results line by line. A summary report is produced after the traced command terminates or when **truss** is interrupted. If the **-f** flag is also used, the counts include all traced Syscalls, Faults, and Signals for child processes.
-d	A timestamp will be included with each line of output. Time displayed is in seconds relative to the beginning of the trace. The first line of the trace output will show the base time from which the individual time stamps are measured. By default timestamps are not displayed.
-D	Delta time is displayed on each line of output. The delta time represents the elapsed time for the LWP that incurred the event since the last reported event incurred by that thread. By default delta times are not displayed.
-e	Displays the environment strings which are passed in each exec system call.
-f	Follows all children created by the **fork** system call and includes their signals, faults, and system calls in the trace output. Normally, only the first-level command or process is traced. When the **-f** flag is specified, the process id is included with each line of trace output to show which process executed the system call or received the signal.
-i	Keeps interruptible sleeping system calls from being displayed. Certain system calls on terminal devices or pipes, such as **open** and **kread**, can sleep for indefinite periods and are interruptible. Normally, **truss** reports such sleeping system calls if they remain asleep for more than one second. The system call is then reported a second time when it completes. The **-i** flag causes such system calls to be reported only once, upon completion.
-l	Display the id (thread id) of the responsible LWP process along with truss output. By default LWP id is not displayed in the output.

Item	Description
-m [!]*Fault*	Traces the machine faults in the process. Machine faults to trace must be separated from each other by a comma. Faults may be specified by name or number (see the **sys/procfs.h** header file). If the list begins with the "!" symbol, the specified faults are excluded from being traced and are not displayed with the trace output. The default is **-m**`all` **-m**`!fltpage`.
-o *Outfile*	Designates the file to be used for the trace output. By default, the output goes to standard error.
-p	Interprets the parameters to **truss** as a list of process ids for an existing process rather than as a command to be executed. **truss** takes control of each process and begins tracing it, provided that the user id and group id of the process match those of the user or that the user is a privileged user.
-r [**!**] *FileDescriptor*	Displays the full contents of the I/O buffer for each read on any of the specified file descriptors. The output is formatted 32 bytes per line and shows each byte either as an ASCII character (preceded by one blank) or as a two-character C language escape sequence for control characters, such as horizontal tab (\t) and newline (\n). If ASCII interpretation is not possible, the byte is shown in two-character hexadecimal representation. The first 12 bytes of the I/O buffer for each traced read are shown, even in the absence of the **-r** flag. The default is **-r!all**.
-s [**!**] *Signal*	Permits listing *Signals* to trace or exclude. Those signals specified in a list (separated by a comma) are traced. The trace output reports the receipt of each specified signal even if the signal is being ignored, but not blocked, by the process. Blocked signals are not received until the process releases them. Signals may be specified by name or number (see sys/signal.h). If the list begins with the "**!**" symbol, the listed signals are excluded from being displayed with the trace output. The default is **-s all**.
-t [**!**] *Syscall*	Includes or excludes system calls from the trace process. System calls to be traced must be specified in a list and separated by commas. If the list begins with an "**!**" symbol, the specified system calls are excluded from the trace output. The default is **-tall**.

Item	Description
-u [!] [*LibraryName* [...]::[!]*FunctionName* [...]]	Traces dynamically loaded user level function calls from user libraries. The *LibraryName* is a comma-separated list of library names. The *FunctionName* is a comma-separated list of function names. In both cases the names can include name-matching metacharacters *****, **?**, **[]** with the same meanings as interpreted by the shell but as applied to the library/function name spaces, and not to files. A leading **!** on either list specifies an exclusion list of names of libraries or functions not to be traced. Excluding a library excludes all functions in that library. Any function list following a library exclusion list is ignored. Multiple **-u** options may be specified and they are honored left-to-right. By default no library/function calls are traced.
-w [**!**] *FileDescriptor*	Displays the contents of the I/O buffer for each write on any of the listed file descriptors (see **-r**). The default is **-w!all**.
-x [**!**] *Syscall*	Displays data from the specified parameters of traced sytem calls in raw format, usually hexadecimal, rather than symbolically. The default is **-x!all**.
-X	Displays data from the specified parameters of traced system calls in human-readable format. The supported system calls are **bind**, **connect**, **socketpair**, **lseek**, **creat**, **access**, **accept**, **socket**, and **statx**.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To produce a trace of the **find** command on the terminal, type:

```
truss find . -print >find.out
```

2. To trace the lseek, **close**, statx, and open system calls, type:

```
truss -t lseek,close,statx,open find . -print > find.out
```

3. To display thread id along with regular output for **find** command, enter:

```
truss -l find . -print >find.out
```

4. To display timestamps along with regular output for **find** command, enter:

```
truss -d find . -print >find.out
```

5. To display delta times along with regular output for **find** command, enter:

```
truss -D find . -print >find.out
```

6. To trace the **malloc()** function call and exclude the **strlen()** function call in the **libc.a** library while running the **ls** command, enter:

```
truss -u libc.a::malloc,!strlen ls
```

7. To trace all function calls in the **libc.a** library with names starting with "m", and exclude the **strlen()** function call in the **libc.a** library while running the **ls** command, enter:

```
truss -u libc.a::m*,!strlen ls
```

8. To trace all function calls in the **libc.a** library with names starting with "m" while running the **ls** command, enter:

```
truss -u libc.a::m* ls
```

9. To trace all function calls from the library **libcurses.a** and exclude calls from **libc.a** while running executable `foo`, enter:

```
truss -u libcurses.a,!libc.a::* foo
```

10. To trace the **refresh()** function call from **libcurses.a** and the **malloc()** function call from **libc.a** while running the executable `foo`, enter:

```
 truss -u libc.a::malloc -u libcurses.a::refresh foo
```

11. To trace the system calls arguments in human-readable format, enter:

```
truss -X -t lseek,bind,statx,creat find . -print > find.out
```

Files

/proc filesystem

trustchk Command

Purpose

Administration of Trusted Signature Database (TSD) and Trusted Execution function.

Syntax

Add Files to TSD

trustchk [**-R** *module name*] **-s** <private key file> **-v** <certificate file> **[-P]** **-a** [tree] { *filename* [size=VOLATILE] [hardlinks=*value*] [symlinks=*value*]...| **-f** *filename* }

Delete Files from TSD

trustchk **-d** { *filename*...| ALL | **-f** *filename* }

Query TSD

trustchk **-q** { *filename*...| ALL |**-f** *filename* }

Switch to New Hashing Algorithm

trustchk **-g** [SHA1 | SHA256 | SHA512]

System Scan

trustchk [-i] [-x] { -n | -t | -y } tree [dirpath.....]

Configure Policies

trustchk [**-@** { *WparName* | ALL }] **-p** { [**TE** [= ON | OFF]] [**CHKEXEC** [= ON | OFF]] [**CHKSHLIB** [= ON | OFF]] [**CHKSCRIPT** [= ON | OFF]] [**CHKKERNEXT** [= ON | OFF]] [**STOP_UNTRUSTD** [= ON | OFF | **TROJAN**]] [**STOP_ON_CHKFAIL** [= ON | OFF]] [**LOCK KERN POLICIES** [= ON | OFF]] [**TEP** [= ON | OFF | *PathList*]] [**TLP** [= ON | OFF | *PathList* [**TSD_FILES_LOCK** [= ON | OFF | **EXVOL**]] [**TSD_LOCK** [= ON | OFF]] }

System Audit

trustchk [**-l**] [**-r**] { **-n** | **-t** | **-y** } { *filename...* | ALL } }

Using Alternate TSD File

trustchk -F *TSDFile* { **-a** | **-d** | **-g** | **-q** | **-y** | **-n** | **-t** }

Update TSD trustchk

trustchk -u <*filename*>[<*attr*>=value]

trustchk -k -s <private key file> **-v** *certificate file* **[-N]** { [**-D**] "OU = distinguished name"}

Note: The plus sign (+) is a special character that can be used only with a distinguished name for the **-D** option.

The following example shows how to use the plus sign as a special character in a distinguished name:

```
trustchk -k -s sign-key -v verify-key -N -D
"OU=IT + OU=jj, OU=zlab037.austin.ibm.com"
```

You cannot use the plus sign in any other format.

Description

The **trustchk** command is used in the following situations:

- Managing the Trusted Signature Database
- Auditing the security state of the system
- Enabling the Trusted Execution Mechanism
- Configuring different policies for Trusted Execution
- Scanning the system for TROJAN detection
- Installing the software or interim fixes with Trusted Execution (TE) policies

Managing the Trusted Signature Database

Privileged users use the **trustchk** command to add, delete, or list entries to the Trusted Signature Database (TSD). The TSD is a database of security attributes of the trusted files that are present on the system. The TSD is in the **/etc/security/tsd/tsd.dat** file. This database gets populated at installation time. It stores the security attributes of the trusted files that are present on the system. The following attribute list forms a part of a file definition (stanza):

Attributes	Usage
Owner	Name of the owner of the file. The owner ID cannot be used.
Group	Name of the group of the file. The group ID cannot be used.
Type	Type of the definition. Specifies if the definition belongs to a file, directory, first-in-first-out special files (FIFO), character device, block device, or a multiplexed device .
Mode	Permission bits, along with additional parameters specifying whether SETUID, SETGID, TCB, or SVTX bits are set in the file.
hardlink	Colon-separated list of hard links pointing to the file.

Attributes	Usage
symlink	Colon-separated list of symbolic links pointing to the file.
size	Size of the file in bytes.
cert_tag	ID of the digital certificate that was used to calculate the signature of this file.
signature	Digital signature of the file calculated using RSA algorithm.
hash_value	Cryptographic hash value of the file. By default, the SHA256 value is used to calculate the hash value.
accessauths	Access authorization on the object.
innateprivs	Innate privileges for the file.
inheritprivs	Inheritable privileges for the file.
authprivs	Privileges that will be assigned to users if they have the given authorizations.
secflags	File security flags associated with the object.
minslabel	Minimum sensitivity label for the object. This is valid only on a Trusted AIX system. If no value is specified, the system low sensitivity label (SLSL) is assumed.
maxslabel	Maximum sensitivity label for the object. This is valid only on a Trusted AIX system. This attribute is not applicable to regular files and FIFO. If no value is specified, the system low sensitivity label (SLSL) is assumed.
intlabel	Integrity label for the object. This is valid only on a Trusted AIX system. If no value is specified, the system high integrity label (SHTL) is assumed.

Note: You must include a blank line between stanzas when you specify multiple stanzas in an external file with the -f flag.

Auditing the security state of the system

To audit the security state of the system, you must check the security parameters stored in the TSD against the parameters of the actual files present on the system. Use the **trustchk** command to do so. Any discrepancy in the values is pointed to the user based on the input flags specified. To check all of the files that are listed in the TSD, use the **ALL** parameter in place of *filename*. You can specify a list of files separated by spaces on the command line.

Enabling the Trusted Execution function

To enable or disable the runtime integrity-verification function that is responsible for verifying of a file's cryptographic hash before being started, use the **trustchk** command. To turn the Trusted Execution function on or off, use the **TE -p** flag.

Configuring different policies for Trusted Execution

To enable or disable different security policies that are used with the Trusted Execution mechanism, use the **trustchk** command. You can specify the following different policies:

Item	Description
CHKEXEC	Checks the integrity of executable file that belongs to the TSD before starting it.
CHKKERNEXT	Checks the integrity of the kernel extensions that belong to the TSD before loading them.
CHKSHLIB	Checks the integrity of shared libraries that belong to the TSD before loading them.
CHKSCRIPT	Checks the integrity of shell scripts that belong to the TSD before starting them.

Item	Description
LOCK_KERN_POLICIES	If this policy is disabled, then any policies can be enabled or disabled at any time. If this policy is enabled, then all of the other policies will be locked. To enable or disable a policy in such condition, disable the **LOCK_KERN_POLICIES** policy and then restart the system.
STOP_ON_CHKFAIL	Stops the loading of files whose integrity check fails.
STOP_UNTRUSTD	Stops the loading of files that do not belong to the TSD. **TROJAN** Stops the loading of files that do not belong to the TSD and have one of the following security settings: • Have suid/sgid bit set • Linked to a file in the TSD • Have entry in the **privcmds** Database • Be linked to a file in the **privcmds** database
TE	Enables or disables Trusted Execution. Policies can only be activated when the **TE** option is set to ON.
TEP	Sets the value of Trusted Execution path, and enables or disables it. The Trusted Execution path consists of a list of colon-separated absolute paths, for example, the **/usr/bin:/usr/sbin**. When this policy is enabled, the files belonging to only these directory paths are allowed to be started. If an executable program that does not belong to the TEP is to be loaded, the program is blocked.
TLP	Sets the value of Trusted Library path, and enables or disables it. The Trusted Library Path consists of a list of colon-separated absolute paths, for example, the **/usr/lib:/usr/ccs/lib**. When this policy is enabled, the libraries belonging to only these directory paths can be loaded. If a program tries to load a library that does not belong to the TLP, the program is blocked.
TSD_FILES_LOCK	Disables opening of files belonging to the TSD in write mode. **EXVOL** Disables the opening of only the nonvolatile files that belong to the TSD in write mode. The volatile files can be changed.
TSD_LOCK	Disallows opening of a TSD file (/etc/security/tsd/tsd.dat) in write mode to disable editing of the TSD.

By default, the TSD defines all the files and programs that are part of the trusted computing base, but the privileged user or a member of the security group can choose to define only those files considered to be security-relevant.

The TE policies are stored in the **/etc/security/tsd/tepolicies.dat** file.

This command writes messages to the standard error log (**stderr**).

Scanning the system for TROJAN detection

Trustchk has the capability to detect the system for TROJAN, if any executable is present on the system and you do not have the entry in TSD and have one of the following security settings:

- have suid/sgid bit set

- linked to a file in the TSD
- have entry in the **privcmds** database
- be linked to a file in the **privcmds** database

Installing the software or interim fixes with Trusted Execution (TE) policies

If the Trusted Execution (TE) policy is turned on along with the TSD_LOCK policy or the TSD_FILE_LOCK policy, the **installp** and **emgr** commands fail. To continue with the installation, manually turn off the TSD_LOCK policy or the TSD_FILE_LOCK policy. The **emgr** and **installp** commands run successfully with TE policies if the TSD_LOCK policy or the TSD_FILE_LOCK policy is not turned on.

Flags

Item	Description
-a *filename*	Adds file definitions in the TSD. The definitions are read from a file (the **-f** option) or are calculated by the command if you specify the absolute file name. The following parameters can be specified by the user with the file name: **size=VOLATILE** Specifies the size of a file. This attribute can only use the VOLATILE value. The VOLATILE value indicates that the file that this definition belongs to is volatile in nature. The contents of the file change frequently, so during audits, the size, hash value and the signature of this file should not be checked. **hardlinks=*value*** Supplies the hard links to a file that cannot be computed independently by the **trustchk** command. **symlinks=*value*** Supplies the symbolic links to a file. **-tree** This tree parameter is used along with the **–a** flag. It supports adding of stanzas to the **trustchk** database recursively when the directory name is provided along with the **–a** flag. If the file name is mentioned, the stanza for the file name is added. To add a regular file to the TSD, you must specify the private key, or specify the signing key with the **-s** flag in ASN.1/DER in **PKCS#8** format without pass phrase (that is, password) protection. You must also specify the associated certificate with the **-v** flag in ASN.1/DER. The associated certificate contains the public key that will be used to verify the signature of the file. The digital certificate that you specified is copied to a certificate store in the **/etc/security/certificates** file so that it can be used during system audits to verify the signatures of the file. To add non-regular files, such as devices, directories and FIFO (that is, the first-in-first-out file), the private key and certificate is not required.
-d	Deletes file definitions from the TSD. The name of the file whose stanza needs to be deleted from the TSD is specified at command line, or is placed in a file that can be specified with the **-f** flag.
-D	This flag is used along with the **–k** flag when you want to enter the issuer DN and the Subject DN from the command-line interface.
-f *filename*	Specifies that file definitions are to be read from the file specified with the *filename* parameter. The file (or stanza) name must end with a colon. There must be a blank line between each file name entry in the external file.
-F	Specifies that a different the TSD file be used as a reference. This flag can be used with the **-a**, **-d**, **-g**, **-q**, **-n**, **-t**, or **-y** flags.

Item	Description
-g [SHA1 | SHA256 | SHA512]	Migrates the TSD to a new hashing algorithm. All of the **hash_value** fields in the file definitions are recomputed and updated in the TSD. The following algorithms are supported: SHA1, SHA256 and SHA512. To see the currently active algorithm, specify the **-g** flag without any algorithm names.
-i	Only used with **–n**,**-t**,**-y** options and long with **tree** parameter. It will ignore the scanning of NFS mounted filesystem.
-l	Specifies that only the Trusted AIX label attributes are to be verified. The **-l** option is valid only on a Trusted AIX system.
-k	Generates the certificate and the private key files by using the **trustchk** command. The key file name and certificate file names must be specified by **-s** and **-v** flag. The generated keys are saved in the files that are specified files by the **-s** and **-v** flags.
-n	Specifies the auditing mode, and indicates that the errors are to be reported. Any discrepancy between the attributes in the TSD and the actual file parameters are printed to the **stderr.** error file. To check all of the entries in the TSD, use the **ALL** parameter. To scan the entire system or directories for **TROJAN** detection, use with **tree** parameter.
-p	Configures Trusted Execution policies. You can turn on the policy configuration from command line, for example, policyA=ON. Specify a policy name to retrieve its current state (for example, **trustchk -p** CHKEXEC). The **TE**=ON option enables policies except the **TEP** and **TLP** policies that are not related to **TE** The **TEP** and **TLP** policies can be automatically turned ON or turn OFF. The **TEP**=ON option enables the **TEP**, and the **TLP**=ON option enables the **TLP** function.
-P	Prompts you to enter the password. This password is used to encrypt or decrypt the private-key file. This option can be used along with **–a** flag. When this flag is used with the**trustchk –a** command, it prompts you to enter the password which is used to decrypt the private-key file.
-q	Queries the TSD for a file name. Prints the entire list of security attributes, for example, stanza for the specified file name. To retrieve all of the entries of the TSD, use the **ALL** parameter instead of listing file path names.
-r	Specifies check that only the authorizations and privileges are to be checked. This flag is valid only on Enhanced RBAC and a Trusted AIX system. To check all of the entries in the TSD, use the **ALL** flag.
-R *module_name*	Specifies that the values for the TSD policy and TE policy to be taken from the module specified instead of the local copy.
-s	Specifies the signing key used for signature calculation of a file while adding it to the TSD. The signing key is an RSA private key in ASN.1/DER in **PKCS#8** format without pass phrase (that is, password) protection.
-t	Specifies the auditing mode and indicates that errors are to be reported with a prompt asking whether the error should be fixed. To check all of the entries in TSD, use the **ALL** option. To scan the entire system or directories for **TROJAN** detection, use with **tree** parameter.

Item	Description
-u	Updates the value of the specified attribute in TSD. If any of the **rbac** attributes are changed using the **trustchk –u** command, you must run the **setkst** explicitly. This updates the kernel table. **Note:** This flag supports the following attributes: Owner, group, mode, Hardlinks, symlinks, accessauths, innateprivs, inheritprivs, authprivs, secflags, t_innateprivs, t_inheritprivs, t_secflags, t_authprivs, t_accessauths, and type.
-v	Specifies the verification certificate that is associated with the signing key (using the **-s** flag). This certificate is copied into a certificate store in the **/etc/security/certificate** file, and is used to verify the file signature during auditing. If a certificate with the same certificate ID already exists in the store, then it is overwritten with a new certificate. The verification certificate is in ASN.1/DER format.
-x	Only used with **-n**, **-t**, **-y** options and long with **tree** parameter. Do not follow the symbolic link.
-y	Specifies the auditing mode, and indicates that errors are to be fixed and reported. To check all of the entries in the TSD, use the **ALL** parameter. To scan the entire system or directories for **TROJAN** detection, use with **tree** parameter. **Attention:** Use the **-y** option with care. It might make a file unusable if the **trustchk** command encounters a discrepancy.
-@ *WparName*	Lists the **TE** polices of a system WPAR.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error has occurred.

Examples

1. To add a new file definition for **/usr/bin/ls** using private key located at **/home/guest/privkey.der** and an associated certificate at **/home/guest/certificate.der** , enter the following command:

```
trustchk –s /home/guest/privkey.der –v /home/guest/certificate.der
–a  /usr/bin/ls
```

2. To add a file as a volatile file to the TSD using same pair of private key and certificate in the previous example, enter the following command:

```
trustchk –s /home/guest/privkey.der –v /home/guest/certificate.der
 –a  /usr/bin/passwd size=VOLATILE
```

3. To add a file **/usr/bin/ls** with a **/usr/local/bin/ls** hardlink to TSD using same pair of private key and certificate in the first example, enter the following command:

```
 trustchk–s /home/guest/privkey.der –v /home/guest/certificate.der
–a  /usr/bin/ls hardlinks=/usr/local/bin/ls
```

4. To delete a file **/usr/bin/logname**, enter the following command:

```
 trustchk –d  /usr/bin/logname
```

5. To add file definitions stored in a file **/home/guest/filedef.in**, enter the following command:

```
trustchk –s /home/guest/privkey.der
–v /home/guest/certificate.der
–a  -f /home/guest/filedef.in
```

6. To enable policy for checking executable file listed in the TSD on every load, follow these steps:
   a. Configure the policy by entering the following command:

```
trustchk –p CHKEXEC=ON
```

   b. Activate the policy by entering the following command:

```
trustchk –p TE=ON
```

7. To check the integrity of all of the files belonging to the TSD, enter the following command:

```
trustchk –n ALL
```

8. To print the value of the currently active hash algorithm for TSD, enter the following command:

```
trustchk -g
```

9. To list all the policies of a WPAR, enter the following command:

```
trustchk -@ <wpar> -p
```

10. To list all the policies of all WPARs, enter the following command:

```
trustchk -@ ALL -p
```

11. To scan the whole system for a TROJAN detection report only, enter the following command:

```
trustchk –n tree
```

12. To scan only **dir /usr** for TROJAN detection and automatically fix them, enter the following command:

```
trustchk  -y /usr
```

13. To scan the entire system for TROJAN detection, except NFS mounts filesystem, and fixes them interactively, enter the following command:

```
trustchk –i –t tree
```

14. To take the values from the LDAP server instead of the local copy, enter the following command:

```
trustchk –R LDAP -p
```

tset Command

Purpose

Initializes terminals.

Syntax

tset [**-e** *C*] [**-k** *C*] [**-i** *C*] [**-**] [**-s**] [**-I**] [**-Q**] [**-m** [*Identifier*] [*TestBaudRate*] **:***Type*] ... [*Type*]

Description

The **tset** command enables you to set the characteristics of your terminal. It performs terminal-dependent processing, such as setting erase and kill characters, setting or resetting delays, and sending any sequences needed to properly initialize the terminal.

The **tset** command first determines the type of terminal involved (specified by the *Type* parameter). It then performs necessary initializations and mode settings. The type of terminal attached to each port is specified in the Object Data Manager (ODM) database. The terminfo database contains possible type names for terminals. If a port is not wired permanently to a specific terminal (that is, it is not hardwired), the **tset** command gives it an appropriate generic identifier, such as `dialup`.

When no flags are specified, the **tset** command reads the terminal type out of the **TERM** environment variable and re-initializes the terminal.

When the **tset** command is used in a startup script (the **.profile** file for **sh** users or the **.login** file for **csh** users), the script should include information about the type of terminal you will usually use on ports that are not hardwired. These ports are identified in the ODM database as `dialup, plugboard,` or ARPANET, among others. To specify which terminal type you usually use on these ports, use the **-m** flag (followed by the appropriate port type identifier), an optional baud rate specification, and the terminal type. If more than one mapping is specified, the first applicable mapping prevails. A missing port type identifier matches all identifiers. Any of the alternate generic names given in the **terminfo** database can be used as the identifier.

You can specify the baud rate in the **tset** command as you would with the **stty** command. The baud rate is compared with the speed of the diagnostic output (which should be the control terminal). The baud rate test can be any combination of the following characters:

- `.` (period)
- `@` (at sign)
- < (less than sign)
- `!` (exclamation point)

The `@` (at sign) stands for the preposition at, and the `!` (exclamation point) inverts the sense of the test. To avoid problems with metacharacters, place the **-m** flag argument inside `' '` (single quotes). Users of the **csh** command must also put a \ (backslash) before any `!` (exclamation point).

The following example sets the terminal type to `adm3a` if the port in use is a dialup at a speed greater than `300` baud. It sets the terminal type to `dw2` if the port is a dialup port at a speed of `300` baud or less:

```
tset -m 'dialup>300:adm3a' -m dialup:dw2 -m 'plugboard:?adm3a'
```

If the *Type* parameter begins with a ? (question mark), you are prompted to verify the type. To use the specified type, press Enter. To use a different type, enter the type you want. In the example given, you are prompted to verify the `adm3` plugboard port type.

If no mapping applies and a final type option (not preceded by an **-m** flag) is given on the command line, that type is used. Otherwise, the default terminal type is the one identified in the ODM database. Hardwired ports should always be identified in the ODM database.

When the terminal type is known, the **tset** command engages in terminal driver mode setting. This usually involves setting:

- An initialization sequence to the terminal
- The single character erase and optionally the line-kill (full-line erase) characters
- Special character delays

Tab and new-line expansion are turned off during transmission of the terminal initialization sequence.

On terminals that can backspace but not overstrike (such as a CRT), and when the erase character is the default erase character (# on standard systems), the erase character is changed to Backspace (Ctrl-H).

Flags

Item	Description
-e *C*	Sets the erase character to the character specified by the *C* parameter.The default is the backspace character.

Item	Description
-I	Suppresses transmission of terminal initialization strings.
-i *C*	Sets the interrupt character to the character specified by the *C* parameter. The *C* parameter defaults to ^C (caret C). The ^ (caret) character can also be used for this option.
-k *C*	Sets the line-kill character to the character specified by the *C* parameter. The *C* parameter defaults to ^X (caret X). The ^ (caret) character can also be used for this option.
-m *IdentifierTestbaudRate*:*Type*	Specifies which terminal type (in the *Type* parameter) is usually used on the port identified in the *Identifier* parameter. A missing identifier matches all identifiers. You can optionally specify the baud rate in the *TestBaudRate* parameter.
-Q	Suppresses printing of the `Erase set to` and `Kill set to` messages.
-s	Prints the sequence of **csh** commands that initialize the **TERM** environment variable, based on the name of the terminal decided upon.
-	The name of the terminal decided upon is output to standard output. This is the **TERM** environment variable.

Examples

The following examples all assume the Bourne shell and usage of the **-** flag. If you use the **csh** command, use the preceding variations. A typical use of the **tset** command in a **.profile** or **.login** file includes the **-e** and **-k** flags, and often the **-n** or **-Q** flags as well. To streamline the examples, these flags have not been included here.

Note: Make sure to enter the **tset** command all on one line regardless of the number of lines used in the example.

1. Now you are a 2621 terminal. Do not use the following example in your **.profile** file, unless you are always a 2621 terminal.

   ```
   export TERM; TERM=\'tset \- 2621\'
   ```

2. You have an h19 terminal at home that you dial up on, but your office terminal is hardwired and specified in the ODM database.

   ```
   export TERM; TERM=\'tset \- \-m dialup:h19"'
   ```

3. You have a switch that connects everything to everything, making it nearly impossible to key on what port you are coming in. You use a vt100 in your office at 9600 baud and dial up from home on a 2621 to switch ports at 1200 baud. Sometimes, you use a different terminal at work. At high speeds, you want to verify your terminal type, but at 1200 baud, you are always on a 2621. Note how the quotation marks protect the greater-than sign and the question mark from interpretation by the shell.

   ```
   export TERM; TERM=\'tset \- \-m 'switch>1200:?vt100' \-m
   'switch<=1200:2621'
   ```

   If none of the conditions hold, the terminal type specified in the ODM database is used.

4. The following entry is appropriate if you always dial up at the same baud rate on many different terminals. Your most common terminal is an `adm3a`. You are always prompted to verify the terminal type, which defaults to `adm3a`.

   ```
   export TERM; TERM=\'tset \- \?adm3a\'
   ```

5. If the ODM database is not properly installed and you want to key entirely on the baud rate, type:

```
export TERM; TERM=\'tset \- \-m 'switch>1200:?vt100' \-m
'switch<=1200:2621'
```

6. You dial up at 1200 baud or less on a Concept100, sometimes over switch ports and sometimes over regular dialups. You use various terminals at speeds higher than 1200 over switch ports, most often the terminal in your office, which is a vt100. However, sometimes you log in from the university over the ARPANET; in this case, you are on an ALTO emulating a dm2500. You also often log in on various hardwired ports, such as the console, all of which are properly entered in the ODM database. To set your erase character to Ctrl-H and your kill character to Ctrl-U, type:

```
export TERM
TERM=\'tset \-e \-k(hat)U \-Q \- "-m 'switch<1200:concept100'
"-m 'switch:?vt100' \-m dialup:concept100 "1-m arpanet: dm2500"'
```

   This also prevents the **tset** command from printing the following line:

```
Erase set to Backspace, Kill set to Ctrl-U
```

7. To set the erase character to a control character, type:

```
tset -e ^Y
```

Files

Item	Description
/usr/share/lib/terminfo	Contains the terminal capability database.

tsh Command

Purpose

Invokes the trusted shell.

Syntax

Press in sequence: the Ctrl+X, Ctrl+R keys.

tsh Command

Description

The **tsh** command is a command interpreter that provides greater security than the Korn shell (the standard login shell). Generally, a user calls the **tsh** shell by pressing Ctrl+X, Ctrl+R, the secure attention key (SAK) sequence, after a login. The **tsh** shell also can be invoked by defining it as the login shell in the **/etc/passwd** file.

To use the SAK sequence to invoke the trusted shell, the terminal the user is using must have SAK enabled, and the user must be allowed to use the trusted path. See the **Trusted Computing Base** in *Operating system and device management* for information on enabling SAK on a terminal, and see the **/etc/security/user** file and the **chuser** command for information on allowing a user to access the trusted path.

To exit from the **tsh** shell, use any of the following commands: the **logout** command, **shell** command, **su** command. The **logout** command ends the login session, while the other commands execute the user's initial program and continue the login session.

The trusted shell differs from the Korn shell in the following ways:

- The function and alias definitions are not supported. Alias definitions are only supported in the **/etc/tsh_profile** file.
- The **IFS** and **PATH** environment variables cannot be redefined.
- Only trusted programs can be run from the **tsh** shell.
- The history mechanism is not supported.
- The only profile used is the **/etc/tsh_profile** file.
- The trusted shell has the following built-in commands:

Item	Description
logout	Exits the login session and terminates all processes.
shell	Re-initializes the user's login session. The effect is the same as logging in to the system.
su	Resets the effective ID to the user's identity on the system and executes another trusted shell.

Security

Access Control: This command should be a standard user program and have the **trusted computing base** attribute.

Files Accessed:

Mode	File
r	**/etc/tsh_profile**

Examples

To invoke the trusted shell, press the Ctrl+X, Ctrl+R key sequence, the secure attention key (SAK).

Files

Item	Description
/usr/bin/tsh	Contains the **tsh** command.
/etc/tsh_profile	Contains initialization commands for the trusted shell.
/etc/passwd	Contains basic user attributes.
/etc/security/user	Contains the extended attributes of users.
/etc/security/login.cfg	Contains configuration information.

tsm Command

Purpose

Provides terminal state management.

Syntax

tsm *Port*

Description

The **tsm** command invokes the terminal state manager, which controls the ports used in the trusted path. The functions are:

- Establishing line communication modes and discipline - functions performed by the **getty** command.
- Verifying the user's account and identity, and setting the initial process credentials and environment - functions performed by the **login** command.
- Performing trusted path management if the secure attention key (SAK) is enabled for the port and the system login program is used.

  Note: The **tsm** command is not entered on the command line.

Trusted path management occurs in two phases:

Item	Description
login	This phase is in effect if a user has not successfully logged in. If the secure attention key (SAK) signal is detected, the system restarts getty-login type processing. The next login puts the user into the trusted state, if the port and the user support the trusted state.
shell	This phase occurs after successful user authentication. The command functions according to the user's **tpath** attribute. The following values are valid: **on** Provides standard trusted path management. When the secure attention key (SAK) signal is detected, all processes that access the port, except the **tsm** process and its siblings (including the trusted shell), are terminated the next time an attempt is made to access the port. The port is reset to its initial state and is marked as trusted, and the trusted shell command (the **tsh** command) is executed. **notsh** The user session terminates when the secure attention key (SAK) signal is detected. **always** The user is not allowed off the trusted path. The user's shell will always be the trusted shell, tsh. **nosak** The secure attention key (SAK) is disabled for the terminal, and the user's initial program runs.

You can configure the **tsm** command to create your home directory at your login if you do not have a home directory already. The **tsm** command calls the **mkuser.sys** command to create the home directory and customize the account. To enable this capability, set the **mkhomeatlogin** attribute of the **usw** stanza in the **/etc/security/login.cfg** file to true.

Security

Access Control: This command should grant execute (x) permission to any user. The command should be setuid to the root user and have the **trusted computing base** attribute.

Files Accessed:	
Mode	**File**
r	**/etc/objrepos/CuAt**
r	**/usr/lib/objrepos/PdAt**
r	**/etc/security/login.cfg**
r	**/etc/security/user**

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To provide terminal state management on `tty0`, add the following line to the **/etc/inittab** file:

```
tty0:2:respawn:/usr/sbin/tsm /dev/tty0
```

This initializes the port `/dev/tty0` and sets up the characteristics of the port.

Files

Item	Description
/usr/sbin/tsm	Contains the **tsm** command.
/etc/security/login.cfg	Contains configuration information.
/etc/security/user	Contains extended user attributes.

tsort Command

Purpose

Sorts an unordered list of ordered pairs (a topological sort).

Syntax

tsort [**—**] [*File*]

Description

The **tsort** command reads from *File* or standard input an unordered list of ordered pairs, builds a completely ordered list, and writes it to standard output.

The input *File* should contain pairs of non-empty strings separated by blanks. Pairs of different items indicate a relative order. Pairs of identical items indicate presence, but no relative order. You can use the **tsort** command to sort the output of the **lorder** command.

If *File* contains an odd number of fields, an appropriate error message is displayed.

Flag

Item	Description
—	(Double hyphen) Interprets all arguments following the — flag as file names. If the file is named —, use tsort — —.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Files

Item	Description
/usr/ccs/bin/tsort	Contains the **tsort** command.
/usr/ccs/bin/tsort	Contains symbolic link to the **tsort** command.

ttt Command

Purpose

Starts the tic-tac-toe game.

Syntax

ttt [**-e**] [**i**]

Description

The **ttt** command starts the tic-tac-toe game. This is a learning version but it learns slowly. It loses nearly 80 games before completely mastering the game. When you start the game you are prompted `Accumulated knowledge? (Yes or No)`. Entering y provides the computer with knowledge gained from previous games.

You are always X and your opponent is always O. You can either make the first move or pass to your opponent. To pass, press the enter key when prompted `Your move?` at the beginning of the game. The first to get three in a row wins the game. For example:

```
new game
123
456
789
Your move?
1
X03
456
789
Your move?
9
X00
456
78X
Your move?
5
You win
```

In the example, your first move was to place an X where 1 was located. The computer placed an O where the 2 was located. The game progressed until you had three in a diagonal row (1,5, 9). The game repeats until you quit. To quit the game, press the Interrupt (Ctrl-C) or End Of File (Ctrl-D) key sequence**.**

Flags

Item	Description
-e	Increases the speed of the learning.
-i	Displays the instructions prior to the start of the game.

Files

Item	Description
$HOME/ttt.a	Specifies the location of the learning file.
/usr/games	Specifies the location of the system's games.

tty Command

Purpose

Writes to standard output the full path name of your terminal.

Syntax

/usr/bin/tty [**-s**]

Description

The **tty** command writes the name of your terminal to standard output.

If your standard input is not a terminal and you do not specify the **-s** flag, you get the message `Standard input is not a tty`.

The following environment variables affect the execution of the **tty** command:

Item	Description
LANG	Determines the locale to use for the locale categories when neither the **LC_ALL** variable nor the corresponding environment variable beginning with **LC_** specifies a locale.
LC_ALL	Determines the locale to be used. This variable overrides any values for locale categories that are specified by any other environment variable beginning with **LC_** or by the **LANG** variable.
LC_CTYPE	Determines the locale for the interpretation of sequences of bytes of text data as characters. For example, this variable may specify multi-byte characters instead of single-byte characters.
LC_MESSAGES	Determines the language for messages.

Flags

Item	Description
-s	Suppresses reporting the path name.

Exit Status

This command returns the following exit values:

Item	Description
0	Standard input is a terminal.
1	Standard input is not a terminal.
>1	An error occurred.

Examples

1. To display the full path name of your display:

```
tty
```

2. To test whether or not the standard input is a terminal:

```
if tty -s
then
echo 'Enter the text to print:' >/dev/tty
qprt -
fi
```

If the standard input is a terminal, this displays the message "`Enter the text to print:`" as a prompt and prints the text that the user types. If the standard input is not a terminal, this displays nothing; it merely prints the text read from the standard input.

The `echo . . . >/dev/tty` displays the prompt on the screen even if you redirect the standard output of the shell procedure. This way the prompt is never written into an output file. The special file **/dev/tty** always refers to your terminal, although it also has another name such as **/dev/console** or **/dev/tty2**.

Files

Item	Description
/usr/bin/tty	Contains the **tty** command.
/dev/tty	Specifies the **tty** pseudo device.

tunchange Command

Purpose

Updates one or more tunable stanzas in a file.

Syntax

tunchange **-f** *Filename* (**-t** *Stanza* ({**-o** Parameter[=*Value*]} | **-D**) | **-m** *Filename2*)

Description

The **tunchange** command unconditionally updates a tunable file. It can also merge a second file with the current file.

Note: No message will be displayed (even when a parameter of type bosboot is changed).

Flags

Item	Description
-f *Filename*	Name of the updated tunable file. If the name does not include the '**/**' (forward slash) character, it is considered to be relative to **/etc/tunables**.
-t *Stanza*	Name of the stanza to update. *Stanza* is either **schedo**, **vmo**, **ioo**, **no**, **nfso**, or `raso`. *Stanza* corresponds to the name of the command which can update the parameter or parameters specified by the **-o** flag.
-o Parameter=*Value*	Parameter to be set to *Value*. It must be valid in the *Stanza* specified by the **-t** flag and consistent with the other parameters of the file specified by the **-f** flag.

Item	Description
-D	Resets all parameters of the *Stanza* to their default value.
-m *Filename2*	Merges the *Filename2* file with the current *Filename* file.

Exit Status

Item	Description
0	Changes were correctly applied.
>0	One of the following conditions caused an error: • The specified *Filename*, *Filename2*, or *Stanza* was invalid. • Parameter=*Value* was invalid for the Parameter. • No message was provided.

Examples

1. To update the **pacefork** parameter in the **/etc/tunables/nextboot** file, type:

```
tunchange -f nextboot -t schedo -o pacefork=10
```

2. To update the **pacefork** parameter in the **/home/mine/mytunable** file, type:

```
tunchange -f /home/mine/mytunable -t schedo -o pacefork=10
```

3. To reset all **schedo** stanza parameters to their default value in the **/etc/tunables/nextboot** file, type:

```
tunchange -f nextboot -t schedo -D
```

4. To merge the **/home/mine/mytunable** file with the **/etc/tunables/nextboot** file, type:

```
tunchange -f nextboot -m /home/mine/mytunable
```

Files

Item	Description
/usr/sbin/tunchange	Contains the **tunchange** command.
/etc/tunables/	Contains the default tunable files.

tuncheck Command

Purpose

Validates a tunable file.

Syntax

tuncheck [**-r** | **-p**] **-f** *Filename*

Description

The **tuncheck** command validates a tunable file. All tunables listed in the specified file are checked for range and dependencies. If a problem is detected, a warning is issued.

There are two types of validation:

against the current context
Checks to see if *Filename* could be applied immediately. Tunables not listed in *Filename* are interpreted as current values. The checking fails if a tunable of type **Incremental** is listed with a smaller value than its current value; it also fails if a tunable of type **Bosboot** or **Reboot** is listed with a different value than its current value.

against the next boot context
Checks to see if *Filename* could be applied during a reboot, that is, if it could be a valid nextboot file. Decreasing a tunable of type **Incremental** is allowed. If a tunable of type **Bosboot** or **Reboot** is listed with a different value than its current value, a warning is issued but the checking does not fail.

Additionally, warnings are issued if ***Filename*** contains unknown stanzas, or unknown tunables in a known stanza. However, that does not make the checking fail.

Upon success, the **AIX_level, Kernel_type** and **Last_validation** fields in the info stanza of the checked file are updated.

Flags

Item	Description
-f *Filename*	Specifies the name of the tunable file to be checked. If it does not contain the '**/**' (forward slash) character, the name is relative to **/etc/tunables**.
-p	Checks *Filename* in both current and boot contexts. This is equivalent to running **tuncheck** twice, one time without any flag and one time with the **-r** flag.
-r	Checks *Filename* in a boot context.

If **-p** or **-r** are not specified, *Filename* is checked according to the current context.

Tuning Parameter Types

Item	Description
Dynamic	Can be changed at any time.
Static	Can never be changed
Reboot	Can only be changed during the reboot sequence
Bosboot	Can only be changed by running bosboot and rebooting the machine
Mount	Changes made are only effective for future filesystems or directory mountings
Incremental	Can only be incremented, except at boot time.
Connect	Changes are only effective for future socket connections.

Exit Status

0
Filename is valid.

>0
Filename is invalid, message have been provided.

Examples

1. To check whether **mytunable** can be applied immediately, type:

```
tuncheck -f ./mytunable
```

2. To check whether **/etc/tunables/nextboot** can be applied during a reboot, type:

```
tuncheck -r -f nextboot
```

3. To check whether **/etc/tunables/nextboot** can be applied immediately and after a reboot, type:

```
tuncheck -p -f nextboot
```

Files

Item	Description
/usr/sbin/tuncheck	Contains the **tuncheck** command.
/etc/tunables	Contains all the tunable files.

tundefault Command

Purpose

Reset all tunable parameters to their default value.

Syntax

tundefault [**-r** | **-p**]

Description

The **tundefault** command launches all the tuning commands (**ioo**, **vmo**, **schedo**, **no**, **nfso**, and `raso`) with the **-D** flag. This resets all the AIX tunable parameters to their default value, except for parameters of type **Bosboot** and **Reboot**, and parameters of type **Incremental** set at values bigger than their default value, unless **-r** was specified. Error messages are displayed for any parameter change impossible to make.

Flags

Item	Description
-p	Makes the changes permanent: resets all the tunable parameters to their default values and updates the **/etc/tunables/nextboot** file.
-r	Defers the reset to their default value to next reboot. This clears stanza(s) in the **/etc/tunables/nextboot** file, and if necessary, proposes **bosboot** and warns that a reboot is needed

Tunable Parameter Types

Item	Description
Dynamic	Can be changed at any time.
Static	Can never be changed
Reboot	Can only be changed during the reboot sequence
Bosboot	Can only be changed by running bosboot and rebooting the machine

Item	Description
Mount	Changes made are only effective for future filesystems or directory mountings
Incremental	Can only be incremented, except at boot time.
Connect	Changes are only effective for future socket connections.

Examples

1. To permanently reset all tunable parameters to their default values, enter:

   ```
   tundefault -p
   ```

   All of the tuning commands are launched with the **-Dp** flags. This resets all the tunable parameters to their default value. This also updates the **/etc/tunables/nextboot** file. This command completely and permanently resets all tunable parameters to their default values.
2. To defer the setting of all tunable parameters until next reboot, enter:

   ```
   tundefault -r
   ```

   Calls all tuning commands with **-Dr**. This clears all of the stanzas in the **/etc/tunables/nextboot** file, and if necessary, proposes **bosboot** and displays a message warning that a reboot is necessary to make the changes effective.

Files

Item	Description
/usr/sbin/tundefault	Contains the **tundefault** command.
/etc/tunables/	Contains all the tunable files.

tunrestore Command

Purpose

Restores tunable parameter values from a file.

Syntax

tunrestore [**-r**] **-f** *Filename*

tunrestore -R

Restriction: tunrestore -R can only be called from **inittab**.

Description

The **tunrestore** command restores all tunable parameters values stored in a file.

The **tunrestore -f** *Filename* immediately applies *Filename*. All tunables listed in *Filename* are set to the value defined in this file. Tunables not listed in *Filename* are kept unchanged. Tunables explicitly set to DEFAULT are set to their default value.

The **tunrestore -r -f** *Filename* applies *Filename* for the next boot. This is achieved by checking the specified file for inconsistencies (the equivalent of running **tuncheck** on it) and copying it over to **/etc/tunables/nextboot**. If bosboot is necessary, the user will be offered to run it.

The **tunrestore -R** is only used during reboot. All of the tunables that are not yet set to the value defined in the **nextboot** file are modified. Tunables that are not listed in the **nextboot** file are forced to their default value. All actions, warnings and errors are logged into the **/etc/tunables/lastboot.log** file. Note

that when modification is made to restricted tunables, a system **errlog** entry is added, including the list of all tunable commands controlling the modified restricted tunables and a reference to the **/etc/tunables/lastboot.log** file.

Additionally, a new tunable file called **/etc/tunables/lastboot** is automatically generated. That file has all of the tunables listed with numerical values. The values representing default values are marked with the comment `DEFAULT VALUE`. The values that are different from the default values for restricted tunables are marked with the comment `# RESTRICTED not at default value`. The info stanza of the new tunable file includes the checksum of the **/etc/tunables/lastboot.log** file to make sure pairs of the **lastboot/lastboot.log** files can be identified.

Flags

Item	Description
-f *Filename*	Specifies the name of the tunable file to apply. If it does not contain the '**/**' (forward slash) character, the name is relative to **/etc/tunables**.
-r	Makes the specified file become the new **nextboot** file.
-R	Restores **/etc/tunables/nextboot** during boot process.

Tunable Parameter Types

Item	Description
Dynamic	Can be changed at any time.
Static	Can never be changed
Reboot	Can only be changed during the reboot sequence
Bosboot	Can only be changed by running bosboot and rebooting the machine
Mount	Changes made are only effective for future filesystems or directory mountings
Incremental	Can only be incremented, except at boot time.
Connect	Changes are only effective for future socket connections.

Examples

1. To restore all tunable values stored in **/etc/tunables/mytunable**, enter:

```
tunrestore -f mytunable
```

2. To validate **/etc/tunables/mytunable** and make it the new nextboot file, enter:

```
 tunrestore -r -f mytunable
```

Files

Item	Description
/usr/sbin/tunrestore	Contains the **tunrestore** command.
/etc/tunables	Contains tunable files.
/etc/tunables/nextboot	Contains the values to be applied during the next boot.
/etc/tunables/lastboot	Contains the values of all tunables after the last boot.

Item	Description
/etc/tunables/lastboot.log	Contains messages, warnings and errors emitted by tunrestore during the last boot.

tunsave Command

Purpose

Saves current tunable parameter values to a file.

Syntax

tunsave [**-a** | **-A**] **-f** | **-F** *Filename* [**-d** *Description*]

Description

The **tunsave** command saves the current state of tunable parameters in a file.

If *Filename* does not already exist, a new file is created. If it already exists, an error message prints unless the **-F** flag is specified, in which case, the existing file is overwritten.

Note that the saved restricted tunables that have been modified to a value different from the default value, are flagged with a comment `# RESTRICTED not at default value`, appended to the line.

Flags

Item	Description
-a	Saves all tunable parameters, including those who are currently set to their default value. These parameters are saved with the special value DEFAULT.
-A	Saves all tunable parameters, including those who are currently set to their default value. These parameters are saved numerically, and a comment, # DEFAULT VALUE, is appended to the line to flag them.
-d *Description*	Specifies the text to use for the *Description* field. Special characters must be escaped or quoted inside the *Description* field.
-f *Filename*	Specifies the name of the tunable file where the tunable parameters are saved. If *Filename* already exists, an error message prints. If it does not contain the '**/**' (forward slash) character, the *Filename* is relative to **/etc/tunables**.
-F *Filename*	Specifies the name of the tunable file where the tunable parameters are saved. If *Filename* already exists, the existing file is overwritten. If it does not contain the '**/**' (forward slash) character, the *Filename* is relative to **/etc/tunables**.

Examples

1. To save all tunables different from their default value into **/etc/tunables/mytunable**, enter:

   ```
   tunsave -f mytunable
   ```

2. To save all tunables, including those who are currently set to their default value, but replace the default values with the special value DEFAULT, enter:

   ```
   tunsave -a -f /home/admin/mytunable
   ```

3. To save all tunables, including those who are currently set to their default value using all numerical values, but flag the default values with the comment DEFAULT VALUE, enter:

```
tunsave -A -f mytunable
```

Files

Item	Description
/usr/bin/tunsave	Contains the tunsave command.
/etc/tunables	Contains all the saved files.

turnacct Command

Purpose

Provides an interface to the **accton** command to turn process accounting on or off.

Syntax

/usr/sbin/acct/turnacct on | **off** | **switch**

Description

The **turnacct** command provides an interface to the **accton** command to turn process accounting on or off. You must specify whether you want process accounting on or off, because there is no default.

The **switch** flag turns off accounting and moves the current active data file (**/var/adm/pacct**) to the next free name in the **/var/adm/pacct***incr* file, where *incr* is a number starting at 1 and increased by one for each additional **pacct** file. After moving the **pacct** file, the **turnacct** command again turns on accounting.

The **turnacct switch** command is usually called by the **ckpacct** command, running under the **cron** daemon, to keep the active **pacct** data file a manageable size.

Security

Access Control: This command should grant execute (x) access only to members of the adm group.

Files

Item	Description
/usr/sbin/acct	Contains the path to the accounting commands.
/var/adm/pacct	Contains the current file for process accounting.
/var/adm/pacct*	Used if the **pacct** file gets too large.

turnoff Command

Purpose

Sets the permission codes off for files in the **/usr/games** directory.

Syntax

turnoff

Description

The **turnoff** command sets the permission codes of files in the **/usr/games** directory. Root user authority is required to run this command.

The **turnoff** command looks for files in **/usr/games** whose permissions are set to 111 and sets these permissions to 000. If you install any new games in the **/usr/games** directory, set these permissions to 111.

Files

Item	Description
/usr/games	Contains the location of the system's games.

turnon Command

Purpose

Sets permission codes on for files in the games directory.

Syntax

turnon

Description

The **turnon** command sets the permission codes of files in the **/usr/games** directory. Root user authority is required to run this command.

The **turnon** command looks for files with permissions set to 000 and sets them to 111 (execute permission for all users). If you install any new games in the **/usr/games** directory, set these permissions to 111.

File

Item	Description
/usr/games	Contains the location of the system's games.

tvi Command

Purpose

Provides a trusted editor with a full screen display.

Syntax

tvi [**- l**] [**-R**] [**-w** *Number*] [**-c** [*Subcommand*]] [*File* ...]

Description

The **tvi** command calls the **tvi** editor, a trusted version of the **vi** editor, to edit the file or files specified by the *File* parameter. Files are edited in the order specified. If you do not provide a file name, the command opens a new file in which you can create text, but if you try to save the text to a file, you are prompted to add a file name to the save command, such as **:w** *File*. See the Examples section for more information.

You enter and leave the **tvi** editor in command mode, but to add or change text, you must enter text input mode. See the description of text input mode for information about the subcommands that initiate text input mode. To leave text input mode, press the **Esc** key. This returns you to command mode where you can save the text to a file with one of the **:w** commands, and exit the **tvi** editor, for example, with the **:q** command.

Because the full-screen display editor started by the **tvi** command is based on the **ex** editor, you can use the **ex** subcommands within the **tvi** editor. Subcommands function at the cursor position on the display screen.

The **tvi** editor makes a copy of the file you are editing in an edit buffer. The contents of the file are not changed until you save the changes.

Note: Several functions of the **vi** editor are not supported by the **tvi** editor. If you refer to information on the **vi** editor, be aware that the **-r** flag, the **-t** flag, shell escapes, user-defined macros, key mapping, and setting **vi** options permanently are not supported by the **tvi** editor.

tvi Editor Limitations

The maximum limits of the **tvi** editor assume single-byte characters. The limits are as follows:

- 256 characters per global command list
- 2048 characters in a shell escape command
- 128 characters in a string-valued option
- 30 characters in a tag name
- 524,230 lines silently enforced
- 128 map macros with 2048 characters total

Editing Modes

The **tvi** editor operates in the following modes:

Item	Description
command mode	The **tvi** editor starts in command mode. Any subcommand can be called except those that only correct text during text input mode. To see a description of the subcommands, refer to the topics in Subcommands for the tvi Editor. To identify the subcommands that cannot be called from command mode, refer to Changing Text While in Input Mode. The **tvi** editor returns to command mode when subcommands and other modes end. Press the Esc key to cancel a partial subcommand.
text input mode	The **tvi** editor enters text input mode when you use a permitted command that adds or changes text. To see a list of subcommands that initiate text input mode, refer to Adding Text to a File and the subcommands that change text from command mode, the **C** subcommand and the ***cx*** subcommands. After entering one of these subcommands, you can edit text with any of the subcommands that function in text input mode. To see a description of the subcommands, refer to the topics in "Subcommands for the tvi Editor". To return to command mode from text input mode, press the Esc key for a typical exit or press the Ctrl+C keys to create an **INTERRUPT** signal.

Item	Description
last line mode	Some subcommands read input on a line displayed at the bottom of the screen. These subcommands include those with the prefix : (colon), / (slash), and ? (question mark). When you enter the initial character, the **tvi** editor places the cursor at the bottom of the screen so you can enter the remaining command characters. To run the subcommand, press the Enter key. To cancel the subcommand, press the Ctrl+C keys to create an **INTERRUPT** signal. When you use the : (colon) to enter last line mode, the following characters have special meaning when used before commands that specify counts: **%** All lines regardless of cursor position **$** Last line . Current line

Customizing the tvi Editor

You can customize the **tvi** editor on a temporary basis by following the directions in "Setting vi Editor Options". The section on "Setting vi Options Permanently" is *not* applicable to the **tvi** editor.

Subcommands for the tvi Editor

Information on **vi** editor subcommands that are applicable to the **tvi** editor is summarized in the following list:

- vi General Subcommand Syntax.
- vi Subcommands for Adjusting the Screen.
- Editing Text with the **vi** Editor.
- Entering Shell Commands within the **vi** Editor is *not* supported by the **tvi** editor.
- Manipulating Files with the **vi** Editor.
- Subcommands for Interrupting and Ending the **vi** Editor.

Flags

Item	Description
-c [*Subcommand***]**	Carries out the **ex** editor subcommand before editing begins. This provides a line-oriented text editor. When a null operand is entered for the *Subcommand* parameter, as in `-c ''`, the editor places the cursor on the last line of the file.
-l	Enters the editor in LISP mode. In this mode, the editor indents appropriately for LISP code, and the **(**, **)**, **{**, **}**, **[[**, and **]]** subcommands are modified to act appropriately for LISP. These subcommands place the cursor at the specified LISP function. For more information on the LISP subcommands, refer to Moving to Sentences, Paragraphs, and Sections.
-R	Sets the **readonly** option to protect the file against overwriting.
-w *Number*	Sets the default window size to the value specified by the *Number* parameter. This is useful when you use the editor over a low-speed line.
+ [*Subcommand***]**	Same as the **-c** Subcommand.

Security

Access Control: This command should grant execute (x) access to all users and have the **trusted computing base** attribute.

Auditing Events:

Event	Information
TVI	filename

Examples

1. To call a trusted editor to edit the `plans` file, type:

   ```
   tvi plans
   ```

   This command puts the **tvi** editor into command mode. To add or change text, you must enter text input mode or use a command accepted in command mode. For more information, refer to the description of text input mode.

2. To save the text you create with the **tvi** editor, leave text input mode by pressing the Esc key, and then enter one of the save commands **:w**, **:w** *File*, or **:w!** *File*, for example:

   ```
   :w plans
   ```

   In this example, a file name, such as `plans`, is needed if you gave the **tvi** command without specifying a file name. If the file is already named, the **:w** command would not need the *File* parameter. If you want to overwrite an existing file, use the **:w!** *File* command, specifying the file you want to overwrite with the *File* parameter.

   If you try to save an unnamed file without supplying a file name, the following message appears:

   ```
   No current filename
   ```

   If this happens, repeat the **:w** command with a file name.

3. To exit the **tvi** editor from text input mode, press the Esc key to type command mode, and then type:

   ```
   :q!
   ```

   If the editor already is in command mode, you do not need to press the Esc key before giving the quit (**q!**) command.

Files

Item	Description
/usr/bin/tvi	Contains the **tvi** command.

twconvdict Command

Purpose

Converts other user dictionary to the operating system user dictionary.

Syntax

twconvdict [**-i** *Type*] [**-v** *CodePage*] [**-f** *Source*] [**-t** *Target*]

Description

The **twconvdict** command converts a dictionary to an operating system user dictionary. The supported code pages are SOPS, PS55 and ET. The dictionary type include both Tseng_Jye and Phonetic user dictionaries.

Flags

Item	Description
-f *Source*	Specifies the name of font file to convert.
-i *Type*	Specifies the type of dictionary to convert to. *Type* can be: **TJ** Tseng_Jye, or **PH** Phonetic.
-t *Target*	Specifies the name of the converted font file.
-v *CodePage*	Specifies the type of code page to convert to. *CodePage* can be: **SOPS** **PS55**, or **ET**.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Auditing Events: N/A

Examples

To convert the dictionary USRFONT.C12 to an operating system dictionary of code page of type SOPS and dictionary type of Tseng_Jye with the name aix, enter:

```
twconvdict -i TJ -v SOPS -f USRFONT.C12 -t aix
```

Files

Item	Description
/usr/lpp/tls/bin/twconvdict	Contains the **twconvdict** command.

twconvfont Command

Purpose

Converts other font files to a BDF font file.

Syntax

twconvfont [**-v** *CodePage*] [**-f** *Source*] [**-t** *Target*]

Description

The twconvfont command converts one font file type to the BDF font file. The supported code pages are SOPS, PS55 and ET.

Flags

Item	Description
-f *Source*	Specifies the name of font file to convert.
-t *Target*	Specifies the name of the converted font file.
-v *CodePage*	Specifies the type of code page to convert to. *CodePage* can be: **SOPS** **PS55**, or **ET**.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Auditing Events: N/A

Examples

To convert the font file USRFONT.C12 to a BDF font file of code page of type SOPS with the name `user.bdf`, enter:

```
twconvfont -v SOPS -f USRFONT.C12 -t user.bdf
```

Files

Item	Description
/usr/lpp/tls/bin/twconvfont	Contains the **twconvfont** command.

type Command

Purpose

Writes a description of the command type.

Syntax

type *CommandName* ...

Description

The standard output of the **type** command contains information about the specified command and identifies whether this is a shell built-in command, subroutine, alias, or keyword. The **type** command indicates how the specified command would be interpreted if used. Where applicable, the **type** command displays the related path name.

Because the **type** command must know the contents of the current shell environment, it is provided as a Korn shell or POSIX shell regular built-in command. If the **type** command is called in a separate command execution environment, the command may not produce accurate results. This would be the case in the following examples:

```
nohup type writer
```

```
find . -type f | xargs type
```

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To learn whether the **cd** command is a base command or an alias or some other command type, enter:

   ```
   type cd
   ```

   The screen displays the following information:

   ```
   cd is a shell builtin
   ```

2. To see the location of the **find** command, enter:

   ```
   type find
   ```

   The screen displays the following information:

   ```
   find is /usr/bin/find
   ```

Files

Item	Description
/usr/bin/ksh	Contains the Korn shell **type** built-in command.

u

The following AIX commands begin with the letter *u*.

ucfgif Method

Purpose

Unloads an interface instance from the kernel.

Syntax

ucfgif [**-l** *InterfaceInstance*]

Description

The **ucfgif** method removes an interface instance from the kernel. To remove the interface instance, the **ucfgif** method does the following:

1. Unloads the interface software by calling the **/usr/sbin/ifconfig** interface detach.
2. Sets the status flag of the interface instance to **defined**.

> **Note:** The **ucfgif** method is a programming tool and should not be executed from the command line.

Flags

Item	Description
-l *InterfaceInstance*	Specifies the interface instance to be unconfigured. If no interface name is specified, all configured interface instances are unconfigured.

Example

To remove an interface instance from the kernel, enter the method in the following format:

```
ucfgif  -l tr0
```

In this example, the name of the interface instance is `tr0`.

ucfginet Method

Purpose

Unloads the Internet instance and all related interface instances from the kernel.

Syntax

ucfginet

Description

The **ucfginet** method unloads the Internet instance from the kernel. This subroutine also deletes the appropriate entries in the Address Family Domain switch table and in the Network Input Interface switch

table. The **ucfginet** method also sets the status flag of the instance to **defined**. The **ucfginet** method is called by the **rmdev** high-level command.

> **Note:** The **ucfginet** method is a programming tool and should not be executed from the command line.

ucfgqos Method

Purpose

Unconfigures and unloads the Quality of Service (QoS) instance from the kernel.

Syntax

ucfgqos

Description

The **ucfgqos** method disables Quality of Service (QoS) for the TCP/IP protocol suite on a host. This method detaches the QoS instance from the TCP/IP instance and unloads it from the kernel.

> **Note:** The **ucfgqos** method is a programming tool and is not intended to be invoked from the command line.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

To configure QoS on a host, use the following format:

```
ucfgqos
```

ucfgvsd Command

Purpose

ucfgvsd – Unconfigures a virtual shared disk.

Syntax

ucfgvsd {-a | *vsd_name* ...}

Description

The **ucfgvsd** command unconfigures the specified virtual shared disks. The specified virtual shared disks must be in the stopped state to be unconfigured. This command does not change any virtual shared disk definitions. It moves virtual shared disks from the stopped state to the defined state.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Unconfigure a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Flags

-a
Specifies that all virtual shared disks in the stopped state are to be unconfigured.

Parameters

vsd_name
Specifies a virtual shared disk. The disk specified must be in the stopped state. If all disks have been unconfigured, and you specify VSD0, this command will attempt to unload the device driver from the kernel.

Security

You must have root authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Standard Output

Current RVSD subsystem run level.

Examples

To unconfigure the virtual shared disk **vsd1vg1n1** in the stopped state, enter:

```
ucfgvsd vsd1vg1n1
```

Location

/opt/rsct/vsd/bin/ucfgvsd

uconvdef Command

Purpose

Compiles or generates a UCS-2 (Unicode) conversion table for use by the **iconv** library.

Syntax

uconvdef [**-f** *SrcFile*] [**-v**] *UconvTable*

Description

The **uconvdef** command reads *SrcFile* and creates a compiled conversion table in *UconvTable*. The *SrcFile* defines a mapping between UCS-2 and multibyte code sets (one or more bytes per character). The *UconvTable* is in a format that can be loaded by the UCSTBL conversion method located in the **/usr/lib/nls/loc/uconv** directory. This method uses the table to support UCS-2 conversions in both directions.

Flags

Item	Description
-f *SrcFile*	Specifies the conversion table source file. If this flag is not used, standard input is read.
-v	Causes output of the processed file statements.
UconvTable	Specifies the path name of the compiled table created by the **uconvdef** command. This should be the name of the code set that defines conversions into and out of UCS-2.

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

To access the compiled UCS-2 conversion table:

1. Create the compiled *UconvTable* using the name of the multibyte code set. For example, the conversion table between IBM-850 and UCS-2 can be compiled as follows:

   ```
   uconvdef -f IBM-850.ucmap IBM-850
   ```

2. Place the table in a directory called **uconvTable**. The default system directory is **/usr/lib/nls/loc/uconvTable**. If another directory is used, the **LOCPATH** environment variable needs to be set to include the parent directory (for example, **/usr/lib/nls/loc**).

   ```
   mv IBM-850 /usr/lib/nls/loc/uconvTable
   ```

3. Create symbolic links for conversions in each direction in a directory called **iconv**. The names for these links should be formed by concatenating the "From" code set and the "To" code set, separated by an underscore. The links should be set to point to the **/usr/lib/nls/loc/uconv/UCSTBL** conversion method. The default directory for these links is **/usr/lib/nls/loc/iconv**. If another directory is used, the **LOCPATH** environment variable needs to be set to include the parent directory (for example, **/usr/lib/nls/loc**).

   ```
   ln -s /usr/lib/nls/loc/uconv/UCSTBL \
   /usr/lib/nls/loc/iconv/IBM-850_UCS-2
   ```

   ```
   ln -s /usr/lib/nls/loc/uconv/UCSTBL \
   /usr/lib/nls/loc/iconv/UCS-2_IBM-850
   ```

   Note: The \ (backslash) is a line continuation character that is only needed if the command is broken into two lines.

udefif Method

Purpose

Removes an interface object from the system configuration database.

Syntax

udefif [**-l** *InterfaceInstance*]

Description

The **udefif** method deletes the specified interface instance from the system configuration database by:

1. Removing the database object associated with the interface instance.
2. Removing the connection and attribute information associated with the interface instance.

Flags

Item	Description
-l *InterfaceInstance*	Specifies the interface instance to be undefined. If no interface instances are specified, the **udefif** method undefines all defined interface instances.

Example

To remove an interface instance from the database, enter a method similar to the following:

```
udefif -l tr0
```

In this example, the interface instance to be removed is `tr0`.

udefinet Method

Purpose

Undefines the Internet instance in the configuration database.

Syntax

udefinet

Description

The **udefinet** method removes the database information associated with the Internet instance, including attribute information associated with the Internet instance.

> **Note:** The **udefinet** method is a programming tool and should not be executed from the command line.

udfcheck Command

Purpose

Performs a file system check on a UDF file system.

Syntax

udfcheck -d *device* [**-t** *tempfile*]

Description

The **udfcheck** command checks and repairs the UDF volume on a specified device.

Flags

Item	Description
-d *device*	Specifies the device on which **udfcheck** checks and repairs the UDF volume.
-t *tempfile*	Specifies a file where the **udfcheck** command stores information needed to perform a file system check.

Examples

1. To check the content of the UDF file system on device **/dev/cd1**, enter the following:

```
udfcheck -d /dev/cd1
```

Files

Item	Description
/usr/sbin/udfcheck	Contains the **udfcheck** command
/usr/lib/libudf.a	Contains the library routines called by the **udfcheck** command

udfcreate Command

Purpose

Creates the user defined functions (UDF) file systems.

Syntax

udfcreate -d *device* [**-b** *bitmap_location*] [**-f** *formatType*]

Description

The **udfcreate** command creates a UDF file system on the specified device and labels it with the generic set ID (*setID*) and volume name (*volName*).

Flags

Item	Description
-b *bitmap_location*	Specifies the location of the bitmap. It can be one of the following, **b**, **e**, or **m**. **b** indicates that the bitmap will be placed at the beginning of the partition. **e** indicates that the bitmap will be placed at the end of the partition. **m** indicates that the bitmap will be placed at the middle of the partition. The default location of the bitmap is the beginning of the partition.
-d *device*	Specifies the device on which to create the UDF volume.
-f *formatType*	Indicates the version of UDF to be present on the media. The format type of 1 represents UDF 1.5 version, 2 represents UDF 2.0 version, and 3 represents UDF 2.01 version. The default version is UDF 1.5.
-s 2048	Forces the newly created UDF filesystem to use 2048 byte logical blocks.

Examples

1. To create a new UDF file system on device **/dev/cd1**, enter the following command:

```
udfcreate -d /dev/cd1
```

Files

Item	Description
/usr/sbin/udfcreate	Contains the **udfcreate** command
/usr/lib/libudf.a	Contains the library routines called by the **udfcreate** command

udflabel Command

Purpose

Fetches and changes the label on a UDF file system.

Syntax

udflabel **-d** *device* [**-l** *label*]

Description

The **udflabel** command displays or changes a UDF volume name. If there is no label provided, it displays the current UDF volume name on the device. If there is a label provided, it sets the current UDF volume name on the device to the new label.

Flags

Item	Description
-d *device*	Specifies the device containing the UDF volume.

Item	Description
-l *label*	Sets the label on the current UDF volume.

Examples

1. To change the current label on device **/dev/cd1** to hello, enter the following command:

```
udflabel -d /dev/cd1 -l hello
```

2. To display the current label on device **/dev/cd1**, enter the following command:

```
udflabel -d /dev/cd1
```

Files

Item	Description
/usr/sbin/udflabel	Contains the **udflabel** command
/usr/lib/libudf.a	Contains the library routines called by the **udflabel** command

uil Command

Purpose

Starts the User Interface Language (UIL) compiler for the AIXwindows system.

Syntax

uil [**-I***PathName*] *InputFile* [**-m**] [**-o** *FileName*] [**-s**] [**-v** *FileName*] [**-w**] [**-wmd** *FileName*]

Description

The **uil** command calls the UIL compiler. The UIL is a specification language for describing the initial state of a user interface for an AIXwindows application. The specification describes the objects (menus, dialog boxes, labels, push buttons, and so on) used in the interface and specifies the functions to be called when the interface changes state as a result of user interaction.

Flags

Item	Description
-I*PathName*	Specifies **I**include*PathName* with no spaces. Causes the compiler to look for include files in a specified directory if include files were not found in the default paths. (uppercase i)
-m	Specifies that machine code is listed. This directs the compiler to place a description of the records that it added to the User Interface Definition (UID) in the listing file. This helps you isolate errors. The default is no machine code.
-o *FileName*	Directs the compiler to produce a UID. By default, UIL creates a UID with the name **a.uid**. The file specifies the file name for the UID. No UID is produced if the compiler issues any diagnostics categorized as error or severe.
-s	Directs the compiler to set the locale before compiling any files. The locale is set in an implementation-dependent manner. On ANSI C-based systems, the locale is usually set by calling the **setlocale (LC_ALL, "")** function. If this option is not specified, the compiler does not set the locale.

Item	Description
-v *FileName*	Directs the compiler to generate a listing. The file specifies the file name for the listing. If the **-v** option is not present, no listing is generated by the compiler. The default is no listing.
-w	Specifies that the compiler suppress all warning and informational messages. If this option is not present, all messages are generated, regardless of the severity.
-wmd *FileName*	Specifies a binary widget meta-language (WML) description file to be used instead of the default WML description.

Example

To start the UIL compiler, enter:

```
uil -I. -o ex.uid ex.uil
```

Exit Status

This command returns the following exit values:

Item	Description
0	Indicates successful completion.
>0	Indicates an error occurred.

uimx Command

Purpose

Starts the UIM/X user-interface management system for the X Window System.

Syntax

uimx [**-dir** *Path* **] [** **-file** *FileName*] **[** **-workspace** *Name* **] [** **-xrm** *Options* **]**

Description

The **uimx** command starts the UIM/X user-interface management system for the X Window System. It supports Motif 1.2 and provides a complete programming environment for developing graphical user interfaces (GUIs). UIM/X supports object-oriented programming in both C and C++.

UIM/X saves and loads text files that use the Xt resource syntax to describe interfaces and projects. It can also load UIL files. It generates C, C++, and UIL code. It can also generate makefiles, message catalogs, and resource files for an application.

UIM/X includes a built-in C Interpreter and the following tools and editors:

- Palette of Motif widgets
- Widget Browser for browsing complex widget hierarchies
- WYSIWYG layout editor for drawing interfaces
- Property Editor for setting initial values of widget properties; initial values can be literal values or C expressions
- Callback Editors for entering callback code
- Event, Action, and Translation Editors

- Menu and Main Window Editors
- Declarations Editor for editing the generated code for an interface
- Program Layout Editor for editing the generated main program and makefile; this editor gives you direct access to the main event loop

UIM/X supports two operating modes: Design and Test. In Test mode, the built-in C Interpreter allows you to test the behavior of your application. In Design mode, the C Interpreter validates the code you enter into the various UIM/X editors.

UIM/X provides a convenience library of functions that simplify the task of programming with X and Motif.

Flags

Item	Description
dir *Path*	Sets UIM/X's current directory to path.
file *FileName*	Loads an existing project, interface, or palette file called *FileName*. *FileName* can include an absolute path name, a path name relative to the current directory, or a path name relative to the **-dir** value.
workspace *Name*	Loads UIM/X into the corresponding CDE workspace called *name*.
xrm *Options*	Enables you to enter any resource specifications (*options*) that you would otherwise put in a resource file.

Security

Access Control: Any User

Files Accessed: None

Example

To start UIM/X, enter:

```
uimx
```

Files

Item	Description
/usr/uimx2.8/bin/uimx	Contains the **uimx** command.

ul Command

Purpose

Performs underlining.

Syntax

ul [**-i**] [**-t** *Terminal*] [*File* ...]

Description

The **ul** command reads the named files specified by the *File* parameter (or standard input if no file is given) and translates occurrences of underscores to the sequence that indicates underlining for the terminal in use, as specified by the **TERM** environment variable.

Flags

Item	Description
-i	Causes the **ul** command to indicate underlining by a separate line containing appropriate _ (underline characters). Use this to see the underlining present in an **nroff** command output stream on a CRT terminal.
-t *Terminal*	Overrides the terminal type specified in the environment. The **terminfo** file is read to determine the appropriate sequences for underlining. If the terminal is incapable of underlining, but is capable of a standout mode, then that mode is used instead. If the terminal can overstrike or automatically underline, the **ul** command acts like the **cat** command and displays on the screen. If the terminal cannot underline and no alternatives are available, underlining is ignored. If the **-t** flag is not specified, the **ul** command translates for the terminal type specified by the **TERM** environment variable. If the value of the *Terminal* variable is not a valid terminal type, the **ul** command translates for a dumb terminal.

Files

Item	Description
/usr/share/lib/terminfo/*	Contains the terminal capabilities database.

ulimit Command

Purpose

Sets or reports user resource limits.

Syntax

ulimit [**-H**] [**-S**] [**-a**] [**-c**] [**-d**] [**-f**] [**-m**] [**-n**] [**-r**] [**-s**] [**-t**] [-u][*Limit*]

Description

The **ulimit** command sets or reports user process resource limits, as defined in the **/etc/security/limits** file. This file contains these default limits:

```
fsize = 2097151
core = 2097151
cpu = -1
data = 262144
rss = 65536
stack = 65536
nofiles = 2000
threads = -1
nproc = -1
```

These values are used as default settings when a new user is added to the system. The values are set with the **mkuser** command when the user is added to the system, or changed with the **chuser** command.

Limits are categorized as either soft or hard. With the **ulimit** command, you can change your soft limits, up to the maximum set by the hard limits. You must have root user authority to change resource hard limits.

Many systems do not contain one or more of these limits. The limit for a specified resource is set when the *Limit* parameter is specified. The value of the *Limit* parameter can be a number in the unit specified with each resource, or the value `unlimited`. To set the specific ulimit to `unlimited`, use the word `unlimited`

Note: Setting the default limits in the **/etc/security/limits** file sets system wide limits, not just limits taken on by a user when that user is created.

The current resource limit is printed when you omit the *Limit* parameter. The soft limit is printed unless you specify the **-H** flag. When you specify more than one resource, the limit name and unit is printed before the value. If no option is given, the **-f** flag is assumed.

Since the **ulimit** command affects the current shell environment, it is provided as a shell regular built-in command. If this command is called in a separate command execution environment, it does not affect the file size limit of the caller's environment. This would be the case in the following examples:

```
nohup ulimit -f 10000
env ulimit 10000
```

Once a hard limit has been decreased by a process, it cannot be increased without root privilege, even to revert to the original limit.

Flags

Item	Description
-a	Lists all of the current resource limits.
-c	Specifies the size of core dumps, in number of 512-byte blocks.
-d	Specifies the size of the data area, in number of K bytes.
-f	Sets the file size limit in blocks when the *Limit* parameter is used, or reports the file size limit if no parameter is specified. The **-f** flag is the default.
-H	Specifies that the hard limit for the given resource is set. If you have root user authority, you can increase the hard limit. Anyone can decrease it.
-m	Specifies the size of physical memory (resident set size), in number of K bytes. This limit is not enforced by the system.
-n	Specifies the limit on the number of file descriptors a process may have.
`-r`	Specifies the limit on the number of threads a process can have.
-s	Specifies the stack size, in number of K bytes.
-S	Specifies that the soft limit for the given resource is set. A soft limit can be increased up to the value of the hard limit. If neither the **-H** nor **-S** flags are specified, the limit applies to both.
-t	Specifies the number of seconds to be used by each process.
-u	Specifies the limit on the number of a process a user can create.

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	A request for a higher limit was rejected or an error occurred.

Example

To set the file size limit to 51,200 bytes, enter:

```
ulimit -f 100
```

To list all the current resource limits, enter:

```
ulimit -a

time(seconds)        unlimited
file(blocks)         2097151
data(kbytes)         131072
stack(kbytes)        32768
memory(kbytes)       65536
coredump(blocks)     2097151
nofiles(descriptors) 2000
threads(per process) unlimited
processes(per user)  unlimited
```

Files

Item	Description
/usr/bin/ksh	Contains the **ulimit** built-in command.

umask Command

Purpose

Displays or sets the file mode creation mask.

Syntax

umask [**-S**] [*Mask*]

Description

If the *Mask* parameter is not specified, the **umask** command displays to standard output the file mode creation mask of the current shell environment. If you specify the *Mask* parameter using a three-digit octal number or symbolic code, the **umask** command sets the file creation mask of the current shell execution environment. The bits set in the file creation mask are used to clear the corresponding bits requested by an application or command when creating a file.

The **chmod** command describes how to use the symbolic and numeric codes to set permissions.

The **-S** flag produces symbolic output. If the flag is not specified, the default output format is octal.

If the **/usr/bin/umask** command is called in a subshell or separate command execution environment, it does not affect the file mode creation mask of the caller's environment. This would be the case in the following example:

```
(umask 002)
```

```
nohup umask ...
```

```
find . -exec umask ... \;
```

Flags

Item	Description
-S	Produces symbolic output.

Exit Status

The following exit values are returned:

Item	Description
0	The file mode creation mask was successfully changed, or no *Mask* parameter was supplied.
>0	An error occurred.

Examples

1. To set the mode mask so that subsequently created files have their **S_IWOTH** bit cleared, enter either:

   ```
   umask a=rx,ug+w
   ```

   OR

   ```
   umask 002
   ```

   After setting the mode mask, display the current values of the mode mask by entering:

   ```
   umask
   ```

   The screen displays the following value:

   ```
   02
   ```

2. To produce symbolic output, enter:

   ```
   umask -S
   ```

   The screen displays the following values:

   ```
   u=rwx,g=rwx,o=rx
   ```

3. Either numeric or symbol output can be used as the *Mask* parameter to a subsequent invocation of the **umask** command. Assume the mode mask is set as shown in example 2. To set the mode mask so that subsequently created files have their **S_IWGRP** and **S_IWOTH** bits cleared, enter:

   ```
   umask g-w
   ```

4. To set the mode mask so that subsequently created files have all their write bits cleared, enter:

   ```
   umask -- -w
   ```

   Note: The **-r**, **-w**, and **-x** *Mask* parameter values (or anything beginning with a hyphen) must be preceded by — (double hyphen, no space between) to keep it from being interpreted as an option.

Files

Item	Description
/usr/bin/ksh	Contains the Korn shell **umask** built-in command.
/usr/bin/umask	Contains the **umask** command.

umcode_latest Command

Purpose

Identifies system resources with firmware or microcode that can be updated from a specified source of image files.

Syntax

`umcode_latest` [`-s` *source*] [`-l`] [`-A`] | [`-a`[`-q`][`-r`]| `-i`] | `-h`

Description

The `umcode_latest` command lists or downloads the system resources that have an older firmware or microcode level than the firmware or microcode level that was found on the specified source for those system resources.

Note: System Firmware images of system types 8842/8844/7047/7013/7015/7017 and 7025-F50 are not supported by this command. For systems with temporary and permanent system firmware images, the `umcode_latest` command uses the temporary system firmware image for comparisons with the images on the specified source. System firmware image file names must end with `.img`.

Flags

Item	Description
`-a`	Updates all system resources that have newer microcode on the source.
`-A`	Lists or updates resource when any of the images on the source is different from the image currently listed or updated. The default is to list or update whenever the source has a newer image.
`-h`	Provides extended usage help.
`-i`	Provides an interactive mode so that each resource that needs an update is prompted.
`-l`	Lists the system resources that need updates. This is the default.
`-q`	Refrains from asking whether to proceed with the update all.
`-r`	Refrains from asking whether to proceed with the update requiring a system IPL.
`-s` *source*	Points to the source of the microcode image. The default is `/etc/microcode`.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Examples

1. To list all system resources with firmware or microcode that can be updated from the images in `/etc/microcode`, enter:

```
/usr/lpp/diagnostics/bin/umcode_latest
```

2. To list all system resources with firmware or microcode that can be updated from the images that are in the `/tmp/fwupdate` directory, enter:

```
/usr/lpp/diagnostics/bin/umcode_latest -s /tmp/fwupdate
```

3. To list all system resources with firmware or microcode that can be updated from the images that are in the `/tmp/fwupdate` directory, and for each resource ask whether the resource should be updated at this time, enter:

```
/usr/lpp/diagnostics/bin/umcode_latest -s /fwupdate -i
```

4. To automatically update all of the system resources with firmware or microcode that have newer images on the ISO 9660 format CD-ROM, which has already been inserted into the cd1 drive, enter:

```
/usr/lpp/diagnostics/bin/umcode_latest -s cd1 -a -q
```

Restrictions

System Firmware images of system types 8842/8844/7047/7013/7015/7017 and 7025-F50 are not supported by this command. For systems with temporary and permanent system firmware images, the `umcode_latest` command uses the temporary system firmware image for comparisons with the images on the specified source. System firmware image file names must end with `.img`.

Location

`/usr/lpp/diagnostics/bin/umcode_latest`

umount or unmount Command

Purpose

Unmounts a previously mounted file system, directory, or file.

Syntax

{ **unmount** | **umount** } [**-f**] [**-a**] | [**all** | **allr** | *Device* | *Directory* | *File* | *FileSystem* | **-n** *Node* | **-t** *Type*]

Description

Another name for the **umount** command is the **unmount** command. Either name can be used. You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit umount
```

The **umount** command unmounts a previously mounted device, directory, file, or file system. Processing on the file system, directory, or file completes and it is unmounted. Members of the system group and users operating with root user authority can issue any **umount** command. Only users with root authority or are members of the system group can unmount a directory or file.

> **Note:** SMIT will not unmount the **/usr/lpp/info/$LANG** directory, the directory on which SMIT helps are located. Typically, this is the CD-ROM.

To unmount local mounts you can specify the device, directory, file, or file system on which it is mounted.

If the file system being unmounted is a JFS2 snapshot, the **umount** command will unmount the snapshot, though the snapshot will still be active. The **snapshot** command must be used to delete the snapshot.

If the file system being unmounted is a snapped file system with mounted snapshots, the **umount** command displays a warning that there are mounted snapshots and exits without unmounting the file system. The snapshots must be unmounted first.

Note: If the **cdromd** CD and DVD automount daemon is enabled, then those devices will be automatically mounted as specified in the **/etc/cdromd.conf** file. Use the **cdumount** or **cdeject** command to unmount an automounted CD or DVD. Use "**stopsrc -s cdromd**" to disable the CD/DVD automount daemon.

Flags

Item	Description
-a	Unmounts all mounted file systems.
all	Unmounts all mounted file systems.
allr	Unmounts all remotely mounted file systems. **Note:** For remote mounts, specify the device, directory, file, or file system parameters. If you specify the **allr** flag, the **umount** command unmounts all remote mounts.
-f	For remote mounted file systems, the **–f** flag forces an unmount to free a client when the server is down and server path names cannot be resolved, or when a file system must be unmounted while it is still in use. **Note:** For remote file systems, using this flag causes all file operations on the file system except **close()** and **unmap()** to fail. Any file data that has been written by an application but has not yet transferred to the server will be lost. A forced unmount of an NFS version 4 file system can cause open file state for other file systems mounted from the same server to be lost as well. For local JFS2 file systems, the **–f** flag forces an unmount when a file system must be unmounted while it is still in use. **Note:** You can use the **-f** flag only in JFS2 file systems, not in other journaled file systems. The following restrictions are applied on a forced unmount of a JFS2 file system: • The **–f** flag cannot force an unmount of a file system if a subdirectory or a file is overmounted on the file system. • The**–f** flag cannot force an unmount of a file system with mounted or open external snapshots until those snapshots are forced unmounted.
-n *Node*	Specifies the node holding the mounted directory you want to unmount. The **umount -n** *Node* command unmounts all remote mounts made from the *Node* parameter.
-t *Type*	Unmounts all stanzas in the **/etc/filesystems** file that contain the **type=***Type* flag and are mounted. The *Type* parameter is a string value, such as the remote value that specifies the name of the group.

Note: You cannot use the **umount** command on a device in use. A device is in use if any file is open for any reason or if a user's current directory is on that device.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To unmount all mounts from remote node Node A, enter:

   ```
   umount -n nodeA
   ```

2. To unmount files and directories of a specific type, enter:

   ```
   umount -t test
   ```

   This unmounts all files or directories that have a stanza in the **/etc/filesystems** file that contains the **type=test** attribute.

Files

Item	Description
/etc/filesystems	Lists the known file systems and defines their characteristics.

umountall Command

Purpose

Unmounts groups of dismountable devices or filesystems.

Syntax

umountall [**-k**] [**-s**] [**-F** *FileSytemType*] [**-l** | **-r**]

umountall [**-k**] [**-s**] [**-h** *Host*]

Description

The **umountall** command by default unmounts all dismountable file systems or devices except root, /proc, /var and /usr. If the *FileSystemType* is specified, **umountall** limits its actions to the file system type specified. There is no guarantee that **umountall** will unmount busy file systems, even if the **-k** option is specified.

Flags

Item	Description
-F *FileSystemType*	Specifies the type of file systems to be dismounted. *FileSystemType* corresponds to the vfs column printed out by the mount command. All dismountable file systems of the given type will be unmounted. This flag cannot be used in combination with the **-h** flag.
-h *Host*	Specifies the host node. All file systems remotely mounted from this host will be unmounted.

Item	Description
-k	Sends a **SIGKILL** to each process on the mount point before unmounting. This option internally uses the **fuser -k** command to kill all the processes running on the mount point. As this option causes each process on the mount point to be killed, the unmount of the mount point does not happen immediately. There is no guarantee that **umountall** will unmount busy file systems, even if the **-k** option is specified. An attempt to unmount the mount point will be made only after all the processes using the mount point are killed.
-l	Limits the action to local filesystems.
-r	Limits the action to remote filesystems.
-s	This is a no-operation flag provided for System V compatibility on serializing the **unmounts**. The serialization of the **unmount** command is done using **-k** option by terminating all the associated processes on the mount point.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To unmount all dismountable file systems, enter:

```
umountall
```

2. To unmount all dismountable filesystems of type **jfs**, enter:

```
umountall -F jfs
```

3. To unmount all dismountable filesystems mounted from host.domain, enter:

```
umountall -h host.domain
```

4. To unmount all remotely mounted filesystems, enter:

```
umountall -r
```

Files

Item	Description
/usr/sbin/umountall	Contains the **umountall** command.

unalias Command

Purpose

Removes alias definitions.

Syntax

unalias **-a**

unalias *AliasName ...*

Description

The **unalias** command removes the definition for each alias name specified, or removes all alias definitions if the **-a** flag is used. Alias definitions are removed from the current shell environment.

Since the **unalias** command affects the current shell execution environment, it is provided as a Korn shell or POSIX shell regular built-in command.

Flags

Item	Description
-a	Removes all alias definitions from the current shell environment.

Exit Status

The following exit values are returned:

Item	Description
0	Successful completion.
>0	One of the alias names specified did not represent a valid alias definition, or an error occurred.

Files

Item	Description
/usr/bin/ksh	Contains the Korn shell **unalias** built-in command.
/usr/bin/unalias	Contains the **unalias** command.

uname Command

Purpose

Displays the name of the current operating system.

Syntax

uname [**-a** | **-x** | **-S** *Name*] [**-F**] [**-f**] [**-l**] [**-L**] [**-m**] [**-M**] [**-n**] [**-p**] [**-r**] [**-s | V**] [**-T** *Name*] [**-u**] [**-v**] [**-W**]

Description

The **uname** command writes to standard output the name of the operating system that you are using.

The machine ID number contains 12 characters in the following digit format: *xxyyyyyymmss*. The *xx* positions indicate the system and is always 00. The *yyyyyy* positions contain the unique ID number for the entire system. The *mm* position represents the model ID. The *ss* position is the submodel number and is always 00. The model ID describes the ID of the CPU Planar, not the model of the System as a whole.

Most machines share a common model ID of 4C.

The machine identifier value returned by the **uname** command may change when new operating system software levels are installed. This change affects applications using this value to access licensed programs. To view this identifier, enter the **uname -m** command.

Contact the appropriate support organization if your application is affected.

Flags

Item	Description
-a	Displays all information specified with the **-m**, **-n**, **-r**, **-s**, and **-v** flags. Cannot be used with the **-x** or **-S***Name* flag. If the **-x** flag is specified with the **-a** flag, the **-x** flag overrides it.
-F	Displays a system identification string comprised of hexadecimal characters. This identification string is the same for all partitions on a particular system.
-f	Similar to the **F** flag, except that the partition number is also used in the calculation of this string. The resulting identification string is unique for each partition on a particular system.
-l	Displays the LAN network number.
-L	Displays LPAR number and LPAR name. If LPAR does not exist, -1 is displayed for LPAR number and NULL for LPAR name. If a system is capable of LPAR, but is currently running in Symmetric Multi Processing (SMP) mode, 1 is displayed for LPAR number and NULL for LPAR name.
-m	Displays the machine ID number of the hardware running the system. **Note:** The **-m** flag cannot be used to generate a unique machine identifier for partitions in an LPAR environment.
-M	Displays the system model name. If the model name attribute does not exist, a null string is displayed.
-n	Displays the name of the node. This may be a name the system is known by to a UUCP communications network.
-p	Displays the architecture of the system processor.
-r	Displays the release number of the operating system.
-s	Displays the system name. This flag is on by default. The **-s** option is mutually exclusive with the **-V** option.
-V	Displays the VIOS Complete version detail if ran in a LPAR that contains VIOS, else displays details of the AIX operating system. The **–V** option is mutually exclusive with the **-s** option.
-S *Name*	Sets the name of the node. This can be the UUCP communications network name for the system.
-T *Name*	Sets the system name. This can be the UUCP communications network name for the system.
-u	Displays the system ID number. If this attribute is not defined, the output is the same as the output displayed by **uname -m**.

Item	Description
-v	Displays the operating system version.
-W	Displays the static workload partition identification number. If the uname command runs in the Global environment, a value of zero is displayed.
-x	Displays the information specified with the **-a** flag as well as the LAN network number, as specified by the **-l** flag.

If you enter a flag that is not valid, the **uname** command exits with an error message, an error return status, and no output.

> **Note:** The uname command does not preserve the new system name and node name values across system reboot.

Exit Status

This command returns the following exit values:

Item	Description
0	The requested information was successfully written.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

To display the complete system name and version banner, enter:

```
uname -a
```

Files

Item	Description
/usr/bin/uname	Contains the **uname** command.

uncompress Command

Purpose

Restores compressed files.

Syntax

uncompress [**-c**] [**-F**] [**-f**] [**-n**] [**-q**] [**-V**] [*File ...*]

Description

The **uncompress** command restores original files that were compressed by the **compress** command. Each compressed file specified by the *File* parameter is removed and replaced by an expanded copy. The expanded file has the same name as the compressed version, but without the **.Z** extension. If the user

has root authority, the expanded file retains the same owner, group, modes, and modification time as the original file. If the user does not have root authority, the file retains the same modes and modification time, but acquires a new owner and group. If no files are specified, standard input is expanded to standard output.

Flags

Item	Description
-c	Write to standard output. No files are changed.
-f or **-F**	Forces expansion. The **-f** and **-F** flags are interchangeable. Overwrites the file if it already exists. The system does not prompt the user that an existing file will be overwritten. File size may not actually shrink.
-n	Omits the compressed file header from the compressed file. **Note:** Use this option if the file was compressed using the **-n** flag. Otherwise, uncompressing the file will not work.
-q	Suppresses the display of compression statistics generated by the **-v** flag. If several **-v** and **-q** flags are on the same command line, the last one specified controls the display of the statistics.
-V	Writes the current version and compile options to standard error.

Parameters

Item	Description
File ...	Specifies the compressed files to restore.

Return Values

The **uncompress** command detects an error and exit with a status of 1 if any of the following events occur:

- The input file was not produced by the **compress** command.
- An input file cannot be read or an output file cannot be written.

If no error occurs, the exit status is 0.

Exit Status

Item	Description
0	Successful completion.
>0	An error occurred.

Example

To uncompress the `foo.Z` file, enter:

```
uncompress foo.Z
```

The `foo.Z` file is uncompressed and renamed `foo`.

undefvsd Command

Purpose

undefvsd – Undefines a virtual shared disk.

Syntax

undefvsd *vsd_name* ...

Description

This command is used to remove a virtual shared disk definition and any special device files from **/dev** for the given *vsd_names* on all the virtual shared disk nodes. The virtual shared disks must be unconfigured and in the defined state on all the virtual shared disk nodes.

You can use the System Management Interface Tool (SMIT) to run the **undefvsd** command. To use SMIT, enter:

```
smit delete_vsd
```

and select the **Undefine a Virtual Shared Disk** option.

Flags

None.

Parameters

vsd_name
Specifies the virtual shared disk whose underlying logical volume you no longer want to be globally accessed by any virtual shared disk nodes.

Security

You must have root authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

To delete the information associated with the virtual shared disk **vsd1vg2n1**, enter:

```
undefvsd vsd1vg2n1
```

Location

/usr/lpp/vsd/bin/undefvsd

unexpand Command

Purpose

Writes to standard output with tabs restored.

Syntax

unexpand [**-a** | **-t** *TabList*] [*File ...*]

Description

The **unexpand** command puts tabs back into the data from the standard input or the named files and writes the result to standard output. By default, only leading spaces and tabs are reconverted to maximal strings of tabs.

Note: The *File* parameter must be a text file.

Flags

Item	Description
-a	Inserts tabs wherever their presence compresses the resultant file by replacing two or more characters.
-t *TabList*	Specifies the position of the tab stops. The default value of a tab stop is 8 column positions. The *TabList* variable must consist of a single positive-decimal integer or multiple positive-decimal integers. The multiple integers must be in ascending order and must be separated by commas or by blank characters with quotation marks around the integers. The single *TabList* variable sets the tab stops an equal number of column positions apart. The multiple *TabList* variable sets the tab stop at column positions that correspond to the integers in the *TabList* variable. A space-to-tab conversion does not occur for characters at positions beyond the last one specified in a multiple *TabList* variable.

Note: When the **-t** flag is specified, the **-a** flag is ignored and conversion is not limited to processing leading blank characters.

Exit Status

This command returns the following exit values:

Item	Description
0	The command ran successfully.

Item	Description
>0	An error occurred.

Example

To replace space characters with tab characters in the **xyz** file, enter:

```
unexpand xyz
```

Files

Item	Description
/usr/bin/unexpand	Contains the **unexpand** command.

unfencevsd Command

Purpose

unfencevsd – Gives applications running on a node or group of nodes access to a virtual shared disk or group of virtual shared disks that were previously fenced from applications running on those nodes.

Syntax

unfencevsd {-a | -v *vsd_name_list*} {-n *node_list* [-f] }

Description

Under some circumstances, the system may believe a node has become inoperable and may begin recovery procedures when the node is actually operational, but is cut off from communication with other nodes running the same application. In this case, the problem node must not be allowed to serve requests for the virtual shared disks it normally manages until recovery is complete and the other nodes running the application recognize the problem node as operational. The fencevsd command prevents the problem node from filling requests for its virtual shared disks. The unfencevsd command allows fenced nodes to regain access to the virtual shared disks.

You can issue this command from any node that is online in the peer domain.

Flags

-a
Specifies all virtual shared disks.

-f
Allows a fenced node to unfence itself.

-n *node_list*
Specifies one or more node numbers separated by commas.

-v *vsd_name_list*
Specifies one or more virtual shared disk names, separated by commas.

Parameters

None.

Security

You must have root authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

1. To unfence node 5 from the virtual shared disks vsd1 and vsd2, enter:

```
unfencevsd -v vsd1,vsd2 -n 5
```

2. To unfence node 7 from the virtual shared disks vsd1 and vsd2 when the `unfencevsd` command must be entered from node 7, enter:

```
unfencevsd -v vsd1,vsd2 -n 7 -f
```

Location

/opt/rsct/vsd/bin/unfencevsd

unget Command (SCCS)

Purpose

Cancels a previous **get** command.

Syntax

unget [**-r***SID*] [**-s**] [**-n**] *File* ...

Description

The **unget** command allows you to restore a g-file created with **get -e** before the new delta is created. Any changes are therefore discarded. If you specify a **-** (dash) for the value of *File*, standard input is read, and each line of standard input is interpreted as the name of an SCCS file. An end-of-file character terminates input.

If you specify a directory for the *File* value, the **unget** command performs the requested actions on all SCCS files that are currently in the process of being edited (those files with the **s.** prefix).

Once you have run an **unget** command on a file, you must reissue a **get -e** command to make changes to the file. The **unget** command automatically deletes the g-file.

Flags

Each flag or group of flags applies independently to each named file.

Item	Description
-n	Prevents the automatic deletion of the g-file. This flag allows you to retain the edited version of the file without making a delta.
-r*SID*	Specifies the new delta that would have been created by the next use of the **delta** command. You must use this flag if you have two or more pending deltas to the file under the same login name. You can look at the p-file to see if you have more than one delta pending to a particular SID under the same login name. The *SID* specification must unambiguously specify only one SID to discard, or the **unget** command displays an error message and stops running.
-s	Suppresses displaying the deleted SID.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Example

To discard the changes you have made to an SCCS file after running a **get -e** command, enter:

```
unget s.prog.c
```

Files

Item	Description
/usr/bin/unget	Contains the path to the SCCS **unget** command.

unifdef Command

Purpose

Removes ifdef lines from a file.

Syntax

unifdef [**-t**] [**-l**] [**-c**] [**-D***Symbol*] [**-U***Symbol*] [**-id***Symbol*] [**-iu***Symbol*] [*File*]

Description

The **unifdef** command is useful for removing ifdef lines from a file while otherwise leaving the file alone. The **unifdef** command recognizes nested ifdefs, comments, and single and double quotes of C syntax in order to function correctly, but does not include files or interpret macros. The **unifdef** command recognizes but does not remove comments.

The **unifdef** command takes its input from standard input if no *File* is specified and copies its output to standard output.

Once a *Symbol* is specified, the lines inside those ifdefs are copied to the output or removed, as appropriate. The ifdef, ifndef, else, elif, and endif lines associated with the symbol are also removed. Ifdefs that involve unspecified symbols are untouched and copied out along with their associated ifdef, else, elif, and endif lines. If the same symbol appears in more than one argument, only the first occurrence is significant. For instance, if an ifdef X occurs nested inside another ifdef X, the inside ifdef is considered an unrecognized symbol.

When using ifdefs to delimit non-C lines such as comments or unfinished code, it is necessary to specify which symbols are to be used for that purpose. Otherwise, the **unifdef** command will try to parse for quotes and comments in those ifdef lines.

The **unifdef** command cannot process **cpp** constructs such as:

```
#if defined(X) || defined(Y)
```

OR

```
#elif X
```

OR

```
#elif defined(X) || defined(Y)
```

Keywords

The following keywords are recognized by the **unifdef** command:

- **ifdef**
- **ifndef**
- **else**
- **endif**
- **elif**

Flags

Item	Description
-c	Complements the operation of the **unifdef** command. That is, the lines which would have been removed are retained and vice versa.
-D *Symbol*	Specifies the symbol to be defined.
File	Specifies the input source.
-id *Symbol*	The **unifdef** command will not try to recognize comments, single quotes, or double quotes inside specified **ifdefs**, but these lines will be copied out.
-iu *Symbol*	The **unifdef** command will not try to recognize comments, single quotes, or double quotes inside specified **ifdefs**. These lines will not be copied out.
-l	Causes removed lines to be replaced with blank lines instead of being deleted.
-t	Allows the **unifdef** command to be used for plain text (instead of C code): the **unifdef** command will not try to recognize comments, single quotes and double quotes.
-U *Symbol*	Specifies the symbol to be undefined.

Exit Status

This command returns the following exit values:

Item	Description
0	The output is an exact copy of the input.
1	The output is not an exact copy of the input.
2	The command failed due to a premature EOF, or to an inappropriate **else**, **elif**, or **endif**.

Examples

1. The following example:

   ```
   unifdef -DA original.c > modified.c
   ```

   causes the **unifdef** command to read the file `original.c`, and remove the #ifdef A lines. It then removes everything following an #elif/#else associated with the #ifdef A, down to the #endif. The output is placed in the `modified.c` file.
2. The following example:

   ```
   unifdef -UA original.c > modified.c
   ```

   causes the **unifdef** command to read the file `original.c`, and remove the #ifdef A down to either its associated #elif//#else, or its associated #endif. In the case of the #elif, the #elif is replaced with #if. In the case of #else, the #else is deleted along with its associated #endif. The output is placed in the `modified.c` file.

Files

Item	Description
/usr/bin/unifdef	Contains the **unifdef** command.

uniq Command

Purpose

Reports or deletes repeated lines in a file.

Syntax

uniq [**-c** | **-d** | **-u**] [**-f** *Fields*] [**-s** *Characters*] [**-***Fields*] [**+***Characters*] [*InFile* [*OutFile*]]

Description

The **uniq** command deletes repeated lines in a file. The **uniq** command reads either standard input or a file specified by the *InFile* parameter. The command first compares adjacent lines and then removes the second and succeeding duplications of a line. Duplicated lines must be adjacent. (Before issuing the **uniq** command, use the **sort** command to make all duplicate lines adjacent.) Finally, the **uniq** command writes the resultant unique lines either to standard output or to the file specified by the *OutFile* parameter. The *InFile* and *OutFile* parameters must specify different files.

The input file must be a text file. A *text* file is a file that contains characters organized into one or more lines. The lines can neither exceed 2048 bytes in length (including any newline characters) nor contain null characters.

The **uniq** command compares entire lines by default. If the **-f** *Fields* or **-***Fields* flag is specified, the **uniq** command ignores the number of fields specified by the *Fields* variable. A *field* is a string of characters separated from other character strings by one or more <blank> characters. If the **-s** *Characters* or **-***Characters* flag is specified, the **uniq** command ignores the number of characters specified by the

Characters variable. Values specified for the *Fields* and *Characters* variables must be positive decimal integers.

The current national language environment determines the <blank> characters used by the **-f** flag as well as how the **-s** flag interprets bytes as a character.

The **uniq** command exits with a value of 0 if successful. Otherwise, it exits with a value greater than 0.

Flags

Item	Description
-c	Precedes each output line with a count of the number of times each line appeared in the input file.
-d	Displays only the repeated lines.
-f *Fields*	Ignores the number of fields specified by the *Fields* variable. If the value of the *Fields* variable exceeds the number of fields on a line of input, the **uniq** command uses a null string for comparison. This flag is equivalent to the **-***Fields* flag.
-u	Displays only the unrepeated lines.
-s *Characters*	Ignores the number of characters specified by the *Characters* variable. If the value of the *Characters* variable exceeds the number of characters on a line of input, the **uniq** command uses a null string for comparison. If both the **-f** and **-s** flags are specified, the **uniq** command ignores the number of characters specified by the **-s** *Characters* flag starting in the field following the fields specified by the **-f** *Fields* flag. This flag is equivalent to the **+***Characters* flag.
-*Fields*	Ignores the number of fields specified by the *Fields* variable. This flag is equivalent to the **-f** *Fields* flag.
+*Characters*	Ignores the number of characters specified by the *Characters* variable. If both the **-***Fields* and **+***Characters* flags are specified, the **uniq** command ignores the number of characters specified by the **+***Characters* flag starting in the field following the fields specified by the **-***Fields* flag. This flag is equivalent to the **-s** *Characters* flag.

Exit Status

This command returns the following exit values:

Item	Description
0	The command ran successfully.
>0	An error occurred.

Example

To delete repeated lines in a file named `fruit` and save it to a file named `newfruit`, enter:

```
uniq fruit newfruit
```

If the `fruit` file contains the following lines:

```
apples
apples
peaches
pears
bananas
```

```
cherries
cherries
```

then the `newfruit` file will contain the following lines after you run the **uniq** command:

```
apples
peaches
pears
bananas
cherries
```

Files

Item	Description
/usr/bin/uniq	Contains the **uniq** command.

units Command

Purpose

Converts units in one measure to equivalent units in another measure.

Syntax

units [**-**] [*File*]

Description

The **units** command converts quantities expressed in one measurement to their equivalents in another. The **units** command is an interactive command. It prompts you for the unit you want to convert *from* and the unit you want to convert *to*. This command only does multiplicative scale changes. That is, it can convert from one value to another only when the conversion involves a multiplication. For example, it cannot convert between degrees Fahrenheit and degrees Celsius because the value of 32 must be added or subtracted in the conversion.

You can specify a quantity as a multiplicative combination of units, optionally preceded by a numeric multiplier.

Indicate powers by entering suffixed positive integers, and indicate division with a / (slash).

The **units** command recognizes `lb` as a unit of mass, but considers pound to be the British pound sterling. Compound names are run together (such as `lightyear`). Prefix British units differing from their American counterparts with `br` (`brgallon`, for instance).

The **/usr/share/lib/unittab** file contains a complete list of the units that the **units** command uses. You can also define new units in this file. The *File* parameter may be used to override the values of the standard conversion factors listed in the **/usr/share/lib/unittab** file. The specified file must follow the same format as the **unittab** file.

Most familiar units, abbreviations, and metric prefixes are recognized by the **units** command, as well as the following:

Item	Description
pi	Ratio of circumference to diameter
c	Speed of light
e	Charge on an electron
g	Acceleration of gravity
force	Same as **g**

Item	Description
mole	Avogadro's number
water	Pressure head per unit height of water
au	Astronomical unit

Flags

Item	Description
-	Lists the conversion factors contained in the **/usr/share/lib/unittab** file before you are prompted to enter your conversion.

Examples

1. To display conversion factors for inches to centimeters, enter:

   ```
   units
   you have: in
   you want: cm
   ```

   The **units** command returns the following values:

   ```
   * 2.540000e+00
   / 3.937008e-01
   ```

   The output tells you to multiply the number of inches by `2.540000e+00` to get centimeters, and to multiply the number of centimeters by `3.937008e-01` to get inches.

   These numbers are in standard exponential notation, so `3.937008e-01` means `3.937008 x 10-1`, which is the same as `0.3937008`.

   > **Note:** The second number is always the reciprocal of the first; for example, `2.54` equals `1/0.3937008`.

2. To convert a measurement to different units, enter:

   ```
   units
   you have: 5 years
   you want: microsec
   ```

   The **units** command returns the following values:

   ```
   * 1.577846e+14
   / 6.337753e-15
   ```

   The output shows that `5 years` equals `1.577846 x 1014` microseconds, and that one microsecond equals `6.337753 x 10-15` years.

3. To give fractions in measurements, enter:

   ```
   units
   you have: 1|3 mi
   you want: km
   ```

   The **units** command returns the following values:

   ```
   * 5.364480e-01
   / 1.864114e+00
   ```

   The | (vertical bar) indicates division, so `1|3` means one-third. This shows that one-third mile is the same as `0.536448` kilometers.

4. To include exponents in measurements, enter:

```
units
you have: 1.2-5 gal
you want: floz
```

The **units** command returns the following values:

```
* 1.536000e-03
/ 6.510417e+02
```

The expression `1.2-5 gal` is the equivalent of `1.2 x 10-5`. Do *not* type an e before the exponent (that is, `1.2e-5 gal` is not valid). This example shows that `1.2 x 10-5 (0.000012)` gallons equal `1.536 x 10-3 (0.001536)` fluid ounces.

5. To specify complex units, enter:

```
units
you have: gram centimeter/second2
you want: kg-m/sec2
```

The **units** command returns the following values:

```
* 1.000000e-05
/ 1.000000e+05
```

The units `gram centimeter/second2` mean "grams x centimeters/second2." Similarly, `kg-m/sec2` means "kilograms x meters/sec2," which is often read as "kilogram-meters per seconds squared."

6. If the units you specify after `you have:` and `you want:` are incompatible:

```
you have: ft
you want: lb
```

The **units** command returns the following message and values:

```
conformability
3.048000e-01 m
4.535924e-01 kg
```

The `conformability` message means the units you specified cannot be converted. Feet measure length, and pounds measure mass, so converting from one to the other does not make sense. Therefore, the **units** command displays the equivalent of each value in standard units.

In other words, this example shows that one foot equals `0.3048` meters and that one pound equals `0.4535924` kilograms. The **units** command shows the equivalents in meters and kilograms because the command considers these units to be the standard measures of length and mass.

Files

Item	Description
/usr/bin/units	Contains the **units** command.
/usr/share/lib/unittab	Lists units that the **units** command creates as well as units defined by the user.

unlink Command

Purpose

Performs an **unlink** subroutine.

Syntax

unlink *File*

Description

The **unlink** command performs the **unlink** subroutine on a specified file.

The **unlink** command does not issue error messages when the associated subroutine is unsuccessful; you must check the exit value to determine if the command completed normally. It returns a value of 0 if it succeeds, a value of 1 if too few or too many parameters are specified, and a value of 2 if its system call is unsuccessful.

> **Attention:** The **unlink** command allows a user with root user authority to deal with unusual problems, such as moving an entire directory to a different part of the directory tree. It also permits you to create directories that cannot be reached or escaped from. Be careful to preserve the directory structure by observing the following rules:

- Be certain every directory has a . (dot) link to itself.
- Be certain every directory has a .. (dot dot) link to its parent directory.
- Be certain every directory has no more than one link to itself or its parent directory.
- Be certain every directory is accessible from the root of its file system.

An attempt to remove a file or directory that has been exported for use by the NFS version 4 server will fail with a message saying that the resource is busy. The file or directory must be unexported for NFS version 4 use before it can be removed.

Example

To remove a directory entry pointed by `file2`, enter:

```
unlink file2
```

Files

Item	Description
/usr/sbin/unlink	Contains the **unlink** command.

unloadipsec Command

Purpose

Unloads a crypto module from the IP Security subsystem.

Syntax

unloadipsec -c *crypto_mod_name*

Description

The **unloadipsec** command unloads a crypto module from the IP Security subsystem. The **unloadipsec** command can be used when a crypto module is no longer being used or when a crypto module is to be replaced with a newer version.

A crypto module can only be unloaded after the IP Security device is stopped. The steps for replacing a crypto module are: change the IP Security device to the defined state; unload the old crypto module using this command; uninstall the old module and install the new module, and bring the IP Security device back to the available state.

Flags

Item	Description
-c *crypto_mod_name*	Specifies the name of the crypto module to be unloaded. When used without any flag, the command lists all the crypto modules installed (but not necessarily loaded).

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

unmirrorvg Command

Purpose

Removes the mirrors that exist on volume groups or specified disks.

Syntax

unmirrorvg [**-c** *Copies*] [**-p** *mirrorpool*] *VolumeGroup* [*PhysicalVolume*...]

Description

The **unmirrorvg** command unmirrors all the logical volumes detected on a given volume group. This same functionality may also be accomplished manually if you execute the **rmlvcopy** command for each individual logical volume in a volume group.

By default, **unmirrorvg** will pick the set of mirrors to remove from a mirrored volume group. If you wish to control which drives no longer are to contain mirrors, you must include the list of disks in the input parameters, *PhysicalVolume.*

When the *PhysicalVolume* parameter is listed in the command, this indicates that only logical volumes with copies that exist on this *PhysicalVolume* should be unmirrored. Logical volumes that exist solely on the other drives in the volume group are unaffected and remain mirrored.

Note:

1. If LVM has not recognized that a disk has failed it is possible that LVM will remove a different mirror. Therefore if you know that a disk has failed and LVM does not show those disks as missing you should specify the failed disks on the command line or you should use **replacepv** to replace the disk or **reducevg** to remove the disk.
2. If a logical volume copy spans more than one disk, the portion of the logical volume copy that resides on a disk not listed by the user is also removed.
3. The **unmirrorvg** command is not allowed on a snapshot volume group.
4. Using a *PhysicalVolume* list with the **-c 1** option (the default) will cause affected triply-mirrored logical volumes to have two copies removed. Only one of these copies will be related to the listed physical volumes. This is because the physical volume list is used to determine affected logical volumes, which are then reduced to the specified number of copies. In this case, the second copy to remove is selected by **unmirrorvg**
5. When a corresponding hard disk and **/dev/ipldevice** are removed then a reboot is required.

6. If you are removing the first mirror pool copy by specifying the disks in the first copy to remove, you might also want to move your logical volumes mirror pool assignments by running the **chlv** command. For example:

```
chlv -m copy1=poolb -M 2 lv00
```

When **unmirrorvg** is executed, the default COPIES value for each logical volume becomes 1. If you wish to convert your volume group from triply mirrored to doubly mirrored, use the **-c** option.

Note: To use this command, you must either have root user authority or be a member of the **system** group.

Attention: The **unmirrorvg** command may take a significant amount of time to complete because of complex error checking and the number of logical volumes to unmirror in a volume group.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit unmirrorvg
```

Flags

Item	Description
-c *Copies*	Specifies the minimum number of copies that each logical volume must have after the **unmirrorvg** command has finished executing. If you do not want all logical volumes to have the same number of copies, then reduce the mirrors manually with the **rmlvcopy** command. If this option is not used, the copies will default to 1.
-p *mirrorpool*	Removes the copy that exists on specified mirror pools. To remove more than one copy, provide multiple [**-p** *mirrorpool*] flags.

The following is a description of **rootvg**:

Item	Description
rootvg unmirroring	When the **rootvg** unmirroring has completed, you must perform two additional tasks: **bosboot** and **bootlist**. The **bosboot** command is required to reinitialize the boot record on the remaining disk. The **bootlist** command needs to be performed so that the system will only boot to the disk left in **rootvg**.

Examples

1. To unmirror a triply mirrored volume group and leave two copies, enter:

   ```
   unmirrorvg -c 2 workvg
   ```

   The logical partitions in the logical volumes held on workvg now have 2 copies.
2. To get default unmirroring of rootvg, enter:

   ```
   unmirrorvg rootvg
   ```

   rootvg now has only 1 copy.
3. To replace a bad disk drive in a mirrored volume group, enter:

   ```
   unmirrorvg workvg hdisk7
   reducevg workvg hdisk7
   rmdev -l hdisk7 -d
   replace the disk drive, let the drive be renamed hdisk7
   extendvg workvg hdisk7
   mirrorvg workvg
   ```

Note: By default in this example, **mirrorvg** will try to create 2 copies for logical volumes in `workvg`. It will try to create the new mirrors onto the replaced disk drive. However, if the original system had been triply mirrored, there may be no new mirrors created onto `hdisk7`, as other copies may already exist for the logical volumes. This follows the default behavior of **unmirrorvg** to reduce the mirror copy count to 1.

Note: When **unmirrorvg workvg hdisk7** is run, **hdisk7** will be the remaining drive in the volume group. This drive is not actually removed from the volume group. You must run the **migratepv** command to move the data from the disk that is to be removed from the system to disk **hdisk7**.

Files

Item	Description
/usr/sbin	Directory where the **unmirrorvg** command resides.

unpack Command

Purpose

Expands files.

Syntax

unpack *File ...*

Description

The **unpack** command expands files created by the **pack** command. For each file specified, the **unpack** command searches for a file called *File***.z**. If this file is a packed file, the **unpack** command replaces it by its expanded version. The **unpack** command names the new file name by removing the **.z** suffix from *File*. If the user has root authority, the new file has the same access modes, access and modification times, owner, and group as the original file. If the user does not have root authority, the file retains the same access modes, access time, and modification time, but acquires a new owner and group.

The **unpack** command operates only on files ending in **.z**. As a result, when you specify a file name that does not end in **.z**, the **unpack** command adds the suffix and searches the directory for a file name with that suffix.

The exit value is the number of files the **unpack** command was unable to unpack. A file cannot be unpacked if any of the following occurs:

- The file name (exclusive of **.z**) has more than 253 bytes.
- The file cannot be opened.
- The file is not a packed file.
- A file with the unpacked file name already exists.
- The unpacked file cannot be created.

Note: The **unpack** command writes a warning to standard error if the file it is unpacking has links. The new unpacked file has a different i-node than the packed file from which it was created. However, any other files linked to the original i-node of the packed file still exist and are still packed.

Exit Status

This command returns the following exit values:

Item	Description
0	The command ran successfully.
>0	An error occurred.

Example

To unpack packed files:

```
unpack chap1.z chap2
```

This expands the packed files `chap1.z` and `chap2.z`, and replaces them with files named `chap1` and `chap2`. Note that you can give the **unpack** command file names either with or without the **.z** suffix.

Files

Item	Description
/usr/bin/unpack	Contains the **unpack** command.

untab Command

Purpose

Changes tabs into spaces.

Syntax

untab [*FileName ...*]

Description

The **untab** command reads the file specified by the *FileName* parameter or standard input, and replaces tabs in the input with space characters. If you specify a file with the *FileName* parameter, the **untab** command writes the resulting file back to the original file. If the input is standard input, the **untab** command writes to standard output. The **untab** command assumes that tab stops are set every eight columns, starting with column nine. The file name specified for the *FileName* parameter cannot exceed **PATH_MAX**-9 bytes in length.

Example

To replace tab characters in the `File` file with space characters, enter:

```
untab File
```

Files

Item	Description
/usr/bin/untab	Contains the **untab** command.

update Command

Purpose

Periodically updates the super block.

Syntax

update

Description

The **update** command executes a **sync** subroutine every 30 seconds. This action ensures the file system is up-to-date in the event of a system crash.

Files

Item	Description
/usr/sbin/update	Contains the **update** command.

update_iscsi Command

Purpose

Lists and updates the configurations of devices for the iSCSI software initiator that is accessed through the iSCSI software initiator or the iSCSI TOE adapter.

Syntax

update_iscsi [-l *name* **]**

Description

The **update_iscsi** command lists and updates the devices for which configuration attributes are related to iSCSI and must be migrated to the Object Data Manager (ODM) of the **rootvg** image.

You can run the **update_iscsi** command in maintenance mode after all of the file systems that contain the base operating system in the **rootvg** image are mounted. Note that only the devices that are causing iSCSI boot problems should be updated.

To list the devices for which the iSCSI configuration attributes are changed, run the **update_iscsi** command without any argument.

To migrate the configuration of a listed device to the ODM of the **rootvg** image, run the **update_iscsi** command with the **-l** *name* flags. The *name* parameter represents the ODM name of a device in the RAM file system.

The **update_iscsi** command displays the devices that are listed in the **iscsi_devlist** file, which is located in the **/etc/objrepos** directory. The command lists these devices after matching them to the corresponding **rootvg** entries. If the **iscsi_devlist** file is missing, or if the file lists no devices, a message will be printed indicating that you did not set the ODM for the RAM file system.

Flags

Item	Description
`-l`	Specifies the ODM name of a device in the RAM file system. This flag is optional.

Parameters

Item	Description
name	The ODM name of a device in the RAM file system.

Sample Output

The following sample shows the output of the **update_iscsi** command with no flag specified:

```
RAM FS DEVICE NAME     ROOTVG DEVICE NAME       DESCRIPTION
inet0                  inet0                    Internet Network Extension
en0                    en1                      Standard Ethernet Network Interface
iscsi0                 iscsi0                   iSCSI Protocol Device
```

Exit Status

If the **update_iscsi** command cannot find the ODM name that the *name* parameter specifies, the value of the **ROOTVG DEVICE NAME** is set to *New Device*.

If the **iscsi_devlist** file is missing or empty, an error message is printed.

Location

`/usr/sbin/`

Files

Item	Description
`iscsi_devlist`	Contains a list of the devices with attributes that are set through the Network Disk Install menu.

updatevsdnode Command

Purpose

Modifies virtual shared disk subsystem options.

Syntax

updatevsdnode
-n {**ALL** | *node_number* [,node_number ...]}

{[**-a** *VSD_adapter*]

[**-b** *min_buddy_buffer_size*]

[**-x** *max_buddy_buffer_size*

[**-s** *max_buddy_buffers*]

[**-M** *vsd_max_ip_packet_size*]}

[**-f**] [-c *cluster_name* | NONE]

Description

Use **updatevsdnode** to modify virtual shared disk subsystem options.

Note: This command only modifies the subsystem options. To effectively configure the virtual shared disks, you must first unconfigure all the virtual shared disks, unload the device driver, and then reconfigure the shared disks.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Set/Show Virtual Shared Disk Device Driver Operational Parameters** option or the **Update virtual shared disk Device Driver Node Parameters** option.

Flags

-n
Specifies the node numbers of the nodes whose information you want this command to update, or **ALL** nodes in the RSCT peer domain. You can issue the command **/usr/bin/lsclcfg** to find out the node number of the node you are running on.

-a
Specifies the adapter name to be used for virtual shared disk communications with this node or nodes. You must specify **ml0** as the adapter name.

-b
Specifies the smallest buddy buffer a server uses to satisfy a remote request to a virtual shared disk. This value must be a power of 2 and greater than or equal to 4096. The suggested value to use is 4096 (4 KB).

-x
The largest buddy buffer a server will use to satisfy a remote request. This value must be a power of 2 and greater than or equal to the *min_buddy_buffer_size*. The suggested value to use is 262144 (256 KB). This value must be the same on all nodes in the RSCT peer domain.

-s
This is the number of *max_buddy_buffer_size* buffers to allocate. The virtual shared disk device driver will have an initial size when first loaded, and then will dynamically allocate and reclaim additional space as needed. The suggested starting value for a 32-bit kernel is 128 256 KB buffers. The suggested value is 2000 256KB buffers.

Buddy buffers are only used on the servers. On client nodes you may want to set *max_buddy_buffers* to 1.

Note: The `statvsd` command will indicate if remote requests are queueing waiting for buddy buffers.

-M
Specifies the maximum message size in bytes for virtual shared disks. This value must not be greater than the maximum transmission unit (MTU) size of the network. The recommended values are:

- 61440 (60 KB) for a switch
- 8192 (8 KB) for jumbo frame Ethernet
- 1024 (1 KB) for 1500-byte MTU Ethernet

-f
Specifies that this command will force updates to virtual shared disk subsystem options by reconfiguring one or more virtual shared disks on all nodes in the RSCT peer domain on which virtual shared disks are currently configured.

-c *cluster_name* | NONE
Changes the cluster the node belongs to. NONE removes the node from the cluster.

Note: The *cluster_name* is required only for SSA (Serial Storage Architecture) disks.

Parameters

vsd_name
Specifies the virtual shared disk whose underlying logical volume you no longer want to be globally accessed by any virtual shared disk nodes.

Security

You must have `root` authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to *RSCT Administration Guide*.

Examples

To increase the buddy buffer size to 48 maximum sized buddy buffers on node 3, enter:

```
updatevsdnode -n 3 -s 48
```

Note: The device driver must be unconfigured from the kernel and reloaded to have this change go into effect.

Location

/opt/lpp/vsd/bin/updatevsdnode

updatevsdtab Command

Purpose

updatevsdtab – Changes the Virtual shared disk subsystem attributes.

Syntax

updatevsdtab {**-v** *vsd_names* | **-a**} {[`-s ]`} [**-f**]

Description

Use this command to update the virtual shared disk size. When you change the virtual shared disk size using the `updatevsdtab` command, the change will not take effect until the virtual shared disk is unconfigured and configured again.

If the **-f** flag is specified, the virtual shared disks involved will be reconfigured on all nodes that are up and initially had these virtual shared disks configured.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the Set/Show virtual shared disk Device Driver Operational Parameters option or the Update virtual shared disk Options option.

Flags

-v *vsd_names*
Specifies a list of virtual shared disk names to be updated.

-a
Specifies that the option is to be changed on all nodes of the system or system partition.

-s
Updates the virtual shared disk size after the associated logical volume size is changed.

-f
Forces changes by reconfiguring a virtual shared disk on all nodes in the current system partition on which the virtual shared disk is configured.

Parameters

None.

Security

You must have root authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

1. To reset the size of the virtual shared disk named USER1n3, enter:

   ```
   updatevsdtab -v USER1n3 -s
   ```

Location

`/usr/lpp/csd/bin/updatevsdtab`

updatevsdvg Command

Purpose

Changes virtual shared disk global volume group characteristics.

Syntax

updatevsdvg { -a | -g *global_volgrp* { -k VSD -p *primary_node* -b *secondary_node* | -k CVSD -l *server_list* [-c *cluster_name*] } }

Description

The updatevsdvg command changes virtual shared disk global volume group characteristics. This command allows you to change global volume groups from concurrent virtual shared disk volume groups to serial-access (or nonconcurrent) virtual shared disk volume groups, and the other way around. This command can be used whenever server node numbers change, such as replacing or re-cabling servers where the new server numbers are different, or when you need to delete a server.

This command performs the following operations:

1. Suspends all virtual shared disks that are part of this volume group
2. Stops all virtual shared disks that are part of this volume group
3. Issues the varyoffvg command for the volume group
4. Verifies that the volume group exists on the new servers and tries to import the volume group if it does not exist
5. Updates the global volume group characteristics
6. Issues the varyonvg command for the volume group to the appropriate servers
7. Starts all virtual shared disks that are part of this volume group

Note:

1. If you issue this command with the -a flag, the recoverable virtual shared disk subsystem should not be active. Otherwise, this command can be run while the recoverable virtual shared disk subsystem is active, as long as no application is using the virtual shared disks that are part of the volume group being updated.
2. Concurrent virtual shared disks are supported for disks that have implemented the SCSI-3 persistent reserve model of the AIX SCSI device drivers, and for SSA (Serial Storage Architecture) disks.

Flags

-a
Specifies that persistent reserve information should be reestablished in the object data manager (ODM) for all VSD volume groups served by this node. This flag is intended for the initial setup phase of allowing multiple clusters to access the same virtual shared disks. It is also useful for recovery after the device ODM entries have been removed inadvertently.

This flag causes all of the volume groups served by the node to be varied offline. The volume groups will be varied offline on this node and on all other servers for the volume groups. For this reason, you should stop the recoverable virtual shared disk subsystem before issuing the updatevsdvg command with this flag.

-b *secondary_node*
Specifies the secondary node.

-c *cluster_name*
Specifies the cluster name for the server nodes that will be serving concurrently accessed shared disks. This flag is applicable only for SSA (Serial Storage Architecture) disks, and a *cluster_name* must be specified for SSA.

-g *global_volgrp*
Specifies an existing global volume group name.

-k VSD | CVSD
Specifies whether the volume group will be of type concurrent virtual shared disk or serial-access (nonconcurrent) virtual shared disk.

-l *server_list*
Specifies a colon-separated list of servers for concurrent virtual shared disks.

-p *primary_node*
Specifies the primary node.

Parameters

vsd_name
Specifies the virtual shared disk whose underlying logical volume you no longer want to be globally accessed by any virtual shared disk nodes.

Security

You must have `root` authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to *RSCT Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

1. To change the global volume group named `ess_gvg` from a virtual shared disk global volume group to a concurrent global volume group with three servers, assuming that the disks are cabled correctly and that the disk subsystem supports persistent preserve such as ESS disks, enter:

   ```
   updatevsdvg -g ess_gvg -k CVSD -l 9:17:21
   ```

2. To remove a server from an SSA global volume group named `ssa_gvg`, where the original server list is `9:10` and belongs to an SSA cluster named `cluster9_10`, (that is, the command `vsdatalst -c` shows SSA cluster information), enter:

   ```
   updatevsdvg -g ssa_gvg -k CVSD -l 9 -c cluster9_10
   ```

3. To change a concurrent global volume group named `ess_gvg` back to a virtual shared disk global volume group, where the original server list is `9:17:21`, the new primary node number is 9, and the new secondary node number is 21, enter:

```
updatevsdvg -g ess_gvg -k VSD -p 9 -b 21
```

Location

`/opt/rsct/vsd/bin/updatevsdvg`

uprintfd Daemon

Purpose

Constructs and writes kernel messages.

Syntax

uprintfd

Description

The **uprintfd** daemon retrieves, converts, formats, and writes kernel messages to processes' controlling terminals. Kernel messages are submitted through the **NLuprintf** and **uprintf** kernel services. Because the **uprintfd** daemon never exits, it should be run only once.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

uptime Command

Purpose

Shows how long the system has been up.

Syntax

uptime

Description

The **uptime** command prints the current time, the length of time the system has been up, the number of users online, and the load average. The load average is the number of runnable processes over the preceding 1-, 5-, 15-minute intervals. The output of the **uptime** command is, essentially, the heading line provided by the **w** command.

useradd Command

Purpose

Creates a new user account.

Syntax

useradd [-c *comment*] [-d *dir*] [-e *expire*] [-g *group*] [-G *group1,group2 ...*] [-m [-k *skel_dir*]] [-u *uid*] [-s *shell*] [-r *role1,role2 ...*] *login*

Description

The useradd command creates a new user account. The *login* parameter must be a unique string (its length is can be configured by administrators using the chdev command). You cannot use the ALL or default keywords in the user name.

The useradd command does not create password information for a user. It initializes the password field with an asterisk (*). Later, this field is set with the passwd or pwdadm command. New accounts are disabled until the passwd or pwdadm commands are used to add authentication information to the /etc/security/passwd file.

The useradd command always checks the target user registry to make sure the ID for the new account is unique to the target registry. The useradd command can also be configured to check all user registries of the system using the dist_uniqid system attribute. The dist_uniqid system attribute is an attribute of the usw stanza of the /etc/security/login.cfg file, and can be managed using the chsec command.

The dist_uniqid system attribute has the following values:

never
Does not check for ID collision against the nontarget registries. This is the default setting.

always
Checks for ID collision against all other registries. If collision is detected between the target registry and any other registry, account creation or modification fails.

uniqbyname
Checks for ID collision against all other registries. Collision between registries is allowed only if the account to be created has the same name as the existing account.

Note: ID collision detection in the target registry is always enforced regardless of the dist_uniqid system attribute.

The uniqbyname system attribute setting works well against two registries. With more than two registries, and with ID collision already existing between two registries, the behavior of the useradd command is unspecified when creating a new account in a third registry using colliding ID values. The new account creation might succeed or fail depending on the order in which the registries are checked.

The check for ID collision only enforces ID uniqueness between the local registry and remote registries, or between remote registries. There is no guarantee of ID uniqueness between the newly created account on the remote registry and existing local users on other systems that make use of the same remote registry. The useradd command bypasses a remote registry if the remote registry is not reachable at the time the command is run.

Flags

Item	Description
-c *comment*	Supplies general information about the user specified by the *login* parameter. The *comment* parameter is a string with no embedded colon (:) characters and cannot end with the characters '#!'.
-d *dir*	Identifies the home directory of the user specified by the *login* parameter. The *dir* parameter is a full path name.

Item	Description
`-e` *expire*	Identifies the expiration date of the account. The *expire* parameter is a 10-character string in the *MMDDhhmmyy* form, where *MM* is the month, *DD* is the day, *hh* is the hour, *mm* is the minute, and *yy* is the last 2 digits of the years 1939 through 2038. All characters are numeric. If the *expire* parameter is 0, the account does not expire. The default is 0. See the `date` command for more information.
`-g` *group*	Identifies the user's primary group. The *group* parameter must contain a valid group name and cannot be a null value.
`-G` *group1,group2,...*	Identifies the groups the user belongs to. The *group1,group2,...* parameter is a comma-separated list of group names.
`-k` *skel_dir*	Copies default files from *skel_dir* to user's home directory. Used only with `-m` flag.
`-m`	Makes user's home directory if it does not exist. The default is not to make the home directory.
`-r` *role1,role2,...*	Lists the administrative roles for this user. The *role1,role2,...* parameter is a list of role names, separated by commas.
`-s` *shell*	Defines the program run for the user at session initiation. The *shell* parameter is a full path name.
`-u` *uid*	Specifies the user ID. The *uid* parameter is a unique integer string. Avoid changing this attribute so that system security will not be compromised.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **`lssecattr`** command or the **`getcmdattr`** subcommand.

Examples

1. To create the `davis` user account with default values, enter:

```
useradd davis
```

Restrictions

To prevent login inconsistencies, avoid composing user names entirely of uppercase alphabetic characters. While the `useradd` command supports multibyte user names, restrict user names to characters with the POSIX-portable filename character set.

To ensure that your user database remains uncorrupted, you must be careful when naming users. User names must not begin with a hyphen (-), plus sign (+), at sign (@), or tilde (~). You cannot use the keywords `ALL` or `default` in a user name. Additionally, do not use any of the following characters within a user-name string:

Item	Description
:	Colon
"	Double quote
#	Pound sign
,	Comma
=	Equal sign
\	Back slash
/	Slash
?	Question mark
'	Single quote
`	Back quote

Finally, the *login* parameter cannot contain any space, tab, or newline characters.

Location

`/usr/sbin/useradd`

Files

The `useradd` command has read and write permissions to the following files.

Item	Description
`/etc/passwd`	Contains the basic attributes of users.
`/etc/security/user`	Contains the extended attributes of users.
`/etc/security/user.roles`	Contains the administrative role attributes of users.
`/etc/security/limits`	Defines resource quotas and limits for each user.
`/etc/security/environ`	Contains the environment attributes of users.
`/etc/security/audit/config`	Contains audit configuration information.
`/etc/security/lastlog`	Contains the last login attributes of users.
`/etc/group`	Contains the basic attributes of groups.
`/etc/security/group`	Contains the extended attributes of groups.

userdel Command

Purpose

Removes a user account.

Syntax

`userdel` [`-r`] *login*

Description

The `userdel` command removes the user account identified by the *login* parameter. The command removes a user's attributes without removing the user's home directory by default. The user name must already exist. If the `-r` flag is specified, the `userdel` command also removes the user's home directory.

If the **AIX_USERDEL_RECURSIVE_DEL** environment variable is set, the **userdel** command recursively deletes the directories and files that belong to the removed user. If another user uses the same home directory, the files and directories of the user is preserved. If the directory of the deleted user contains content owned by a different user, the directory ownership of the user is changed to the user **nobody** with a permission of **777** and a **sticky bit** set. This operation is performed for the continued access of the directory and its content for the affected users by using the same home space. It is very important to change the permission and ownership of the affected directories to a new user immediately after running the **userdel** command. The system administrator can change the permission and ownership setting of the affected directories to a new user to prevent illegal access.

Only the root user or users with `UserAdmin` authorization can remove administrative users. Administrative users are those users with `admin=true` set in the `/etc/security/user` file.

Flags

Item	Description
`-r`	Removes the home directory of the user. Files located in other file systems must be searched manually and deleted. Removing the home directory, which is shared by other users, might leave the system in an inconsistent state.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To remove the user `davis` account and its attributes from the local system, enter:

```
userdel davis
```

Location

`/usr/sbin/userdel`

Files

The `userdel` command has read and write permissions to the following files.

Item	Description
`/etc/passwd`	Contains the basic attributes of users.

Item	Description
/etc/security/user	Contains the extended attributes of users.
/etc/security/user.roles	Contains the administrative role attributes of users.
/etc/security/limits	Defines resource quotas and limits for each user.
/etc/security/environ	Contains the environment attributes of users.
/etc/security/audit/config	Contains audit configuration information.
/etc/security/lastlog	Contains the last login attributes of users.
/etc/group	Contains the basic attributes of groups.
/etc/security/group	Contains the extended attributes of groups.

usermod Command

Purpose

Changes user attributes.

Syntax

usermod [-u *uid*] [**-g** *pgroup*] [-G *group1,group2 ...*] [-d *dir* [-m]] [-s *shell*] [-c *comment*] [-l *new_name*] [-e *expire*] [-r *role1,role2 ...*] *login*

Description

Attention: Do not use the usermod command if you have a Network Information Service (NIS) database installed on your system.

The usermod command changes attributes for the user identified by the *login* parameter. The user name must already exist. To change an attribute, specify the flag and the new value. The following files contain local user attributes that are set by this command:

- /etc/passwd
- /etc/security/environ
- /etc/security/limits
- /etc/security/user
- /etc/security/user.roles
- /etc/security/audit/config
- /etc/group
- /etc/security/group

Avoid changing the ID for an account so that system security is not compromised. However, when the ID is changed using the usermod command, ID collision checking is also controlled by the dist_uniqid attribute in the usw stanza of the /etc/security/login.cfg file. The behavior of ID collision control is the same as that described for the mkuser command.

Flags

Item	Description
-c *comment*	Supplies general information about the user specified by the *login* parameter. The *comment* parameter is a string with no embedded colon (:) characters and cannot end with the characters '#!'.

Item	Description
-d *dir*	Changes the home directory to the directory specified by the *dir* parameter.
-g *pgroup*	Identifies the primary group. The *pgroup* parameter must be a valid group name or ID.
-e *expire*	Identifies the expiration date of the account. The *expire* parameter is a 10-character string in the *MMDDhhmmyy* form, where *MM* is the month, *DD* is the day, *hh* is the hour, *mm* is the minute, and *yy* is the last 2 digits of the years 1939 through 2038. All characters are numeric. If the *expire* parameter is 0, the account does not expire. The default is 0. See the `date` command for more information.
-G *group1,group2,...*	Identifies the groups the user belongs to. The *group1,group2,...* parameter is a comma-separated list of group names.
-l *new_name*	Specifies the new name of the user.
-m	Moves the contents of the user's current home directory to the new home directory. Only used with the `-d` flag.
-r *role1,role2,...*	Lists the administrative roles for this user. The *role1,role2,...* parameter is a list of role names, separated by commas.
-s *shell*	Defines the program run for the user at session initiation. The *shell* parameter is a full path name.
-u *uid*	Specifies the user ID. The *uid* parameter is a unique integer string. Avoid changing this attribute so that system security will not be compromised.

Exit Status

Item	Description
`0`	The command completed successfully.
`>0`	An error occurred.

Examples

1. To change the user `davis` to be a member of the `system` group, enter the following command:

```
usermod -G system davis
```

Restrictions

To ensure the integrity of user information, some restrictions apply when using the **usermod** command. Only the root user or users with `UserAdmin` authorization can use the `usermod` command to perform the following tasks:

- Make a user an administrative user by setting the admin attribute to true.
- Change any attributes of an administrative user.
- Add a user to an administrative group

An administrative group is a group with the `admin` attribute set to True. Members of the security group can change the attributes of non-administrative users and add users to non-administrative groups.

The `usermod` command manipulates local user data only. You cannot use it to change data in registry servers like NIS and DCE.

Location

`/usr/sbin/usermod`

Files

The `usermod` command has read and write permissions to the following files.

Item	Description
`/etc/passwd`	Contains the basic attributes of users.
`/etc/security/user`	Contains the extended attributes of users.
`/etc/security/user.roles`	Contains the administrative role attributes of users.
`/etc/security/limits`	Defines resource quotas and limits for each user.
`/etc/security/environ`	Contains the environment attributes of users.
`/etc/security/audit/config`	Contains audit configuration information.
`/etc/security/lastlog`	Contains the last login attributes of users.
`/etc/group`	Contains the basic attributes of groups.
`/etc/security/group`	Contains the extended attributes of groups.

users Command

Purpose

Displays a compact list of the users currently logged on to the system.

Syntax

users [*FileName* | *WparName*]

Description

The **users** command lists the login names of the users that are currently logged on to the system to standard output (**stdout**) in a compact, one-line list format. If you specify absolute path name of a file, then it is used as an alternate file instead of **/etc/utmp**. If you do not specify an absolute path name, it is considered to be the name of a workload partition. If the name is "Global", it indicates the global environment.

Files

Item	Description
/etc/utmp	Contains list of current users.
/usr/bin/users	Contains the **users** command.

Note: The `/etc/utmp` file for a particular workload partition can be indicated by prefixing the root path for the workload partition.

usrck Command

Purpose

Verifies the correctness of a user definition.

Syntax

usrck { **-l** [**-b**] | **-n** | **-p** | **-t** | **-y** } { **ALL** | *User ...* }

Description

The **usrck** command verifies the correctness of the user definitions in the user database files, by checking the definitions for **ALL** the users or for the users specified by the *User* parameter. If more than one user is specified, there must be a space between the names. You must select a flag to indicate whether the system should try to fix erroneous attributes.

The command first checks the entries in the **/etc/passwd** file. If you indicate that the system should fix errors, duplicate user names are reported and disabled. Duplicate IDs are reported only, because there is no system fix. If an entry has fewer than six colon-separated fields, the entry is reported, but not fixed. The **usrck** command next checks specific user attributes in other files.

The **usrck** command verifies that each user name listed in the **/etc/passwd** file has a stanza in the **/etc/security/user**, **/etc/security/limits** and **/etc/security/passwd** files. The **usrck** command also verifies that each group name listed in the **/etc/group** file has a stanza in the **/etc/security/group** file. The **usrck** command using the **-y** flag creates stanzas in the security files for the missing user and group names.

Note:

- This command writes its messages to **stderr**.
- If the *domainlessgroups* attribute is set, the **usrck** command will throw an error for the Lightweight Directory Access Protocol (LDAP) users.

A list of all the user attributes follows, with notations stating which attributes are checked:

Item	Description
account_locked	No check. The **usrck** command sets this attribute to True and disables accounts.
admgroups	Checks to see if the **admgroups** are defined in the user database and, if you indicate that the system should fix errors, the command removes any groups that are not in the database.
auditclasses	Checks to see if the **auditclasses** are defined for the user in the **/etc/security/audit/config** file. If you indicate that the system should fix errors, the command deletes all the auditclasses that are not defined in the **/etc/security/audit/config** file.
auth1	Checks the primary authentication method. Unless the method is NONE or SYSTEM, it must be defined in the **/etc/security/login.cfg** file and the program attribute must exist and be executable by the root user. If you indicate that the system should fix errors, it will disable the user account if an error is found. **Note:** The **auth1** attribute is deprecated and should not be used.

Item	Description
auth2	Checks the secondary authentication method. Unless the method is NONE or SYSTEM, it must be defined in the **/etc/security/login.cfg** file and the program attribute must exist and be executable by the root user. There is no system fix. **Note:** The **auth2** attribute is deprecated and should not be used.
core	Ensures that the values are sensible. If not, the command resets the values to 200 blocks, the minimum value.
core_hard	Ensures that the values are sensible. If not, the command resets the values to 200 blocks, the minimum value.
cpu	Ensures that the values are sensible. If not, the command resets the values to 120 seconds, the minimum value.
cpu_hard	Ensures that the values are sensible. If not, the command resets the values to 120 seconds, the minimum value.
data	Ensures that the values are sensible. If not, the command resets the values to 1272 blocks (636K), the minimum value.
data_hard	Ensures that the values are sensible. If not, the command resets the values to 1272 blocks (636K), the minimum value.
dictionlist	Checks the list of dictionary files. If you indicate that the system should fix errors, all dictionary files that do not exist are deleted from the user database.
expires	No check.
fsize	Ensures that the values are sensible. If not, the command resets the values to 200 blocks, the minimum value.
fsize_hard	Ensures that the values are sensible. If not, the command resets the values to 200 blocks, the minimum value.
gecos	No check.
histexpire	Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.
histsize	Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.
home	Checks the existence and accessibility of the home directory by read mode and search mode. If you indicate that the system should fix errors, it will disable the user account if an error is found.
id	Checks the uniqueness of the user ID. If you indicate that the system should fix errors, the command deletes any invalid entry in the **/etc/passwd** file.
login	No check.
loginretries	Checks if the user attempted unsuccessful logins more than the allowable amount. If so, the system disables the user account.
logintimes	Ensures that the string of time specifiers is valid. If you indicate that the system should fix errors, the system disables the user account if an error is found.

Item	Description
maxage	Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.
maxexpired	Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.
maxrepeats	Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.
minage	Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value. The system also indicates if the **minage** attribute is larger than the **maxage** attribute.
minalpha	Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.
mindiff	Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.
minlen	Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.
minother	Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value. The system also indicates if the **minage** attribute plus the **maxage** attribute is greater than the maximum password size.
name	Checks the uniqueness and composition of the user name. The name must be a unique string of eight bytes or less. It cannot begin with a + (plus sign), a : (colon), a - (minus sign), or a ~ (tilde). Names beginning with a + (plus sign) or with a - (minus sign) are assumed to be names in the NIS (Network Information Service) domain, and no further processing is performed. It cannot contain a colon (:) in the string and cannot be the **ALL** or **default** keywords. If you indicate that the system should fix errors, the command disables the user account if an error is found and deletes any invalid entry in the **/etc/passwd** file. The **usrck** command verifies that, for each user name listed in the **/etc/passwd** file, there is a stanza in the **/etc/security/user**, **/etc/security/limits**, and **/etc/security/passwd** files. The command adds stanzas for each one identified as missing. The **usrck** command additionally verifies that each group name listed in the **/etc/group** file has a stanza in the **/etc/security/group** file.
nofiles	Ensures that the value is sensible. If not, resets the value to 200, the minimum value.
nofiles_hard	Ensures that the value is sensible. If not, resets the value to 200, the minimum value.
pgrp	Checks for the existence of the primary group in the user database. If you indicate that the system should fix errors, it will disable the user account if an error is found.

Item	Description
pwdchecks	Checks the list of external password restriction methods. If you indicate that the system should fix errors, all methods that do not exist are deleted from the user database.
pwdwarntime	Ensures that the value is sensible. If not, the system resets the value to the difference between the **maxage** and **minage** values.
rlogin	No check.
rss	Checks to ensure that the values are sensible. If not, the command resets the values to 128 blocks (64KB), the minimum value. The value is not set by the system.
rss_hard	Checks to ensure that the values are sensible. If not, the command resets the values to 128 blocks (64KB), the minimum value. The value is not set by the system.
shell	Checks the existence and accessibility of the shell by execute mode. If you indicate that the system should fix errors, it will disable the user account if an error is found.
stack	Checks to ensure that the values are sensible. If not, the command resets the values to 128 blocks (64KB), the minimum value.
stack_hard	Checks to ensure that the values are sensible. If not, the command resets the values to 128 blocks (64KB), the minimum value.
su	No check.
sugroups	Checks for the existence of the **sugroups** in the user database files. If you indicate that the system should fix errors, it will delete all the groups that are not in the database.
sysenv	No check.
tpath	Checks to ensure that the **shell** attribute is tagged as a trusted process if **tpath=always**. If you indicate that the system should fix errors, it will disable the user account if an error is found.
ttys	Checks for the existence of the ttys in the user database files. If you indicate that the system should fix errors, it will delete all the ttys that do not exist from the user database.
usrenv	No check.

If the fix involves disabling a user account, use the **chuser** command to reset the value of the **account_locked** attribute to False. You can use the System Management Interface Tool (SMIT) to run the **chuser** command by entering:

```
smit chuser
```

The root user or a member of the security group can enable a user account again by removing the **account_locked** attribute or setting the **account_locked** attribute to False. The root user's account is not disabled by the **usrck** command.

Generally, the **sysck** command calls the **usrck** command as part of the verification of a trusted-system installation. If the **usrck** command finds any errors in the user database, the root user or a member of the security group should execute both the **grpck** command and the **pwdck** command.

The **usrck** command checks to see if the database management security files (**/etc/passwd.nm.idx**, **/etc/passwd.id.idx**, **/etc/security/passwd.idx**, and **/etc/security/lastlog.idx**) files are up-to-date or newer than the corresponding system security files. Please note, it is acceptable for the **/etc/security/lastlog.idx** to be not newer than **/etc/security/lastlog**. If the database management

security files are out-of-date, a warning message appears indicating that the root user should run the **mkpasswd** command.

The **usrck** command checks if the specified user can log in. If the user cannot log in because of too many unsuccessful login attempts or because the password is expired, the **usrck** command issues a warning message indicating why the user cannot log in. If you indicate that the system should fix errors, the system disables the user account if the user cannot log in for the above reasons.

If the **-l** flag is specified, the **usrck** command scans all users or the users specified by the *User* parameter to determine if users can access the system. The criteria used to determine accessibility for a user are listed in the following table:

Table 23. User Accessibility Criteria

Criterion	Description	Cause
1	User account is locked.	The user's **account_locked** attribute is set to **true**.
2	User account is expired.	The user's **expires** attribute is set to a value (expiration time) that is expired.
3	User has too many consecutive failed login attempts.	The user's **unsuccessful_login_count** value is greater than the user's **loginretries** value.
4	User has no password.	The user's **password** field is '*' in **/etc/password** or **/etc/security/password**.
5	User is not allowed to log in for this date/time.	The current date/time is not within the allowed time as defined by the user's **logintimes** attribute.
6	The **/etc/nologin** file exists.	The **/etc/nologin** file prevents a non-root user from logging in.
7	User password is expired and only system administrator can change it.	The user's password is expired and the ADMIN password flag is set.
8	User is denied login to host.	The user's **hostallowedlogin** and **hostsdeniedlogin** attributes do not allow access to the current host.
9	User is denied access by applications.	The user's **login**, **rlogin**, and **su** attributes are set to false and the **rcmds** attribute is set to deny. If at least one but not all of these attribute values deny authorization, the system is considered partially accessible by the user.
10	User is denied login to terminal.	The user's **ttys** attribute does not allow access to the current terminal. The system is considered partially accessible for the user.

If the **-b** flag is also specified, the output consists of two fields, the user name and a 16-digit bit mask, separated by a tab. Each digit in the bit mask corresponds to a criteria in the User Accessibility Criteria table above, with criteria 1 represented by the rightmost digit. If the bit location for a criteria is set to 1, the check for this criteria failed for the user. Extra digits in the output are reserved for future use.

The following is an example of the **usrck** command with the **-l** flag:

```
# usrck -l testusr1 testusr2
3001-689 The system is inaccessible to testusr1, due to the following:
         User account is locked
         User denied login to terminal.

3001-689 The system is inaccessible to testusr2, due to the following:
         User account is expired.
         User has too many consecutive failed login attempts.
         User denied login to host.
```

The following is an example of the **usrck** command with the **-l** and **-b** flags:

```
# usrck -lb testusr1 testusr2
 testusr1       0000000000000001
 testusr2       0000000001000110
```

Flags

Item	Description
-b	Reports users who are not able to access the system and the reasons, with the reasons displayed in a bit-mask format. The **-l** flag must be specified if the **-b** flag is specified. **Note:** The bit mask does not report criteria 10 (user denied login to terminal), since this cannot be considered a complete scenario when determining if a system is inaccessible to a user. Likewise, the bit mask does not report criteria 9 (User denied access by applications) if at least one but not all of the attributes' values deny authentication; this criteria is only reported when all four attribute values deny authentication.
-l	Scans all users or the users specified by the *User* parameter to determine if the users can access the system.
-n	Reports errors but does not fix them.
-p	Fixes errors but does not report them.
-t	Reports errors and asks if they should be fixed.
-y	Fixes errors and reports them.

Exit Status

This command returns the following exit values:

Item	Description
0	User definition files are appropriate.

Item	Description
>0	An error occurred or there is an error in one or more user definition files. The following error codes are returned: **EINVAL (22)** Invalid command-line arguments **ENOENT (2)** One or more user definition files do not exist **ENOTRUST (114)** Errors in user definitions in the database files or users unable to access the system (found by `-l` option)

Security

Access Control: This command should grant execute (x) access to the root user and members of the security group. The command should be **setuid** to the root user and have the **trusted computing base** attribute.

Files Accessed:

Mode	File
r	**/etc/passwd**
r	**/etc/security/user**
rw	**/etc/security/group**
rw	**/etc/group**
rw	**/etc/security/lastlog**
rw	**/etc/security/limits**
rw	**/etc/security/audit/config**
rw	**/etc/security/login.cfg**

Auditing Events:

Event	Information
USER_Check	user, attribute-error, status

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To verify that all the users exist in the user database, and have any errors reported (but not fixed), enter:

   ```
   usrck -n ALL
   ```

2. To delete from the user definitions those users who are not in the user database files, and have any errors reported, enter:

   ```
   usrck -y ALL
   ```

3. To display the list of users who are unable to access the system, enter:

```
usrck -l ALL
```

4. To display the list of users who are unable to access the system, in a bit mask format, enter:

```
usrck -l -b ALL
```

Files

Item	Description
/usr/bin/usrck	Specifies the path of the **usrck** command.
etc/passwd	Contains basic user attributes.
/etc/security/user	Contains the extended attributes of users.
/etc/group	Contains basic group attributes.
/etc/security/group	Contains the extended attributes of groups.
/etc/security/lastlog	Contains the last login attributes for users.
/etc/security/limits	Contains the process resource limits of users.
/etc/security/audit/config	Contains audit system configuration information.
/etc/security/login.cfg	Contains configuration information.

usrrpt Command

Purpose

Reports the security capabilities of users.

Syntax

usrrpt [**-R** <load_module>] [-C] [**-a** | **-c** | **-f**] user_list

Description

The **usrrpt** command reports security capability information of users such as privileged commands executable by them, privileged files that can be accessed, and also the authorizations associated with the user.

Either of –a, -c, -f flags can be specified. When the –a option is specified, the list of authorizations associated with the user is displayed. When the -c option is specified, the privileged commands present in the /etc/security/privcmds database that can be executed by that user is listed. When the –f option is specified, the list of privileged files present in the /etc/security/privfiles database that can be accessed by the authorized user is listed.

The command takes a list of **comma** separated user names as input. When no option is specified, all the capability information such as authorizations, commands and privileged files information associated with the user is listed.

Flags

Item	Description
-a	Specify that a report of authorizations associated with the users is to be obtained.
-c	Specify that a report of privileged commands executable by the users is to be obtained.

Item	Description
-f	Specify that a report of privileged files accessible by the user is to be obtained.
-R	Specifies the loadable module to obtain the report of authorization capabilities from.
-C	Displays the authorization attributes in colon-separated records, as follows: #user:attribute1:attribute2: ... user1:value1:value2: ... user2:value1:value2: ...

Exit status

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: This command should grant execute (x) access to the root user.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** Command or the **getcmdattr** Subcommand.

Examples

1. To report the commands associated with user Bob:

```
usrrpt –c Bob
```

2. To report all capabilities of user Simon:

```
usrrpt Simon
```

3. To report all capabilities of user Simon in colon separated format

```
usrrpt –C Simon
```

Information similar to the following appears:

```
#user:authorizations:commands:privfiles
Simon:aix.security.user:/usr/bin/mkuser,/usr/bin/chuser:/etc/csh.cshrc,/etc/csh.login
```

Files

/etc/security/roles
/etc/security/authorizations
/etc/security/privcmds
/etc/security/privfiles

utmpd Daemon

Purpose

Monitors and maintains **/etc/utmp** file.

Syntax

/usr/sbin/utmpd [*Interval*]

Description

The **utmpd** daemon monitors the **/etc/utmp** file for validity of the user process entries at regular intervals. An user process that has been terminated, but has not been cleaned up in the **/etc/utmp** file, is removed by cross checking the process id of the entry against the process table.

The Interval parameter specifies the amount of time in seconds between each scan of the **/etc/utmp** file. The default interval time would be 300 seconds.

Usage

To start **utmpd** from **/etc/inittab**, add the following entry to the file:

```
utmpd:2:respawn:/usr/sbin/utmpd
```

init starts the **utmpd** daemon during system startup. To have the changes take effects immediately without rebooting, type:

```
telinit q
```

Security

Only the root user can read and execute this command.

Files

Item	Description
/etc/inittab	Specifies stanzas read by the **init** command.
/etc/utmp	Contains a record of users logged into the system.

uucheck Command

Purpose

Checks for files and directories required by BNU.

Syntax

uucheck [**-v**] [**-x** *DebugLevel*]

Description

The **uucheck** command verifies the presence of the files and directories required by the Basic Networking Utilities (BNU) facility. The command also checks for some errors in the **/etc/uucp/Permissions** file.

> **Note:** The **uucheck** command does not check for correct file and directory modes or for errors in the **/etc/uucp/Permissions** file, such as duplicate login or machine names.

Issue the **uucheck** command from the command line after installing the BNU program, configuring the BNU facility for your site, or making changes in part of the BNU facility, such as the **/etc/uucp/Permissions** file.

> **Note:** Only someone with root user authority can use the **uucheck** command at the command line.

Flags

Item	Description
-v	Displays a detailed explanation of how BNU interprets the **/etc/uucp/Permissions** file.
-x*DebugLevel*	Displays debugging information. The valid range for the *DebugLevel* variable is 0 to 9, with a default of 5. The higher the number, the more detailed the information.

Examples

1. To find out how the BNU programs interpret the **/etc/uucp/Permissions** file, enter:

   ```
   uucheck -v
   ```

   The **-v** flag instructs the **uucheck** command to verify that the BNU files exist and displays a detailed explanation of how the BNU programs interpret the **/etc/uucp/Permissions** file. The output is similar to the following:

   ```
   *** uucheck: Check Required Files and Directories
   *** uucheck: Directories Check Complete

   *** uucheck:  Check /etc/uucp/Permissions file
   ** LOGNAME PHASE (when they call us)

   When a system logs in as: (unostro)
      We DO allow them to request files.
      We WILL send files queued for them on this call.
      They can send files to
         /
      They can request files from
         /
      Myname for the conversation will be plague.austin..
      PUBDIR for the conversation will be
      /var/spool/uucppublic.

   ** MACHINE PHASE (when we call or execute their uux requests)

   When we call system(s): (nostromo)
      We DO allow them to request files.
      They can send files to
         /
      They can request files from
         /
      Myname for the conversation will be plague.austin..
      PUBDIR for the conversation will be
      /var/spool/uucppublic.

   Machine(s): (nostromo)
   CAN execute the following commands:
   command (ALL), fullname (ALL)

   *** uucheck:  /etc/uucp/Permissions Check Complete
   ```

   For an explanation of these permissions, see the **/etc/uucp/Permissions** file.

2. To debug with the **uucheck** command, enter:

   ```
   uucheck -x8
   ```

   The `-x8` flag produces extensive debugging output.

Files

Item	Description
/etc/uucp//etc/uucp/Permissions	Describes access permissions for remote systems.
/etc/uucp/Systems	Describes accessible remote systems.

uucico Daemon

Purpose

Transfers Basic Networking Utilities (BNU) command, data, and execute files to remote systems.

Syntax

uucico [**-r** *RoleNumber*] [**-x** *DebugLevel*] **-s** *SystemName*

Description

The **uucico** daemon transfers Basic Networking Utilities (BNU) command (**C.***), data (**D.***), and execute (**E.***) files, created by the **uucp** and **uux** commands, to a specified remote system. Both the local and remote systems run the **uucico** daemon, and the two daemons communicate with each other to complete transfer requests.

The **uucico** daemon performs the following actions:

1. Scans the spooling directory (**/var/spool/uucp/***SystemName*) on the local system for transfer requests.
2. Selects the device used for the communications connection after checking the **/etc/uucp/Devices** file and the lock files in the **/etc/locks** directory.
3. Places a call to the specified remote system using information in the **Systems**, **Dialers**, and **Dialcodes** files located in the **/etc/uucp** directory.
4. Performs the required login sequence specified in the **Systems** file.
5. Checks permissions listed in the **/etc/uucp/Permissions** file.
6. Checks scheduling limits in the **Maxuuscheds** and **Maxuuxqts** files located in the **/etc/uucp** directory.
7. Runs all transfer requests from both the local and the remote system, placing the transferred files in the public directories (**/var/spool/uucppublic/***).
8. Logs transfer requests and completions in files in the **/var/spool/uucp/.Log/uucico** directory.
9. Notifies specified users of transfer requests.

Usually the **uucico** daemon is called by the **uucp** and **uux** commands when needed and is started periodically by the BNU scheduling daemon, **uusched**, which is started by the **cron** daemon.

The **uucico** daemon can be started from the command line for debugging. The BNU **uutry**, **Uutry**, and **uukick** commands also start the **uucico** daemon with debugging turned on.

Requirement: Either you must be in the **/usr/sbin/uucp** directory when you call the **uucico** daemon, or you must call the daemon with the full path name, **/usr/sbin/uucp/uucico**.

Tip: In the case of a **uux** command request for the execution of a command on a remote system, the **uucico** daemon transfers the files and the **uuxqt** daemon executes the command on the remote system.

Flags

Item	Description
-r *RoleNumber*	Specifies the server and client relationship. The role numbers are 1 for server mode and 0 for client mode. If the **-r** flag is not used, the **uucico** daemon is started in client mode (**-r** 0), because the **uucico** daemon is generally started automatically by a BNU command or daemon. When the **uucico** daemon is started manually, this flag should be set to 1.
-x *DebugLevel*	Displays debugging information on the screen of the local terminal. The valid range for the *DebugLevel* variable is 0 to 9, with a default of 5. Higher numbers cause the information to be more detailed. This flag is useful for diagnosing problems with the expect-send sequence in the **/etc/uucp/Systems** file.
-s *SystemName*	Specifies the name of the remote system. This flag is required when starting the **uucico** daemon from the command line. The *SystemName* variable is supplied internally when the **uucico** daemon is started automatically. **Note:** System names must contain only ASCII characters.

Example

To call the **uucico** daemon from the command line, enter:

```
/usr/sbin/uucp/uucico  -r 1  -s hera &
```

to start the daemon as a background process and contact remote system `hera`.

Files

Item	Description
/etc/locks /*	Contains lock files which prevent multiple uses of devices and multiple calls to systems.
/usr/sbin/uucp/*	Contains the **uucico** daemon and the configuration files for BNU.
/etc/uucp/Devices	Contains information about available devices.
/etc/uucp/Dialcodes	Contains dialing code abbreviations.
/etc/uucp/Dialers	Specifies initial handshaking on a connection.
/etc/uucp/Maxuuscheds	Limits scheduled jobs.
/etc/uucp/Maxuuxqts	Limits remote command executions.
/etc/uucp/Permissions	Describes access permissions for remote systems.
/etc/uucp/Systems	Describes accessible remote systems.
/var/spool/uucp/.Admin/errors	Lists **uucico** daemon errors that BNU cannot correct.
/var/spool/uucp/.Log/uucico /*	Contains **uucico** daemon log files.
/var/spool/uucp/.Status/*SystemName*	Lists the last time a remote system was contacted and the minimum time until the next retry.

Item	Description
/var/spool/uucp/*SystemName* **/***	Contains **C.***, **D.***, and **X.*** files to be transferred by the **uucico** daemon.
/var/spool/uucp/*SystemName***/C.***	Contains command files.
/var/spool/uucp/*SystemName***/D.***	Contains data files.
/var/spool/uucp/*SystemName***/X.***	Contains execute files.
/var/spool/uucppublic/*	Contain files after transfer by the **uucico** daemon.

uuclean Command

Purpose

Removes files from the BNU spool directory.

Syntax

/usr/sbin/uucp/uuclean [**-m**] [**-n***Hours*] [**-p***Prefix*] [**-d***Subdirectory*]

Description

The **uuclean** command checks the Basic Networking Utilities (BNU) spool directory (**/var/spool/uucp**) for files with the specified prefixes and deletes those that are older than the given number of hours. If the **-n***Hours* flag is not included, the **uuclean** command deletes files that are older than 72 hours.

If the **-p** flag is not included, the **uuclean** command deletes all files in the specified subdirectories of the spool directory that meet the age requirement. If the **-d** flag is not included, the command deletes all the files (that meet the age and prefix requirements) in all the subdirectories of the spool directory. Thus if neither the **-d** or the **-p** flag is included, the **uuclean** command deletes *all* files in *all* subdirectories of the **/var/spool/uucp** directory that meet the age requirement.

If the **-m** flag is not specified, the **uuclean** command sends mail to owners of all command (**C.***) files that it deletes. If the **-m** flag is specified, the command sends mail to the owner of each file it deletes, including data (**D.***) and execute (**X.***) files. The mail message includes the name of the deleted file.

The **uuclean** command is usually run by the **cron** daemon.

> **Note:** Only someone with root user authority or who is logged in as **uucp** can issue the **uuclean** command.

Flags

Item	Description
-d*Subdirectory*	Deletes files from the specified subdirectory of the **/var/spool/uucp** directory if they match specifications given with the **-n** and **-p** flags. If the **-d** flag is not specified, the **uuclean** command checks all subdirectories of the **/var/spool/uucp** directory. Up to 10 subdirectories can be specified with the **-d** flag.
-m	Instructs the **uuclean** command to send mail to the owner of each file when it is deleted.
-n*Hours*	Deletes files whose ages are more than the number of hours specified by the *Hours* variable, if they match specifications given with the **-d** and **-p** flags. The default is 72 hours.

Item	Description
-p*Prefix*	Deletes files with the prefix given by the *Prefix* variable, if they match specifications given with the **-n** and **-d** flags. Up to 10 prefixes can be specified with the **-p** flag.

Examples

1. To delete all old command files, enter:

   ```
   /usr/sbin/uucp/uuclean -pC
   ```

   This command deletes all files in all subdirectories of the **/var/spool/uucp** directory whose names begin with C and that are older than 72 hours (the default). The system sends mail to the original owner of each file, stating that the file has been deleted.
2. To delete all old files from the spool directory for systems `venus` and `nostromo`, enter:

   ```
   /usr/sbin/uucp/uuclean -n84 -dvenus -dnostromo
   ```

   This command deletes all files in the **/var/spool/uucp/venus** and **/var/spool/uucp/nostromo** directories that are older than 84 hours. By default, the system notifies owners of **C.*** files that the files have been deleted; however, it does not notify owners of other files it deletes.
3. To delete all old files from all spool directories and notify users that they have been deleted, enter:

   ```
   /usr/sbin/uucp/uuclean -m
   ```

   This command deletes all files in all subdirectories of the spool directory, if the files are older than 72 hours (the default). It sends mail to the owner of each file it deletes.
4. To schedule the **uuclean** command to be started periodically by the **cron** daemon, add an entry similar to the following to your **/var/spool/cron/crontabs/uucp** file:

   ```
   15 22 * * * /usr/sbin/uucp/uuclean -n96 -pC -pD -pX
   ```

   This entry will cause the **cron** daemon to start the **uuclean** command at 22:15 (10:15 p.m.) daily. The **uuclean** command will delete all command (**C.***), data (**D.***), and execute (**X.***) files that are older than 96 hours from all subdirectories of the spool directory.

Files

Item	Description
/usr/sbin/uucp/uuclean	Contains the **uuclean** command.
/var/spool/uucp /*	Contains spooling files removed by the **uuclean** command.
/var/spool/cron/crontabs/uucp	Schedules **uucp** jobs for the **cron** daemon.

uucleanup Command

Purpose

Deletes selected files from the Basic Networking Utilities (BNU) spooling directory.

Syntax

uucleanup [**-C***Days*] [**-W***Days*] [**-m***String*] [**-D***Days*] [**-T***Days*] [**-X***Days*] [**-o** *Days*] [**-s***SystemName*]

Description

The Basic Networking Utilities (BNU) **uucleanup** command scans the spooling directory (**/var/spool/uucp**) for files that are older than a specified number of days and removes them. The **uucleanup** command performs the following tasks:

- Informs the requester of send and receive requests for systems that cannot be reached.
- Warns users about requests that have been waiting for a given number of days. The default is 1 day.
- Returns to the sender mail that cannot be delivered.
- Removes from the spool directory all other files older than a specified number of days.

Requirements:

- Only someone with root user privileges can issue the **uucleanup** command from the command line. The **uucleanup** command is not usually entered on the command line but is executed by the **uudemon.cleanu** command, a shell procedure.
- When BNU is installed, automatic cleanup is not enabled. Edit the **/var/spool/cron/crontabs/uucp** file and remove the comment character (*#*) from the beginning of the **uudemon.cleanu** line to instruct the **cron** daemon to start the **uudemon.cleanu** command.

Flags

Item	Description
-C*Days*	Removes **C.*** (command) files as old as, or older than, the number of days specified by the *Days* variable, and notifies the requester that the files have been deleted. The default time is 7 days.
-D*Days*	Removes **D.*** (data) files as old as, or older than, the number of days specified by the *Days* variable. Also attempts to deliver any remaining mail messages. The default time is 7 days.
-m*String*	Includes a specified line of text in the warning message generated by the **-W***Days* option. The default line is `See your local administrator to locate the problem`.
-o*Days*	Removes other files as old as, or older than, the number of days specified by the *Days* variable. The default time is 2 days.
-s*SystemName*	Executes the **uucleanup** command only on the spooling directory specified by the *System* variable. The default is to clean up all BNU spooling directories. **Restriction:** System names can contain only ASCII characters.
-T*Days*	Removes **TM.*** (temporary) files as old as, or older than, the number of days specified by the *Days* variable. Also attempts to deliver any remaining mail messages. The default time is 7 days.
-W*Days*	Sends an electronic mail message to the requester warning that **C.*** (command) files as old as, or older than, the number of days specified by the *Days* variable are still in the spooling directory. The message includes the job ID and, if the request included mail, the mail message. The administrator can use the **-m** option to include a message line telling whom to call to check the problem. The default time is 1 day.
-X*Days*	Removes any **X.*** (execute) files as old as, or older than, the number of days specified by the *Days* variable. The default time is 2 days.

Examples

Warning Users That Their Command Files Have Not Been Sent

1. To send a warning for **C.*** (command) files 2 or more days old, enter:

   ```
   uucleanup  -W2
   ```

   This warns the requester that the files have not been sent.
2. To send a message with the warning, enter:

   ```
   uucleanup  -m"Check these files waiting in the BNU job queue."
   ```

   This locates **C.*** (command) files 1 or more days old (default), warns requesters that their files have not been sent, and gives the message: `Check these files waiting in the BNU job queue.`

Cleaning Up Command, Data, Execute, and Miscellaneous Files

1. To clean up command files 5 or more days old, enter:

   ```
   uucleanup  -C5
   ```

   This removes all **C.*** (command) files 5 or more days old and sends an appropriate message to the requesters.
2. To clean up data and execute files 3 or more days old, enter:

   ```
   uucleanup  -D3  -X3
   ```

   This removes all **D.*** (data) files and all **X.*** (execute) files 3 or more days old.
3. To clean up all files at once using defaults, enter:

   ```
   uucleanup
   ```

   This removes all **C.***, **D.***, **T.***, and **X.*** files, and all other files older than the default times.

   Important: Whenever the **-C** and **-W** flags are used together, make sure the value specified for the **-W** flag is less than that for the **-C** flag. Otherwise, the **-C** flag will delete all the **C.*** (command) files before any warnings can be printed.

Cleaning Up Files for a Specific System

To delete files for one system, enter:

```
uucleanup  -shera
```

This removes all files using defaults for system `hera`, but does not remove any files for any other systems.

Files

Item	Description
/usr/sbin/uucp/*	Contains the **uudemon.cleanu** shell procedure and all the configuration files for BNU.
/var/spool/cron/crontabs/uucp	Schedules BNU jobs for the **cron** daemon, including the **uudemon.cleanu** shell procedure.
/var/spool/uucp/*	Contain files removed by the **uucleanup** command.

uucp Command

Purpose

Copies files from one system to another.

Syntax

uucp [**-c** | **-C**] [**-d** | **-f**] [**-g***Grade*] [**-j**] [**-m**] [**-n***User*] [**-r**] [**-s***File*] [**-x***DebugLevel*] *SourceFile ...* *DestinationFile ...*

Description

The **uucp** command is a Basic Networking Utilities (BNU) command that copies one or more source files from one system to one or more destination files on another UNIX system. Files can be copied within a local system, between a local and a remote system, and between two remote systems.

The **uucp** command accomplishes the file transfer in two steps: first, by creating a command (**C.***) file in the spooling directory on the local computer and then by calling the **uucico** daemon to send the request to the specified computer. Command files include information such as the full path name of the source and destination files and the sender's login name. The full path name of a command file is a form of the following:

```
/var/spool/uucp/SystemName/C.SystemNameNxxxx
```

where *N* is the grade of the request and *xxxx* is the hexadecimal sequence number used by BNU.

If the **uucp** command is used with the **-C** flag to copy the files to the spool directory for transfer, the **uucp** command creates not only a command file, but also a data (**D.***) file that contains the actual source file. The full path name of a data file is a form of the following:

```
/var/spool/uucp/SystemName/D.SystemNamexxxxYYYY
```

Once the command files (and data files, if necessary) are created, the **uucp** command then calls the **uucico** daemon, which in turn attempts to contact the remote computer to deliver the files.

It is useful to issue the **uuname** command to determine the exact name of the remote system before issuing the **uucp** command. The **uulog** command provides information about **uucp** activities with another system.

Source and Destination File Names

File names and system names can contain only ASCII characters. Each can either be a path name on the local system or have the following form:

*SystemName***!***PathName*

where *SystemName* is taken from a list of system names that BNU knows about.

The destination *SystemName* can also be a list of names, such as the following:

*SystemName***!***SystemName***!** . . . **!** *SystemName***!***PathName*

In this case, an attempt is made to send the file using the specified route to the destination. Make sure that intermediate nodes in this route are willing to forward information, and that they actually talk to the next system.

The shell pattern-matching characters ? (question mark), * (asterisk), and [. . .] (brackets and ellipsis) can be used in the path names of the source file; the appropriate system expands them. The shell pattern-matching characters should not be used in the path name of the destination file.

If the *DestinationFile* is a directory rather than a file, the **uucp** command uses the last part of the *SourceFile* name to name the transferred file on the remote system.

Path Names

Path names for the *SourceFile* and *DestinationFile* parameters contain only ASCII characters. Paths for the source file can be one of the following:

- A full path name
- A relative path name

Paths for the *DestinationFile* parameter can be in the forms for the *SourceFile* parameter, or can be one of the following:

- A path name preceded by **~***User* (for example, `~jkimble`) where *User* is a login name on the remote system. The specified user's login directory is then considered the destination of the transfer. If the user specifies an invalid login name, the files are transferred to the public directory, **/var/spool/uucppublic**, which is the default.
- A path name preceded by ~**/***Destination*, where *Destination* is appended to **/var/spool/uucppublic**. The destination is treated as a file name unless more than one file is being transferred by the request, the destination already exists as a directory on the remote system, or the destination is specified as a directory.

  To specify the destination as a directory, follow the destination name with a / (slash). For example, `~/amy/` as the destination creates the directory `/var/spool/uucppublic/amy`, if it does not already exist, and puts the requested files in that directory.

Permissions

- The system administrator should restrict the access to local files by users on other systems.
- When transmitting files, the **uucp** command preserves execute permissions and grants read and write permissions to the owner, the group, and all others. (The **uucp** command owns the file.)
- Sending files to arbitrary *DestinationFile* path names on other systems or getting files from arbitrary *SourceFile* path names on other systems often fails because of security restrictions. The files specified in the path name must give read or write permission not only for the same group of users but also for any group.
- Protected files and files in protected directories owned by the requestor can be sent by the **uucp** command.

Flags

Item	Description
-c	Prevents files from being copied. This flag is the default and should not be used with the **-C** flag. If both flags are specified, the **-c** flag is overridden.
-C	Copies local files to the spool directory for transfer. Depending on the configuration of the **Poll** and **Systems** files and on how often the **uusched** daemon is run, the files may be transferred immediately on demand polling or in the future. Occasionally, problems occur while transferring a source file; for example, the remote computer may not be working or the login attempt may fail. In such a case, the file remains in the spool directory until it is either transferred successfully or removed by a cleanup command. This flag counteracts the **-c** flag.
-d	Creates any intermediate directories needed to copy the source files to the destination files on a remote system. Instead of first creating a directory and then copying files to it, the **uucp** command can be entered with the destination path name, and the BNU Program will create the required directory. This flag is the default and cannot be used with the **-f** flag.
-f	Does not create intermediate directories during the file transfer. This flag is used if the destination directory already exists and you do not want BNU to write over it. This command counteracts the **-d** flag.
-g*Grade*	Specifies when the files are to be transmitted during a particular connection. The *Grade* variable is a single number (0 to 9) or letter (A to Z, a to z); lower ASCII-sequence characters cause the files to be transmitted earlier than do higher sequence characters. The number 0 is the highest (earliest) grade; z is the lowest (latest) grade. The default is **N**.

Item	Description
-j	Displays the job identification number of the transfer operation on standard output. This job ID can be used with the **uustat** or **uuq** command to obtain the status of a particular job or with the **uustat -k** command or **uuq -d** command to terminate the transfer before it is completed.
-m	Sends a mail message to the requester when the source file is successfully copied to the destination file on a remote system. The message is sent to the requester's mailbox, **/var/spool/mail/***User*. The **mail** command does not send a message for a local transfer. The **-m** flag works only when sending files or receiving a single file. It does not work when forwarding files.
-n*User*	Notifies the recipient on the remote system identified by the *User* entry that a file has been sent. The mail system does not send a message for a local transfer. User names can contain only ASCII characters. Receiving multiple files specified by the shell pattern-matching characters ? (question mark), * (asterisk), and [. . .] (brackets and ellipses) does not activate the **-n** option.
-r	Prevents the starting of the **uucico** file transfer daemon, even if the command was issued at a time when calls to the remote system are permitted. (By default, a call to the remote system is attempted if the command is issued during a time period specified in the **Poll** and **Systems** files.) The **-r** option is useful for debugging.
-s*File*	Reports the status of the transfer to the specified file. In this case, the *File* variable must designate a full path name.
-x*DebugLevel*	Displays debugging information on the screen of the local system. The *DebugLevel* variable is a number from 0 to 9. The higher the number, the more detailed the report.

Examples

1. To copy a file from the local system to a remote system, enter:

   ```
   uucp /home/geo/f1 hera!/home/geo/f1
   ```

   In this example, the `f1` file from the local system is copied to remote system `hera`.
2. To copy a file from the remote system and place it in the public directory, enter:

   ```
   uucp hera!geo/f2 /var/spool/uucppublic/f2
   ```

   In this example, the `f2` file from remote system `hera` is copied and placed in the public directory.
3. To copy a file from the remote system and place it in a directory other than the public directory, enter:

   ```
   uucp hera!geo/f2 /home/geo/f2
   ```

   In this example, the `f2` file from the remote system `hera` is copied to the `/home/geo/f2` directory. The `geo` login directory must allow write permission to members of the other group, for example, with mode 777.

Files

Item	Description
/usr/bin/uucp	Contains the **uucp** command.
/etc/uucp/Poll	File listing times when remote systems are automatically called (polled).

Item	Description
/etc/uucp/Systems	File describing accessible remote systems.
/etc/uucp/Sysfiles	Specifies alternate files to be used as **Systems** files.
/var/spool/uucp	Spooling directory containing BNU status information.
/var/spool/uucppublic	Public directory containing files awaiting transfer by the **uucico** daemon.
/var/spool/uucppublic/*SystemName***/C.***	Contains command files.
/var/spool/uucppublic/*SystemName***/D.***	Contains data files.

uucpadm Command

Purpose

Enters basic BNU configuration information.

Syntax

uucpadm

Description

The **uucpadm** command provides interactive entry and modification of basic BNU configuration information in the **Devices**, **Systems**, **Permissions**, **Poll**, and **Dialcodes** files in the **/etc/uucp** directory. You can use the **uucpadm** command repeatedly to adjust the same file.

When you enter the **uucpadm** command at the command line, the command displays a list of the files you can change. After you choose a file to modify, the command displays a vertical list of the names of the fields in that file. You can enter the appropriate entry in each field. When you press the Enter key, the cursor moves to the next field in the list.

The command uses a copy of a file to record changes. The original file remains unchanged until you press the Ctrl+U or Ctrl+X key sequence at the appropriate menu. You can exit to the main **uucpadm** menu at any time, without saving your changes, by using the Ctrl+D key sequence.

The help routine provides instructions for each data field. Type a ? (question mark) in any menu field to access the help routine for that field.

Type a **~** (tilde) in any field to enter an ASCII editor and edit the appropriate file for that field. The **uucpadm** command invokes the editor designated by the **EDITOR** environment variable. If the **EDITOR** variable is not defined, the command invokes the **vi** editor.

If your entry for the first menu item matches an existing record, the **uucpadm** command retrieves that record for update. The command also tells you how many records have that first entry. If your entry for the first menu item does not match any existing record, the **uucpadm** command displays the word ADD at the top of the screen.

The **uucpadm** command checks the data as you enter it. If an inconsistency among the files is found, the command displays a warning message.

If the **uucpadm** command recognizes the entry you make for the first menu item, it fills in the default values for the remaining fields. For example, if you type TCP as the Type in the **Devices** file menu, the command places a - (hyphen) in each remaining field for you. It also checks for consistency with other files and for processes that should be running on the system. For example, when you type TCP as the

Type in the **Devices** file menu, the **uucpadm** command checks to see if the **uucpd** daemon is running. If the daemon is not running, the command displays a note after the **Type** field, as follows:

```
Type: TCP
       <Note: Make certain uucpd is enabled.>
Line1: -
```

Note: The **uucpadm** command does not edit the **/etc/uucp/Dialers** file. Use an ASCII editor to edit this file.

Mode	File
rw	**/etc/uucp/Devices**
rw	**/etc/uucp/Dialcodes**
rw	**/etc/uucp/Permissions**
rw	**/etc/uucp/Poll**
rw	**/etc/uucp/Systems**

Examples

1. To start the **uucpadm** command, type the following:

   ```
   /usr/sbin/uucp/uucpadm
   ```

   A menu listing the files you can change is displayed.
2. To make an entry to the **/etc/uucp/Devices** file, choose the `Add/Change Uucp Devices` option at the **uucpadm** menu. The following is a sample **uucpadm** screen defining a `direct 9600 baud` connection to system `merlin` over the `tty3` device:

   ```
   Type: merlin
   line1: tty3
   line2: -
   class: 9600
   dialers: direct
   ```

3. To make an entry to the **/etc/uucp/Systems** file, choose the `Add/Change Uucp Systems` option at the **uucpadm** menu. The following is a sample **uucpadm** screen defining the `nostromo.aus.ibm.com` system connected to an ACU device in `class 2400`:

   ```
   Name: nostromo.aus.ibm.com
   Time: Any
   Type: ACU
   Class: 2400
   Phone: 997-7942
   Login: nuucp
   Password: gotcha
   ```

4. To change the **/etc/uucp/Permissions** file, choose the `Add/Change Uucp Permissions File` option at the **uucpadm** menu.

   a. Following is a sample **uucpadm** screen defining a LOGNAME entry in the **Permissions** file:

      ```
      L/M: LOGNAME=uucpz
      Request: yes
      Sendfiles: yes
      Read: /
      Write: NOWRITE=/etc
      Callback:
      Commands:
      Validate: merlin:nostromo
      ```

      If the remote machine is `merlin` or `nostromo`, the login ID must be `uucpz` (VALIDATE option). Remote hosts using this ID can request to send files, and the local host can sendfiles as requested.

Users with this ID can read all files with permissions granted to the others group, and can write to all files, except those in the **/etc** directory, with permissions granted to the others group.

b. Following is a sample **uucpadm** screen defining a MACHINE entry in the **Permissions** file:

```
L/M: MACHINE=merlin
Request: yes
Sendfiles:
Read: NOREAD=/etc
Write: NOWRITE=/etc
Callback:
Commands: ALL
Validate:
```

The machine ID is `merlin`. Requests for file transfers can be made. The user can read all files and can write to all files except those in the **/etc** directory. The execution of all commands is permitted.

5. To make an entry in the **/etc/uucp/Poll** file, choose the `Add/Change Uucp Poll File` option at the **uucpadm** menu. Following is a sample **uucpadm** screen defining an entry in the **Poll** file:

```
System: merlin
Hours: 0 7 13 19
```

This entry instructs BNU to poll the `merlin.aus.ibm.com` system at 2400 hours (midnight), 700 hours (7 a.m.), 1300 hours (1 p.m.), and 1900 hours (7 p.m.).

6. To make an entry in the **/etc/uucp/Dialcodes** file, choose the `Add/Change Uucp Dialcodes` option at the **uucpadm** menu. Following is a sample **uucpadm** screen defining an entry in the **Dialcodes** file:

```
Abr: LA
Dialcode: 1-213-
```

This entry assigns LA as the abbreviation for the Los Angeles area code.

Files

Item	Description
/usr/sbin/uucp/uucpadm	Contains the **uucpadm** command.
/etc/uucp/Devices	Contains information about available devices.
/etc/uucp/Dialcodes	Contains dialing code abbreviations.
/etc/uucp/Dialers	Specifies initial handshaking on a connection.
/etc/uucp/Permissions	Describes access permissions for remote systems.
/etc/uucp/Poll	Specifies when BNU polls remote systems to initiate tasks.
/etc/uucpSystems/	Describes accessible remote systems.

uucpd Daemon

Purpose

Handles communications between BNU and TCP/IP.

Syntax

The **uucpd** daemon cannot be started from the command line. It is started by the **inetd** daemon.

uucpd

Description

The **uucpd** daemon is an internal program that enables users of systems linked by the Basic Networking Utilities (BNU) program to establish a TCP/IP connection to other systems linked over a Token-Ring, Ethernet, or other network.

The **uucpd** daemon is a subserver of the **inetd** daemon. The **uucpd** daemon must be running as a background process on all the networked systems before the BNU program can use TCP/IP system to communicate. If the **uucpd** daemon is not running, reconfigure the **inetd** daemon to start the **uucpd** daemon. Use the **netstat** command to find out if the **uucpd** daemon is running.

Files

Item	Description
/etc/hosts	Contains the host name table used by TCP/IP.
/etc/inetd.conf	Contains the configuration of the **inetd** daemon.
/etc/services file	Defines socket assignments used by TCP/IP.
/usr/sbin /uucpd	Contains the **uucpd** daemon.
/etc/uucp/Devices	Contains information about available devices.
/etc/uucp/Permissions	Describes access permissions for remote systems.
/etc/uucp/Systems	Describes accessible remote systems.

uudecode Command

Purpose

Decodes a binary file that was used for transmission using electronic mail.

Syntax

uudecode [**-o** *OutputFile*] [*InFile*]

Description

The **uudecode** command reads an encoded file, strips off leading and trailing lines added by mailers, and recreates the original file with the specified mode and name. Decoding a file causes the result to be automatically saved to a file. The file name is identical to the remote file argument originally supplied to the **uuencode** command unless an output file name is specified with the **-o** flag.

Flags

Item	Description
-o *OutputFile*	Specifies the output file name that will be used instead of any pathname contained in the input data. You can direct the output of **uudecode** to standard output by specifying **/dev/stdout** as the *OutputFile*.

Parameters

Item	Description
InFile	Specifies the name of the file to decode.

Example

To decode the file `/tmp/con` on a local system that was encoded with the follwing command:

```
uuencode /usr/lib/boot/unix pigmy.goat > /tmp/con
```

enter:

```
uudecode /tmp/con
```

The file `pigmy.goat` will be identical to the originally encoded file `/usr/lib/boot/unix.`

Files

Item	Description
/usr/bin/uudecode	Contains the **uudecode** command.

uudemon.admin Command

Purpose

Provides periodic information on the status of BNU file transfers.

Syntax

uudemon.admin

Description

The **/usr/sbin/uucp/uudemon.admin** command is a shell procedure that mails status information about the Basic Networking Utilities (BNU) activities to the **uucp** login ID at intervals specified in the **/var/spool/cron/crontabs/uucp** file. The command executes both the **uustat -p** and the **uustat -q** commands:

- The **-p** flag instructs the **uustat** command to run the **ps -flp** command (process status, which generates a full, long list of specified process IDs) for all process ID (PID) numbers in the lock files.
- The **-q** flag lists the jobs currently queued to run on each system. These jobs either are waiting to execute or are in the process of executing. If a status file exists for the system, its date, time, and status information are reported.

Execute the **uudemon.admin** command at least once a day. The **uudemon.admin** command is not enabled when you install the BNU program. To run this command automatically, edit the **/var/spool/cron/crontabs/uucp** file, removing the comment character (#) from the beginning of the line that governs running the **uudemon.admin** command.

Examples

To run the **uudemon.admin** command automatically, edit the **/var/spool/cron/crontabs/uucp** file and remove the comment character (#) from the beginning of the **uudemon.admin** command line. Change:

```
#48 8,12,16 * * * /usr/bin/sh -c
 "/usr/sbin/uucp/uudemon.admin > /dev/null"
```

to:

```
48 8, 12, 16 * * * /usr/bin/sh -c "/usr/sbin/uucp/uudemon.admin > /dev/null"
```

The 48 notation represents minutes, the `8,12,16` notation represents hours based on the 24-hour clock, and the three asterisks (`* * *`) are placeholders representing the day of the month, the month of the year, and the day of the week, respectively. This line therefore instructs the **cron** daemon to run the

uudemon.admin command daily at 48 minutes past the hours 0800, 1200, and 1600, that is, at 8:48 a.m., 12:48 p.m., and 4:48 p.m., respectively.

Note: These run intervals are defaults. By altering them, you can change the times at which the **cron** daemon executes the **uudemon.admin** command to fit the needs of your site.

Files

Item	Description
/usr/sbin/uucp/uudemon.admin	Contains the **uudemon.admin** command and the configuration files for BNU.
/etc/locks/*	Contains lock files which prevent multiple uses of devices and multiple calls to systems.
/var/spool/cron/crontabs/uucp	Schedules BNU jobs, including the **uudemon.admin** command, for the **cron**daemon.

uudemon.cleanu Command

Purpose

Cleans up BNU spooling directories and log files.

Syntax

uudemon.cleanu

Description

The **/usr/sbin/uucp/uudemon.cleanu** command is a shell script that cleans up the Basic Networking Utilities (BNU) spooling directories and log files. The command deletes files in the spooling directories that are as old as, or older than, a specified number of days, and then removes empty spooling directories.

The **uudemon.cleanu** command also updates archived log files by removing log information more than three days old. The command removes log files for individual computers from the **var/spool/uucp/.Log** directory, merges them, and places them in the **var/spool/uucp/.Old** directory, which contains old log information.

After performing the cleanup operations, the **uudemon.cleanu** command mails the **uucp** login ID a summary of the status information gathered during the current day.

Instruct the **cron** daemon to run the **uudemon.cleanu** command daily, weekly, or at longer intervals, depending on the amount of transactions the **uucico** and **uuxqt** daemons perform on the local system.

To run this command automatically, remove the comment character (#) at the beginning of the **uudemon.cleanu** command line in the **/var/spool/cron/crontabs/uucp** file.

Note: The **uudemon.cleanu** command is not usually entered on the command line but is instead executed by the **cron** daemon.

Example

To run the **uudemon.cleanu** procedure automatically, edit the **/var/spool/cron/crontabs/uucp** file and uncomment the **uudemon.cleanu** line. Change:

```
# 45 23 * * * /usr/bin/sh -c
  "/usr/sbin/uucp/uudemon.cleanu > /dev/null"
```

to:

```
45 23 * * * /usr/bin/sh -c "/usr/sbin/uucp/uudemon.cleanu > /dev/null"
```

The 45 notation represents minutes, the 23 notation represents hours based on the 24-hour clock, and the three asterisks (`* * *`) are placeholders representing the day of the month, the month of the year, and the day of the week, respectively. This line therefore instructs the **cron** daemon to run the **uudemon.cleanu** shell procedure at 45 minutes after hour 2300-that is, at 11:45 p.m.

Note:

1. These run intervals are defaults. By altering them, you can change the times at which the **cron** daemon executes the **uudemon.cleanu** command so that they fit the needs of your site.
2. The system allots the BNU program a specified amount of storage space for any one particular log file; the number of blocks is determined by the default **ulimit** value. If the **uudemon.cleanu** command fails to execute because the **ulimit** value is set too low for the requirements of the local system, delete the **uudemon.cleanu** command line (shown previously) from the **/var/spool/cron/crontabs/uucp** file and add the following entry to the **root** crontabs file, **/var/spool/cron/crontabs/root**:

   ```
   45 23 * * * ulimit 5000; /usr/bin/su uucp
    -c "/usr/sbin/uucp/uudemon.cleanu > /dev/null"
   ```

   Put the text on one line when entering it in the **root** crontabs file.

Files

Item	Description
/usr/sbin/uucp/uudemon.cleanu	Contains the **uudemon.cleanu** command.
/var/spool/cron/crontabs/uucp	Schedules BNU jobs, including the **uudemon.cleanu** command, for the **cron** daemon.
/var/spool/cron/crontabs/root	Schedules root user jobs for the **cron** daemon.
/var/spool/uucp/.Log /*	Contains the BNU program log files.

uudemon.hour Command

Purpose

Initiates file transport calls to remote systems using the BNU program.

Syntax

uudemon.hour

Description

The **/usr/sbin/uucp/uudemon.hour** command is a shell procedure used by the Basic Networking Utilities (BNU). In conjunction with the **Poll** file, the **uudemon.poll** command, and the **/var/spool/cron/crontabs/uucp** file, the **uudemon.hour** command initiates calls to remote systems.

The **uudemon.hour** command calls the following programs, which are involved in transferring files between systems at specified hourly intervals:

- The **uusched** daemon first searches the spooling directory on the local system for command files that have not been transferred to the specified remote system, and then schedules the transfer of those files.
- The **uuxqt** daemon searches the spooling directory for execute files that have been transferred to the local system but have not yet been processed on that system.

Instruct the **cron** daemon to run the **uudemon.hour** command at specified hourly intervals. The frequency at which you run the **uudemon.hour** command depends on the amount of file-transfer activity originating from the local computer. If users on the local system initiate a large number of file transfers, you may need to specify that the **cron** daemon should start the **uudemon.hour** command several times an hour. If the number of file transfers originating from the local system is low, you can probably specify a start time once every 4 hours, for example.

To run the **uudemon.hour** command automatically, remove the comment character (#) from the beginning of the **uudemon.hour** command line in the **/var/spool/cron/crontabs/uucp** file.

Note: The **uudemon.hour** command is not usually entered on the command line, but is executed by the **cron** daemon.

Example

To run the **uudemon.hour** command automatically, edit the **/var/spool/cron/crontabs/uucp** file and remove the comment character (#) at beginning of the **uudemon.hour** command line. Change:

```
#25,55 * * * * /usr/bin/sh -c "/usr/sbin/uucp/uudemon.hour >  /dev/null"
```

to:

```
25,55 * * * * /usr/bin/sh -c "/usr/sbin/uucp/uudemon.hour >  /dev/null"
```

The 25 , 55 notation represents minutes, and the four asterisks (* * * *) are placeholders representing the hour of the day, the day of the month, the month of the year, and the day of the week, respectively. Therefore, this line instructs the **cron** daemon to run the **uudemon.hour** command at 25 minutes past the hour and again at 55 minutes past the hour; for example, at 8:25 and 8:55 a.m., again at 9:25 and 9:55 a.m., and again every hour of every day.

Note:

1. These run intervals are defaults. By altering them, you can change the times at which the **cron** daemon executes the **uudemon.hour** command to fit the needs of your site. For example, to run the **uudemon.hour** command once every 4 hours, type the numeral 4 in the **time-interval** field.
2. If you change the run times for the **uudemon.hour** command, you should also change the run times for the **uudemon.poll** command so that it polls remote systems 5 to 10 minutes before the **uudemon.hour** command is run.

Files

Item	Description
/usr/sbin/uucp/uudemon.hour	Contains the **uudemon.hour** command.
/etc/uucp/Poll	Specifies when the BNU program should poll remote systems to initiate tasks.
/var/spool/cron/crontabs/uucp	Schedules BNU jobs, including the **uudemon.hour** and **uudemon.poll** commands, for the **cron** daemon.

uudemon.poll Command

Purpose

Polls the systems listed in the BNU **Poll** file.

Syntax

uudemon.poll

Description

The **/usr/sbin/uucp/uudemon.poll** command is a shell procedure used by the Basic Networking Utilities (BNU). In conjunction with the **/etc/uucp/Poll** file, the **uudemon.hour** command, and the **/var/spool/cron/crontabs/uucp** file, the **uudemon.poll** command initiates calls to remote systems.

The **uudemon.poll** command performs the following actions:

- Polls (contacts) the systems listed in the **Poll** file (**/etc/uucp/Poll**).
- Creates command (**C.***) files for the systems listed in the **Poll** file.

The time at which you run the **uudemon.poll** command depends on the time at which you run the **uudemon.hour** command. In general, schedule the polling shell procedure before the hourly procedure. This schedule enables the **uudemon.poll** command to create any required command files before the **cron** daemon runs the **uudemon.hour** command.

Instruct the **cron** daemon to run the **uudemon.poll** command about 5 to 10 minutes before running the **uudemon.hour** command. To run this procedure automatically, remove the comment character (#) from the beginning of the **uudemon.poll** command line in the **/var/spool/cron/crontabs/uucp** file.

Note: The **uudemon.poll** command is not usually entered on the command line, but is executed by the **cron** daemon.

Example

To run the **uudemon.poll** shell procedure automatically, edit the **/var/spool/cron/crontabs/uucp** file and remove the # (comment character) at the beginning of the line which starts the **uudemon.poll** command. Change:

```
#20,50 * * * * /usr/bin/sh -c "/usr/sbin/uucp/uudemon.poll >  /dev/null"
```

to:

```
20,50 * * * * /usr/bin/sh -c "/usr/sbin/uucp/uudemon.poll >  /dev/null"
```

The `20,50` notation represents minutes, and the four asterisks (`* * * *`) are placeholders representing the hour of the day, the day of the month, the month of the year, and the day of the week, respectively. This line therefore instructs the **cron** daemon to run the **uudemon.poll** command at 20 minutes past the hour and again at 50 minutes past the hour-for example, at 8:20 and 8:50 a.m., and at 9:20 and 9:50 a.m.-every hour of every day.

Note: Change the times at which the **cron** daemon executes the **uudemon.poll** command to correspond to the times you set up for the **uudemon.hour** command. The defaults specified in the **/var/spool/cron/crontabs/uucp** file instruct the **cron** daemon to run the **uudemon.poll** command 5 minutes before running the **uudemon.hour** command.

Files

Item	Description
/usr/sbin/uucp/*	Contains the **uudemon.poll** and **uudemon.hour** commands and all the configuration files for BNU.
/etc/uucp/Poll	Specifies when the BNU program should poll remote systems to initiate tasks.
/var/spool/cron/crontabs/uucp	Schedules BNU jobs, including the **uudemon.poll** command, for the **cron** daemon.

uuencode Command

Purpose

Encodes a binary file for transmission using electronic mail.

Syntax

uuencode [**-m**] [*SourceFile*] *OutputFile*

Description

The **uuencode** command converts a binary file to ASCII data. This is useful before using BNU (or uucp) mail to send the file to a remote system. The **uudecode** command converts ASCII data created by the **uuencode** command back into its original binary form.

The **uuencode** command takes the named *SourceFile* (default standard input) and produces an encoded version on the standard output. The encoding uses only printable ASCII characters, and includes the mode of the file and the *OutputFile* filename used for recreation of the binary image on the remote system.

Use the **uudecode** command to decode the file.

Flags

Item	Description
-m	Encode the output using the MIME Base64 algorithm. If **-m** is not specified, the old **uuencode** algorithm will be used.

Parameters

Item	Description
OutputFile	Specifies the name of the decoded file. You can direct the output of the **uuencode** command to standard output by specifying **/dev/stdout** as the *OutputFile*.
SourceFile	Specifies the name of the binary file to convert. Default is standard input.

Examples

1. To encode the file `unix` on the local system and mail it to the user `jsmith` on another system called `mysys`, enter:

   ```
   uuencode unix unix | mail jsmith@mysys
   ```

2. To encode the file `/usr/lib/boot/unix` on your local system with the name `pigmy.goat` in the file `/tmp/con` , enter:

   ```
   uuencode /usr/lib/boot/unix pigmy.goat > /tmp/con
   ```

Files

Item	Description
/usr/bin/uuencode	Contains the **uuencode** command.

uuid_get command

Purpose

Generates Universal Unique Identifiers (UUIDs).

Syntax

```
uuid_get [ -n count ] [ -o outfile ] [ -c ]
```

Description

The **uuid_get** command generates UUIDs. By default, the **uuid_get** command generates a hexa-string representation of a UUID. You can use the **uuid_get** command along with the **-c** option to generate source-code representation of UUIDs.

Flags

-n count
Generates number of UUIDs that is specified in the count parameter. The value of the **count** parameter must be greater than zero.

-o outfile
Redirects the generated UUID to an output file that is specified by using the **outfile** parameter.

-c
Generates a C programming source-code representation of a UUID.

Examples

1. To generate a hexa-string representation of a UUID, enter the following command:

   ```
   #uuid_get
   ```

   An output similar to the following example is displayed:

   ```
   6ae84954-9ef6-11e6-8003-3a0ea8d2f402
   ```

2. To generate a C programming source-code representation of a UUID, enter the following command:

   ```
   #uuid_get -c
   ```

   An output similar to the following example is displayed:

   ```
   { 0xd966286a,
     0x9ef6,
     0x11e6,
     0x8004,
     {0x3a, 0x0e, 0xa8, 0xd2, 0xf4, 0x02} };
   ```

3. To generate 5 UUIDs by using a single command, enter the following command:

   ```
   # ./uuid_gen -n 5
   ```

   An output similar to the following example is displayed:

   ```
   ba4dae20-f6d7-11e5-8007-3a0ea8d2f402
   ba4daf56-f6d7-11e5-8007-3a0ea8d2f402
   ba4dafe2-f6d7-11e5-8007-3a0ea8d2f402
   ba4db06e-f6d7-11e5-8007-3a0ea8d2f402
   ba4db0fa-f6d7-11e5-8007-3a0ea8d2f402
   ```

uukick Command

Purpose

Uses debugging mode to contact a specified remote system.

Syntax

uukick [**-x***DebugLevel*] *SystemName*

Description

The **uukick** command contacts a remote system, named by the *SystemName* parameter, using debugging mode. The debugging mode provides a means of monitoring Basic Networking Utilities (BNU) file transfers and connections to remote computers.

The **uukick** command starts the **uucico** daemon, which actually contacts the specified remote system. The **uucico** daemon produces debugging output that enables you to monitor its progress as it establishes the connection to the remote system, performs the remote login, and transfers a file.

The debugging output is scrolled on the screen of the local system. Once the system has finished displaying this information, press the Interrupt key to return to the prompt.

Requirement: Either you must be in the **/usr/lib/uucp** directory when you issue the **uukick** command, or you must issue the command with the full path name, **/usr/sbin/uucp/uukick**.

Tip: The **uukick** command is a shell script stored in the **/usr/lib/uucp** directory.

Flags

Item	Description
-x*DebugLevel*	Overrides the default amount of detail in the debugging information the command displays on the screen. The valid range for the *DebugLevel* variable is 0 to 9, with a default of 5. Higher numbers cause the final report to be more detailed. If the **-x** flag is not used, the **uucico** daemon is started with the default level, which produces a moderate amount of information.

Example

To change the amount of detail in the information about the progress of the operation of the **uucico** daemon, use the **-x** flag to specify a higher or lower debugging level. For example, enter:

```
uukick  -x9 hera
```

This instructs the **uukick** command to generate as much information as possible about the way in which the **uucico** daemon is working while trying to connect to system `hera`. Or, enter:

```
uukick  -x3 hera
```

This instructs the command to generate less than the default amount of information about the connection.

Files

Item	Description
/usr/sbin/uucp/uukick	Contains the **uukick** shell script.
/etc/uucp	Contains the configuration files for BNU.

Item	Description
/etc/uucp/Devices	Contains information about available devices.
/etc/uucp/Dialcodes	Contains dialing code abbreviations.
/etc/uucp/Dialers	Specifies initial handshaking on a connection.
/etc/uucp/Permissions	Describes access permissions for remote systems.
/etc/uucp/Systems	Describes accessible remote systems.
/var/spool/uucp/*	Contain files to be transferred and files recording transfer statistics.
/var/spool/uucppublic/*	Contain files that have been transferred.

uulog Command

Purpose

Provides information about BNU file-transfer activities on a system.

Syntax

uulog [**-x**] [**-***Number*] [**-f***System* | **-s***System*]

Description

The Basic Networking Utilities (BNU) **uulog** command displays the contents of the log files containing the activities of the **uucico** and **uuxqt** daemons. Individual log files are created for each remote system with which the local system uses the **uucp**, **uuto**, and **uux** commands to communicate.

Use the **uulog** command to display a summary of **uucp**, **uuto**, and **uux** command requests by the user or by the system. All of these transactions are logged in files in the **/var/spool/uucp/.Log** directory. The files are named *DaemonName***/***SystemName* where the *DaemonName* directory is named for the daemon involved and the *SystemName* file is named for the remote system the daemon is contacting.

The **uucp** and **uuto** commands call the **uucico** daemon. The **uucico** daemon's activities are logged in the *SystemName* file in the **/var/spool/uucp/.Log/uucico** directory.

The **uux** command calls the **uuxqt** daemon. The **uuxqt** activities are logged in the *SystemName* file in the **/var/spool/uucp/.Log/uuxqt** directory.

You can examine these individual log files by issuing the **uulog** command directly. However, you can also have the BNU program automatically append these temporary log files to a primary log file that you can then examine. This is called *compacting the log files* and is handled by the **uudemon.cleanu** command, a shell script.

Flags

Item	Description
-f*System*	Issues a **tail** command with the **-f** flag on the file transfer log for the specified *System* variable, displaying the end of the log file. Press the Interrupt key to leave the file and return to the prompt.
-s*System*	Displays a summary of copy (**uucico** daemon) requests involving the specified system. **Restrictions:** • System names can contain only ASCII characters. • The **-f** and **-s** flags cannot be combined.

Item	Description
-x	Displays the **uuxqt** daemon log file for the given system.
-*Number*	Displays the last lines of the file. The number of lines is determined by the *Number* variable. (To display the lines, the **uulog** command issues a **tail** command with the **-f** flag for the specified number of lines.)

Examples

1. To display the **uucico** log file for system `hera`, enter:

   ```
   uulog  -shera
   ```

   The output from the command is similar to the following:

   ```
   uucp hera (10/30-10:18:38,3833,0) SUCCEEDED (call to  hera)
   uucp hera (10/30-10:18:39,3833,0) OK (startup)
   jim hera heraN661d (10/30-10:18:39,3833,0) REQUEST
   (nostromo!D.hera661e6c9 --> hera!X.heraN661d (jim))
   jim hera heraN661d (10/30-10:18:40,3833,0) FAILED (CAN'T
   READ /var/spool/uucp/hera/D.hera661e6c9 13)
   uucp hera (10/30-10:18:41,3833,0) OK (conversation
   complete -8)
   ```

   The preceding lines log a conversation between the local system (`nostromo`) and remote system `hera`. The conversation began at `10:18:38` (a.m.) on October 30th, and ended at `10:18:41`. User `jim` attempted to transfer a data file, `D.hera661e6c9`, to system `hera`. The connection to `hera` was successful, but the file could not be transferred because BNU could not read it.
2. To display the **uuxqt** log file, enter:

   ```
   uulog  -x
   ```
3. To display the last forty lines of the file transfer log for system `zeus`, enter:

   ```
   uulog  -fzeus  -40
   ```

Files

Item	Description
/usr/bin/uulog	Contains the **uulog** command.
/var/spool/uucp/.Log	Contain the BNU log files.

uuname Command

Purpose

Provides information about other systems accessible to the local system.

Syntax

uuname [**-c** | **-l**]

Description

The **uuname** command is a Basic Networking Utilities (BNU) command that displays a list of all the computers networked to the local system. This list of accessible systems is displayed on the screen of the local terminal.

In order for a local system to communicate with a remote system by way of BNU, the remote system must:

- Have a UNIX-based operating system.
- Be connected to the local system. (A telephone line can serve as the connection media.)

BNU can be used to communicate between a workstation and an operating system except UNIX, but such communications may require additional hardware or software. The remote systems accessible with BNU commands are identified when the BNU programs are installed and listed in a BNU **Systems** file (by default, the **/etc/uucp/Systems** file, or one or more files specified in the **/etc/uucp/Sysfiles**file).

Before copying a file to another system with the **uuto** or **uucp** command, issue the **uuname** command to determine the exact name of the remote system.

Flags

Item	Description
-c	Displays only the names of systems contained in the **cu Systems** files (configured by the **/etc/uucp/Sysfiles** file). Omission of this flag displays the names of systems contained in the **uucico Systems** files (also configured by the **/etc/uucp/Sysfiles** file). If **/etc/uucp/Sysfiles** is not used to separate **cu** and **uucico** configuration into separate **Systems** files, the names of all systems listed in **/etc/uucp/Systems** are displayed regardless of the **-c** flag.
-l	Displays the name of the local system.

Examples

1. To identify the remote systems connected to the local system, enter:

   ```
   uuname
   ```

   The system responds with a list similar to the following:

   ```
   arthur
   hera
   merlin
   zeus
   ```

2. To identify the name of the local system, enter:

   uuname -l

   The system responds with something similar to the following:

   ```
   nostromo
   ```

Files

Item	Description
/usr/bin/uuname	Contains the **uuname** command.
/etc/uucp/Systems	Lists accessible remote systems.
/etc/uucp/Sysfiles	Specifies alternate files to be used as **Systems** files.
/var/spool/uucp	Contains BNU administrative files.
/var/spool/uucppublic	Contains BNU files awaiting transfer (public directory).

uupick Command

Purpose

Completes the transfer of and handles files sent by the **uuto** command.

Syntax

uupick [**-s***System*]

Description

The **uupick** command is a Basic Networking Utilities (BNU) command that completes the transfer and handles files that the BNU **uuto** command has transmitted to a designated user ID.

Once the copied file is the receive directory, the **rmail** command notifies the recipient that the file has arrived. The recipient then issues the **uupick** command, which searches the public directory on the local system for files sent with some form of the following name:

/var/spool/uucppublic/receive/*User***/***System***/***File*

For each file or directory found, the **uupick** command displays the following message on the screen of the local system:

```
from System: [file File] [dir Directory]
?
```

The question mark prompt (?) following the message indicates you can now enter one of the file-handling options.

Flags

Item	Description
-s *System*	Searches **/var/spool/uucppublic/receive/***User***/***System* for files sent from the specified system. System names contain only ASCII characters.

File-Handling Options

The question mark prompt (?) following a message indicates that one of the following file-handling options should be entered:

Option	Action
!*Command*	Escapes to a shell to run the specified command. After the command executes, the user is automatically returned to the **uupick** command.
*****	Displays all the file-handling options.
a [*Directory*]	Moves all **uuto** files currently in the **receive** directory into a specified directory on the local system. The default is the current working directory. Use a full or relative path name to specify the destination directory.
Ctrl-D	Stops processing and exits from the **uupick** command.
d	Deletes the specified file.
m [*Directory*]	Moves the file to a specified directory. If the *Directory* variable is not specified as a complete path name, a destination relative to the current directory is assumed. If no destination is given, the default is the current working directory on the local system.
new-line	Moves to the next entry in the receive directory when the Enter key is pressed.

Option	Action
p	Displays the contents of the file on the workstation screen.
q	Stops processing and exits from the **uupick** command.

Examples

1. To receive a file sent with the **uuto** command and add it to the current working directory, enter:

   ```
   uupick
   ```

   The system responds with a message similar to:

   ```
   from system anchor: file file1
   ?
   ```

   Enter:

   ```
   a
   ```

   In this example, the `/usr/bin/file1` file sent with the **uuto** command from system `anchor` is added to the current working directory.
2. To receive a file sent with the **uuto** command and add it to a specified directory on your local system, enter:

   ```
   uupick
   ```

   The system responds with a message similar to:

   ```
   from system anchor: file file2
   ?
   ```

   Enter:

   ```
   a /usr/bin1
   ```

   In this example, the `/usr/bin/file2` file sent with the **uuto**command from system `anchor` is added to the `/usr/bin1` directory on the local system.

   > **Note:** The `a /usr/bin1` instruction means move *all* files, not just one. Thus, if any other files are in the `~/anchor/...` directory, they will also be moved.
3. To search for files sent from system `anchor`, enter:

   ```
   uupick -s anchor
   ```

   The system responds with a message similar to:

   ```
   from system anchor: file file1
   ```

Files

Item	Description
/usr/bin/uupick	Contains the **uupick** command.
/var/spool/uucppublic	Contains the BNU public directory.

uupoll Command

Purpose

Forces a poll of a remote BNU system.

Syntax

uupoll [**-g***Grade*] [**-n**] *SystemName*

Description

The **uupoll** command forces the Basic Networking Utilities (BNU) to poll the remote system specified by the *SystemName* parameter. The command is usually run by the **cron** daemon or by a user who wants to force a job to be executed immediately. Otherwise, remote systems are polled by the **uudemon.poll** command at times scheduled in the **/etc/uucp/Poll** file and the **/var/spool/cron/crontabs/uucp** file.

Normally, the **uucico** daemon contacts a remote system only at times specified in the **Poll** file or when there is a job queued for that system. The **uupoll** command queues a null job for the remote system and then invokes the **uucico** daemon. This forces the **uucico** daemon to contact the remote system immediately and attempt to send any jobs which are queued for that system. Use the **-g** flag to specify that only high priority jobs be sent.

Use the **-n** flag to queue the null job without starting the **uucico** daemon. Use this option to:

- Queue a null job before invoking the **uucico** daemon for debugging.
- Queue a null job just before the **uucico** daemon is usually invoked, thus forcing the daemon to poll the specified system.

The *SystemName* parameter is required, and specifies the name of the remote system to be polled.

Flags

Item	Description
-g*Grade*	Instructs the **uupoll** command to send only jobs of the given grade (specified by the *Grade* parameter) or higher on this call. Jobs of a lower grade will remain in the queue until the next time the remote system is polled.
-n	Queues the null job, but does not invoke the **uucico** daemon.

Examples

1. To run the **uupoll** command with the **cron** daemon, place an entry in your **crontabs** file similar to:

   ```
   0  1,7,16  *  *  *  /usr/bin/uupoll hera
   ```

   This polls system `hera` at 0100 hours (1 a.m.), 0700 hours (7 a.m.), and 1600 hours (4 p.m.) daily.
2. If the local system already runs the **uucico** daemon at specific times, you may want to queue a null job just before the **uucico** daemon normally runs. For example, if your system runs the **uucico** daemon hourly, place an entry similar to the following in your **crontabs** file:

   ```
   0  1,7,16  *  *  *  /usr/bin/uupoll  -n zeus
   0  5,12,21 *  *  *  /usr/bin/uupoll -n hera
   5  *       *  *  *  /usr/sbin/uucp/uucico -r1
   ```

   This queues null jobs for the remote sites on the hour, and they are processed by the **uucico** daemon when it runs at 5 minutes past the hour.
3. To force the **uucico** daemon to transfer all jobs of grade `N` or higher for system `zeus`:

```
uupoll  -gN zeus
```

Files

Item	Description
/usr/bin/uupoll	Contains the **uupoll** command.
/etc/uucp/Poll	Specifies when the BNU program should poll remote systems to initiate tasks.
/var/spool/cron/crontabs/uucp	Schedules automatic polling of remote systems.
/var/spool/uucp/*SystemName*	Contain files to be transferred to remote systems.

uuq Command

Purpose

Displays the BNU job queue and deletes specified jobs from the queue.

Syntax

uuq [**-l** | **-h**] [**-s***SystemName*] [**-u***User*] [**-d***JobNumber*] [**-r***SpoolDir*] [**-b***BaudRate*]

Note: Only a user with root authority can use the **-d** flag.

Description

The **uuq** command is used to list or delete job entries in the Basic Networking Utilities (BNU) job queue.

When listing jobs, the **uuq** command uses a format similar to that used by the **ls** command. In the default format, the **uuq** command lists only the job numbers of the jobs waiting in the queue, followed by a summary line for each system.

In summary format (**uuq -h**) only the summary lines are listed. Summary lines give:

- System name
- Number of jobs for the system
- Total number of bytes to send

In the long format (**uuq -l**), which can be quite slow, the information listed for each job is:

- Job number
- Number of files to transfer
- User who sent the job
- Number of bytes to be sent
- Type of job requested:

Item	Description
S	Sending a file
R	Receiving a file
X	Executing a command on the remote system

- File to be sent or received or the command to be executed

A user with root authority can use the **-d***JobNumber* flag to delete jobs from the queue after running a **uuq** listing to discover the job numbers.

Flags

Item	Description
-b*BaudRate*	Uses the baud rate given, instead of the default (1200 baud), to compute the transfer time.
-d *JobNumber*	Deletes the job designated by the *JobNumber* variable from the BNU queue. Only someone with root authority can delete jobs from the queue.
-h	Shows only the summary lines for each system.
-l	Lists the output in the long format.
-s*SystemName*	Lists only jobs for systems whose system names begin with the string specified in the *SystemName* variable.
-r *SpoolDir*	Searches for files in the spooling directory designated by the *SpoolDir* variable, instead of in the default spooling directory.
-u*User*	Lists only jobs queued by users whose login names begin with the string specified in the *User* variable.

Examples

1. To get a long listing of all jobs spooled for system `hera`, type:

   ```
   uuq -l -shera
   ```

2. To get a summary listing for all systems, type:

   ```
   uuq -h
   ```

3. To delete a job for user `nita` from the queue, first use the **uuq** command to find the number of the job you want to delete, as follows:

   ```
   uuq -l -unita
   ```

   This produces a list of jobs spooled for user `nita`. Find the job you wish to remove. If its job number is `13451`, for example, the following command will delete the job:

   ```
   uuq -d13451
   ```

   Note: You must have root authority or be logged in as **uucp** to delete jobs from the queue.

Files

Item	Description
/usr/bin/uuq	Contains the **uuq** command.
/var/spool/uucp/*SystemName*	Contains spool files for the remote system designated by *SystemName*.
/var/spool/uucp/*SystemName***/C.***	Contain instructions for file transfers.
/var/spool/uucp/*SystemName***/D.***	Contain information about data files to be transferred.
/var/spool/uucp/*SystemName***/X.***	Contain instructions for executing remote commands.

uusched Daemon

Purpose

Schedules work for the Basic Networking Utilities (BNU) file transport program.

Syntax

uusched [**-u***DebugLevel*] [**-x***DebugLevel*]

Description

The **uusched** daemon schedules work for the Basic Networking Utilities (BNU) file transport program. It schedules the transfer of files that are queued in the **/var/spool/uucp/***SystemName* directory. The scheduling daemon first randomizes the work and then starts the **uucico** daemon, which transfers the files.

The **uusched** daemon is usually started by the **uudemon.hour**command, a shell procedure, which is run periodically by the **cron** daemon based on instructions from the **/var/spool/cron/crontabs/uucp** file.

The **uusched** daemon can also be started from the command line for debugging purposes.

> **Note:** Either you must be in the **/usr/sbin/uucp** directory when you start the **uusched** daemon, or you must start the daemon with the full path name, **/usr/sbin/uucp/uusched.**

Flags

Item	Description
-u*DebugLevel*	Passes as the **-x***DebugLevel* flag to the **uucico** daemon. The *DebugLevel* variable is a number from 0 to 9, with a default of 5. Higher numbers give more detailed debugging information, which is displayed on the screen of the local system.
-x*DebugLevel*	Outputs debugging messages from the **uusched** daemon. The *DebugLevel* variable is a number from 0 to 9, with a default of 5. Higher numbers give more detailed debugging information, which is displayed on the screen of the local system.

Example

To start the **uusched** daemon from the command line, enter:

```
/usr/sbin/uucp/uusched &
```

This starts the **uusched** daemon as a background process. (Note that the path name is included in the command.)

Files

Item	Description
/etc/locks/*	Contains lock files that prevent multiple uses of devices and multiple calls to systems.
/usr/sbin/uucp/*	Contains the **uusched** daemon and the BNU configuration files.
/etc/uucp/Devices	Contains information about available devices.
/etc/uucp/Maxuuscheds	Limits scheduled jobs.
/etc/uucp/Systems	Describes accessible remote systems.

Item	Description
/var/spool/cron/crontabs/uucp	Schedules BNU jobs for the **cron** daemon, including the **uudemon.hour** shell procedure.
/var/spool/uucp/*SystemName* **/***	Contain files waiting to be transferred.

uusend Command

Purpose

Sends a file to a remote host.

Syntax

uusend [**-m***Mode*] [**-r**] *Sourcefile System* [**!***System* ...] **!** *RemoteFile*

Description

The **uusend** command sends a file to a given location on a remote system. The remote system need not be directly connected to the local system, but a chain of UUCP links must connect the two systems, and the **uusend** command must be available on each system in the chain.

The chain of systems is given by the *System*[**!***System* ...] parameter, which lists each remote system the file is to be transferred to, separated by ! (exclamation points). The **!***Remotefile* parameter gives the name under which the file is to be stored when it reaches the last system in the chain.

> **Note:** Do not put any spaces between the system names and exclamation points or between the last exclamation point and the remote file name.

The *SourceFile* parameter specifies the name of the file on the local system. If a **-** (dash) is used, the **uusend** command uses standard input.

Flags

Item	Description
-m *Mode*	Specifies that the mode of the file on the remote system will be taken from the octal number given. If this flag is not specified, the mode of the input file will be used.
-r	Prevents the starting of the **uucico** daemon, which transfers files between systems. The default is to start the **uucico** daemon.

The flags are primarily used internally by the **uusend** command when it is transferring files to the next remote system in the chain.

Example

To send a file across one system to another system, enter:

```
uusend /etc/motd nostromo!gandalf!~nuucp
```

The `/etc/motd` file is sent to system `nostromo` and then to system `gandalf`, and placed in `nuucp`'s home directory, `/var/spool/uucppublic/nuucp`, where `nuucp` is a BNU login ID.

Files

Item	Description
/usr/bin/uusend	

Item	Description
	Contains the **uusend** command.

uusnap Command

Purpose

Displays the status of BNU contacts with remote systems.

Syntax

uusnap

Description

The **uusnap** command displays a table showing the status of the Basic Networking Utilities (BNU). The table includes the following information for each remote system:

Item	Description
`SystemName`	Specifies the name of the remote system.
`Number Cmds`	Specifies the number of command files (**C.*** files) queued for the remote system.
`Number Data`	Specifies the number of data transfers (**D.*** files) queued for the remote system.
`Number Xqts`	Specifies the number of remote command executions (**X.*** files) queued for the remote system.
`Message`	Specifies the current status message for the site, from the **/var/spool/uucp/.Status/***SystemName* file. The `Message` field may include the time remaining before BNU can retry the remote system, and the count of the number of times (if any) BNU has tried unsuccessfully to reach the system.

Example

To see a snapshot of the status of BNU, enter:

```
uusnap
```

The output from this command is similar to the following:

```
nostromo  4 Cmds  2 Data  2 Xqts  SUCCESSFUL
zeus      2 Cmds  1 Data  2 Xqts  NO DEVICES AVAILABLE
```

These lines indicate that four command files, two data files, and two execute files are currently queued for system `nostromo`. The last connection to `nostromo` was successful. The last attempt to contact system `zeus`, on the other hand, was not successful because no device was available on the local system.

Files

Item	Description
/usr/bin/uusnap	Contains the **uusnap** command.
/var/spool/uucp/.Status/*SystemName*	Records the status of BNU contacts with a remote system.
/var/spool/uucp/*SystemName*	Contains **C.***, **D.***, and **X.*** files to be transferred by the **uucico** daemon.

Item	Description
/var/spool/uucp/*SystemName***/C.***	Instruct BNU about files to be transferred.
/var/spool/uucp/*SystemName***/D.***	Contain files to be transferred by BNU.
/var/spool/uucp/*SystemName***/X.***	Specify commands to be remotely executed by BNU.

uustat Command

Purpose

Reports the status of and provides limited control over BNU operations.

Syntax

uustat [[**-n** *Number*] [**-a** | **-k** *JobID* | **-m** | **-p** | **-q** | **-r** *JobID*] | [**-s** *System*] [**-u** *User*]]

Description

The **uustat** command is a Basic Networking Utilities (BNU) command that displays status information about several types of BNU operations. It is particularly useful in monitoring the status of BNU requests.

In addition, the **uustat** command also gives a user limited control over BNU jobs queued to run on remote systems. By issuing the command with the appropriate flag, a user can check the general status of BNU connections to other systems and cancel copy requests made with the **uucp** and **uuto** commands.

If the **uustat** command is issued without any flags, the command reports the status of all BNU requests issued by the current user since the last time the holding queue was cleaned up. Such status reports are displayed in the following format:

```
jobid date/time status system_name user_ID size file
```

There are two types of BNU queues:

- The current queue, accessed with the **-q** flag, lists the BNU jobs either queued to run on or currently running on one or more specified computers.
- The holding queue, accessed with the **-a** flag, lists all jobs that have not executed during a set period of time.

  After the time has elapsed, the entries in the holding queue are deleted either manually with the BNU **uucleanup** command or automatically by commands such as **uudemon.cleanu** started by the **cron** daemon.

When sending files to a system that has not been contacted recently, it is a good idea to use the **uustat** command to see when the last access occurred; the remote system may be down or out of service.

Flags

The following flags are mutually exclusive. Use only one at a time with the **uustat** command.

Item	Description
-a	Displays information about all the jobs in the holding queue, regardless of the user who issued the original BNU command.

Item	Description
-k*JobID*	Cancels the BNU process specified by the *JobID* variable. The person using this flag must either be the one who made the **uucp** request now being canceled or be operating with root authority. This flag cancels a process only when that job is still on the local computer. After BNU has moved the job to a remote system for execution, the **-k** *JobID* flag cannot be used to cancel the remote job.
-m	Reports the status of the most recent attempt to contact the specified system with a BNU command. If the BNU request was completed, the status report is successful. If the job was not completed, the status report is an error message saying that the login failed.
-n *Number*	Allows the user to specify the amount of machines from which to collect BNU status information. The amount specified should be greater than or equal to the amount of machines in the Systems file. The default is 200.
-p	Runs a **ps -flp** (process status: full, long list of specified process IDs) for all PID numbers in the lock files.
-q	Lists the jobs currently queued to run on each system. These jobs are either waiting to execute or in the process of executing. If a status file exists for the system, its date, time, and status information are reported. When the job is finished, BNU removes that job listing from the current queue. In a status report, a number in parentheses next to the number of a **C.*** (command) file or an **X.*** (execute) file represents the age in days of the oldest **C.*** or **X.*** file for that system. The **retry** field represents the number of times BNU tried and failed to execute the command because of, for example, a failed login, locked files, or an unavailable device.
-r*JobID*	Marks the files in the holding queue specified by the *JobID* variable with the current date and time. Use this flag to ensure that a cleanup operation does not delete files until the job's modification time reaches the end of the specified period. You can use either one or both of the following flags with the **uustat** command:
-s *System*	Reports the status of BNU requests for the workstation specified by the *System* variable. The *System* name can contain only ASCII characters.
-u *User*	Reports the status of BNU requests by the user specified by the *User* variable, for any workstation. The *User* name can contain only ASCII characters.

Examples

1. To display the status of all BNU jobs in the holding queue, type:

   ```
   uustat -a
   ```

   The system responds with a message similar to the following:

   ```
   heraC3113   11/06-17:47 S  hera    amy  289   D.venus471afd8
   zeusN3130   11/06-09:14 R  zeus    geo  338   D.venus471bc0a
   merlinC3120 11/05-16:02 S  merlin  amy  828   /home/amy/tt
   merlinC3119 11/05-12:32 S  merlin  msg  rmail amy
   ```

Field	Description
1	Job ID of the operation
2	Date and time the BNU command was issued
3	An S or an R, depending on whether the job is to send or receive a file

Field	Description
4	Name of the system on which the command was entered
5	User ID of the person who issued the command
6	Size of the field or the name of the remote command
7	Name of the file.

When the size of the file is given, as in the first three lines of the example output, the file name is also displayed. The file name can be either the name given by the user, as in the `/home/amy/tt` entry, or a name that BNU assigns internally to data files associated with remote executions, such as `D.venus471afd8`.

2. To display the status of all jobs in the current queue, type:

   ```
   uustat  -q
   ```

   The system responds with a message similar to the following:

   ```
   merlin  3C       07/15-11:02   NO DEVICES AVAILABLE
   hera    2C       07/15-10:55   SUCCESSFUL
   zeus    1C (2)   07/15-10:59   CAN'T ACCESS DEVICE
   ```

   This output tells how many **C.*** (command) files are waiting for each system. The number in parentheses `(2)` in the third line of the example indicates that the **C.*** file has been in the queue for two days. The date and time refer to the current interaction with the system, followed by a report of the status of the interaction.

3. To display all process IDs in the lock file, type:

   ```
   uustat  -p
   ```

   The system responds with a message similar to the following:

   ```
   LCK..tty0:  881
   LCK.S.0:  879
   LCK..hera:  881
   F   S UID  PID  PPID  C  PRI NI ADDR SZ  WCHAN    STIME    TTY
   101 S uucp 881  879   26 39  39 370  296 3fffe800 09:57:03  -
   TIME  COMD
   0:00  UUCICO  -r1  -shera
   101 S uuc  879  1     11 33  39 770  156 8d874    09:57:02  -
   0:00  /usr/sbin/uucp/uusched
   ```

4. To cancel a job in the current queue, first determine its job ID and then issue the command to cancel the job. To determine the job ID, type:

   ```
   uustat  -a
   ```

   The system responds with a message similar to the following:

   ```
   heraC3113   11/06-17:47  S  hera    amy  289  D.venus471afd8
   merlinC3119 11/06-17:49  S  merlin  geo  338  D.venus471bc0a
   ```

   To cancel the job with the ID of `heraC3113`, type:

   ```
   uustat  -k heraC3113
   ```

5. To report the status of jobs requested by system `hera`, type:

   ```
   uustat  -s hera
   ```

   The system responds with a message similar to the following:

   ```
   heraN1bd7  07/15-12:09  S  hera  amy  522    /usr/amy/A
   heraC1bd8  07/15-12:10  S  hera  amy   59    D.3b2a12ce4924
   heraC3119  07/15-12:11  S  hera  amy  rmail   msg
   ```

Files

Item	Description
/etc/locks	Contains lock files to prevent multiple uses of devices.
/usr/bin/uustat	Specifies the command pathname.
/var/spool/uucp	Contains BNU status information.

uuto Command

Purpose

Copies files from one system to another.

Syntax

uuto [**-m**] [**-p**] *Source ... User*

Description

The **uuto** command is a Basic Networking Utilities (BNU) command that copies one or more *Source* files from one system to a specified *User* on another UNIX based system. This program uses the **uucp** command for the actual file transfer, but the **uuto** command enables the recipient to use the **uupick** command options to handle the transferred file on the local system.

The sender issues the **uuto** command to copy one or more files to a specific user ID on another system. The **uucp** command then copies the file to the BNU public directory, **/var/spool/uucppublic**, on the destination system. The **uucp** command also creates an additional subdirectory called **receive** (if it does not already exist) and directories below it in which to hold the files until the recipient retrieves them with the **uupick** command. The full path names to the copied files are some form of the following name:

```
/var/spool/uucppublic/receive/UserName/System/File
```

where the *UserName* and *System* directories are created based on the *User* parameter given with the **uuto** command.

Once the copied file is in the **receive** directory, the **rmail** command notifies the recipient that a file has arrived. The recipient then issues the **uupick** command, and this command searches the public directory for files sent to the recipient and notifies the recipient about each file it locates. The recipient then enters one of the **uupick** options to handle the file.

Source and Destination File Names

The sender must give the name of the file to be sent and user and system to which the file is to be transferred. The *Source* parameter is the path name of the source file. This can be the name of the file if the file is in the directory from which the **uuto** command is issued. If the file is in a different directory, the complete or relative path name of the file must be given.

The *User* parameter is the path name to the specific location where the source file is to be copied. This path name must include the user identification of the person the file is being sent to. The *User* parameter has the form:

```
System!UserName
```

where *System* is the name of the remote system connected to the local system, and *UserName* is the login name of the recipient of the transferred files on the specified system.

When copying a file from one user to another user on the local system, omit the *System* entry; the destination is the ID of the user to whom the file is being sent. System names can contain only ASCII characters.

Flags

Item	Description
-m	Notifies the sender by the **bellmail** command when the source file has been successfully copied.
-p	Copies the source file to the spool directory on the local system. The source file resides in the spooling directory for a set period of time (defined in the **uusched** program) before the **uucp** command calls the **uucico**daemon, which actually transfers the copy to the public directory on the specified remote system. The default is to transfer a source file directly to the specified user.

Examples

1. To copy a file to a user on a remote system, enter:

   ```
   uuto /home/bin/file1 zeus!karen
   ```

   In this example, the `/home/bin/file1` file is sent to user `karen` on the remote system `zeus.`
2. To copy a file to a user on a remote system and be notified whether the source file was successfully copied, enter:

   ```
   uuto  -m /home/bin/file2 zeus!karen
   ```

   In this example, the `/home/bin/file2` file is sent to user `karen` on the remote system `zeus` and a message is returned to the sender verifying that the copy was successful.
3. To copy a file to another user on your local system, enter:

   ```
   uuto /home/bin/file3 ron
   ```

   In this example, the `/home/bin/file3` file is sent to user `ron` on the local system. No mail message is sent to the recipient in a local transfer.

Files

Item	Description
/usr/bin/uuto	Contains the **uuto** command.
/var/spool/uucppublic	Is the BNU public directory.

uutry Command

Purpose

Contacts a specified remote system with debugging turned on and allows the user to override the default retry time.

Syntax

uutry [**-x***DebugLevel*] [**-r**] SystemName

Description

The **uutry** command contacts a remote system, specified by the *SystemName* parameter, using debugging mode. Debugging mode provides a means of monitoring Basic Networking Utilities (BNU) connections to remote computers and file transfers. The **uutry** command calls the **uucico** daemon to contact the remote system.

The debugging output is scrolled on the screen of the local system. Once the system has finished displaying this information, press the Interrupt key to return to the prompt.

The **-r** flag overrides the default retry time if the first attempt to contact the remote system is unsuccessful. The default retry time is 5 minutes.

The *SystemName* parameter, which is required, specifies the name of the remote system you wish to contact.

Requirement: Either you must be in the **/usr/sbin/uucp** directory when you issue the **uutry** command or you must issue the command with the full path name, **/usr/sbin/uucp/uutry.**

Tips:

- The **uutry** command is a shell script stored in the **/usr/lib/uucp** directory.
- If the debugging output scrolls too quickly to be read, use the **Uutry** command to save the output in a temporary file.

Flags

Item	Description
-r	Overrides the default retry time. If for some reason the **uucico** daemon cannot complete the requested connection, the daemon waits for a set amount of time and tries again. The default retry time is 5 minutes. **Note:** The time at which the remote system was last polled is recorded in the *SystemName* file in the **/var/spool/uucp/.Status** directory.
-x*DebugLevel*	Overrides the default amount of detail in the debugging information that the **uutry** command displays on the screen. The valid range for the *DebugLevel* variable is 0 to 9, with a default of 5. Higher numbers cause the final report to be more detailed. If the **-x** flag is not used, the **uucico** daemon is started with the default level, which produces a moderate amount of information.

Examples

1. To change the amount of detail the **uutry** command provides about the progress of the **uucico** operation, use the **-x** flag to specify a different debugging level. For example, entering:

   ```
   /usr/sbin/uucp/uutry  -x9 venus
   ```

   instructs the **uutry** command to generate as much information as possible about the way in which the **uucico** daemon is working.
2. The default time at which to retry a contact to a remote system when the first contact was unsuccessful is 5 minutes. To shorten the default retry time for contacting the remote system, enter:

   ```
   /usr/sbin/uucp/uutry  -r venus
   ```

   Using the **-r** flag instructs the **uucico** daemon to contact remote system `venus`, overriding the default retry time. The daemon attempts to contact system `venus`, retrying periodically until the connection is successful, and then produces debugging output on the display screen of the local system.

Files

Item	Description
/usr/sbin/uucp/uutry	Contains the **uutry** command.
/etc/uucp/Devices	Contains information about available devices.
/etc/uucp/Dialcodes	Contains dial-code abbreviations.

Item	Description
/etc/uucp/Dialers	Specifies initial handshaking on a connection.
/etc/uucp/Permissions	Describes access permissions for remote systems.
/etc/uucp/Systems	Describes accessible remote systems.
/var/spool/uucp/.Status/*SystemName*	Lists the last time the remote system named by the *SystemName* file was contacted.
/var/spool/uucppublic/*	Contain the BNU public directories.

uux Command

Purpose

Runs a command on another UNIX-based system.

Syntax

uux [**-c** | **-C**] [**-n** | **-z**] [**-**] [**-a***Name*] [**-b**] [**-g***Grade*] [**-j**] [**-p**] [**-e**] [**-r**] [**-s***File*] [**-x***DebugLevel*] *CommandString*

Description

The **uux** command is a Basic Networking Utility (BNU) that runs a specified command on a specified UNIX-based system while enabling the user to continue working on the local system. Before running the requested command, the **uux** command gathers any necessary files from the designated systems. The user can direct the output from the command to a specific file on a specific system. For security reasons, many installations permit the **uux** command to run only the **rmail** command.

The **uux** commands on other systems create execute (**X.***) files that run commands on the local system. In addition, the **uux** command on the local system creates both command (**C.***) files and data (**D.***) files for transfer to other systems. Execute files contain the command string to be executed on the designated system. Command files contain the same information as those created by the **uucp** command. Data files either contain the data for a remote command execution or else become **X.*** files on remote systems for remote command executions.

The full path name of an execute file is a form of the following:

```
/var/spool/uucp/System/X.SystemNxxxx
```

After creating the files in the spooling directory, the **uux** command calls the **uucico** daemon to transfer the files from the spooling directory on the local system to the designated remote system. Once the files are transferred, the **uuxqt** daemon on the remote system executes the *CommandString* on the specified system, placing any output from the command in the file designated by the original **uux** command request.

The *CommandString* argument is made up of one or more arguments that look like an operating system command line, except that *CommandString* argument may be prefixed by the name of the remote system in the form *System***!**. The default *System* is the local system. Unless the user entering the **uux** command includes the **-n** flag, the command notifies that user if the remote system does not run the command. This response comes by mail from the remote system.

Source and Destination File Names

- When specifying the destination of the output of a command, the **uux** command can be entered in either one of the following formats:

  - **uux** [*Options*] "*CommandString*> *Destination*"
  - **uux** [*Options*] *CommandString*\ *{Destination*\}.
- Destination names can be either of the following:
  - A full path name
  - A full path name preceded by **~***User*, where *User* is a login name on the specified system. The **uux** command replaces this path name with the user's login directory.
- The shell pattern-matching characters ? (question mark), * (asterisk), and [...] (brackets) can be used in the path name of a source file (such as files compared by the **diff** command); the appropriate system expands them. However, using the * character may occasionally produce unpredictable or unanticipated results. Shell pattern-matching characters should not be used in the destination path name.
- Place either two backslashes (\ . . . \) or a pair of quotation marks (" . . . ") around pattern-matching characters in a path name so the local shell cannot interpret them before the **uux** command sends the command to a designated system.
- If you are using the special shell characters > (greater than), < (less than), ; (semicolon), or | (vertical bar) in a path name, place either \ . . . \ or " . . . " around the individual character or around the entire command string.
- Do not use the shell redirection characters << or >> in a path name.
- The **uux** command attempts to move all files specified on the command line to the designated system. Enclose the names of all output files in parentheses so that the **uux** command does not try to transfer them.
- When specifying a *System*, always place it before the *CommandString* argument in the entry. System names can contain only ASCII characters.
- The ! (exclamation point) preceding the name of the local system in a command is optional. If you choose to include the ! to run a command on the local system using files from two different remote systems, use ! instead of *System*! to represent the local system, and add *System*! as the first entry in any path name on the remote systems.
- The exclamation point representing a system in BNU syntax has a different meaning in C shells. When running the **uux** command in a C shell, place a \ (backslash) before the exclamation point in a system name.

  Note: The notation ~ (tilde) is the shorthand way of specifying the public spooling directory, **/var/spool/uucppublic**.

Flags

Item	Description
-	Makes the standard input to the **uux** command the standard input to the *CommandString* argument.
-a*Name*	Replaces the user ID of the person issuing the command with the user ID specified with the *Name* variable.
-b	Returns standard input to the command if the exit status is not zero.
-c	Transfers the source files to the destination on the specified system. The source files are copied into the spooling directory, and the **uucico** daemon is invoked immediately. This flag is the default.

Item	Description
-C	Transfers the source files to the spool directory. After a set period of time (specified in the **uusched** program), the **uucico** daemon attempts to transfer the files to the destination on the specified computer. Occasionally, there are problems in transferring a source file; for example, the remote computer may not be working or the login attempt may fail. In such cases, the file remains in the spool directory until it is either transferred successfully or removed by the **uucleanup** command.

Item	Description
-e	Enables file expansion.
-g*Grade*	Specifies when the files are to be transmitted during a particular connection. The *Grade* variable specifies a single number (0 through 9) or letter (A through Z, a through z); lower ASCII-sequence characters cause the files to be transmitted earlier than do higher sequence characters. The number 0 is the highest (earliest) grade; z is the lowest (latest). The default is **N**.
-j	Displays the job identification number of the process that is running the command on the specified system. Use this job ID with the BNU **uustat** command to check the status of the command or with the **uustat -k** flag to terminate the process.
-n	Prevents user notification by the **mail** command of the success or failure of a command. The default is to notify the user if the command fails.
-p	Uses the standard input to the **uux** command as the standard input to the *CommandString* argument. A **-** (minus) has the same effect.
-r	Prevents the starting of the spooling program that transfers files between systems. The default is to start the spooling program.
-s*File*	Reports the status of the transfer in a file specified by the *File* variable on the designated system. File names can contain only ASCII characters.
-x*DebugLevel*	Displays debugging information on the screen of the local system. The *DebugLevel* variable must be a number from 0 to 9. A higher number gives a more detailed report.
-z	Notifies the user if the command completes successfully. This flag is the opposite of the system default, which is to notify the user only in the event of a failure.

Examples

1. To run the **qprt** command on a remote system, enter:

```
uux merlin!qprt /reports/memos/lance
```

   In this example, the remote file `/reports/memos/lance` is printed on remote system `merlin`. Since neither the **-n** nor **-z** flag is specified, the **uux** command notifies the user only if the remote system fails to run the command. The response comes by the **mail** command from the remote system.

2. To run commands on two remote systems, enter the information on separate command lines:

```
uux merlin!qprt /reports/memos/lance
uux zeus!qprt /test/examples/examp1
```

   In this example, the remote `/reports/memos/lance` file is printed on remote system `merlin`, and the remote `/test/examples/examp1` file is printed on remote system `zeus`. Since neither the **-n** nor **-z** flag is specified, the **uux** command notifies the user only if the remote system fails to run the command. The response comes by the **mail** command from the remote system.

3. To queue a job that compares a file on the local system with a file on a remote system, using the **diff** command on the local system, and get the job ID of the job, enter:

   ```
   uux  -j "/usr/bin/diff /usr/amy/f1 hera!/home/amy/f2 > ~/f1.diff"
   ```

   In this example, the `/usr/amy/f1` file on the local system is compared to the `/home/amy/f2` file on the remote system `hera` and the output is placed in the `f1.diff` file in the local public directory (the full path name of this file is `/var/spool/uucppublic/f1.diff`). The destination name must be entered either preceded by a > with the whole command string enclosed in **" "** (quotation marks) or entered enclosed in braces and backslashes, as **\{** *DestinationName* **\}**. The **-j** flag causes the **uux** command to return the BNU job ID of the job.

4. To use the **diff** command on the local system to compare files that are located on two different remote systems, enter:

   ```
   uux "!/usr/bin/diff hera!/usr/amy/f1 venus!/home/amy/f2 > \ !f1.diff"
   ```

   In this example, the `/usr/amy/f1` file from the remote system `hera` is compared to the `/home/amy/f2` file from the remote system `venus` and the output is placed in the file `f1.diff`, located in the current working directory on the local system.

   The output file must be write-enabled. If you are uncertain about the permission status of a specific target output file, direct the results to the public directory. The exclamation points representing the local system are optional. The destination name must be entered either preceded by a > with the whole command string enclosed in **" "** (quotation marks) or entered enclosed in braces and backslashes, as **\{** *DestinationName* **\}**.

5. To execute the **diff** command on two separate files from different systems, enter:

   ```
   uux "hera!/usr/bin/diff /tmp/out1 zeus/tmp/out2 > ~/DF"
   ```

   In this example, the `diff` file is on the remote system `hera`. The first source file is on the remote system `hera`, and the secondfile is on the system `zeus`. (`zeus` may be the local system or another remote system.) The output is directed to the file DF in the public directory on the local system.

6. To specify an output file on a different remote system, enter:

   ```
   uux hera!uucp venus!/home/amy/f1 \{merlin!/home/geo/test\}
   ```

   In this example, the **uucp**“uucp Command” on page 4411 command is run on the remote system `hera`, and the `/home/amy/f1` file, stored on system `venus`, is sent to user `geo` on system `merlin` as `test`. The destination name is entered enclosed in braces and backslashes.

7. To get selected fields from a file on a remote system and place them in a file on the local system, enter:

   ```
   uux "cut -f1 -d: hera\!/etc/passwd > ~/passw.cut"
   ```

   cut command is run on the local system. The first field from each line of the password file on system `hera` is placed in the `passw.cut` file in the public directory on the local system. The **uux** command is running in a C shell, so a \ (backslash) must precede the exclamation point in the name of the remote system.

8. To use the **uux** piping option to specify a remote copy of the `/tmp/example` file to `/tmp/examplecopy` on system `mercury` use the following syntax:

   ```
   uux -p mercury!
   cp /tmp/example /tmp/examplecopy
   ```

   The user must enter a Ctrl-D in order to terminate the command input. After Ctrl-D is pressed, the command will be spooled for remote execution on system `mercury`.

Files

Item	Description
/usr/bin/uux	Contains the **uux** command.
/var/spool/uucp	Is the spooling directory.
/var/spool/uucppublic	Is the public directory.

uuxqt Daemon

Purpose

Executes Basic Networking Utilities (BNU) remote command requests.

Syntax

uuxqt [**-e**] [**-s***SystemName*] [**-x***DebugLevel*]

Description

The Basic Networking Utilities (BNU) **uuxqt** daemon executes commands on designated remote systems.

The **uuxqt** daemon on each networked system periodically searches the spool directory for remote execute (**X.***) files. These files are sent to the directory by the **uucico** daemon in response to a **uux** command.

When it finds **X.*** files, the **uuxqt** daemon checks each file to make sure that:

- All the required data (**D.***) files are available.
- The requesting system has the necessary permissions to access the data files and run the requested commands.

  Note: The **uuxqt** daemon uses the **/etc/uucp/Permissions** file to validate file accessibility and command execution permission.

If the data files are present and the requesting system has the appropriate permissions, the **uuxqt** daemon executes the commands.

Note: The **uuxqt** command is usually executed from the **uudemon.hour** command, a shell procedure, and not entered from the command line. You must have root user privileges to issue the **uuxqt** command from the command line.

Flags

Item	Description
-e	Enables file expansion.
-s*SystemName*	Designates the remote system to be contacted. Use only when starting the **uuxqt** command manually. The system name is supplied internally when the **uuxqt** command is started automatically. **Note:** System names can contain only ASCII characters.
-x*DebugLevel*	Displays debugging information on the screen of the local system. The *DebugLevel* variable is a single digit between 0 and 9, with a default of 5. The higher the *DebugLevel* variable, the more detailed the debugging information.

Security

Access Control: You must have root authority to start the **uuxqt** daemon from the command line.

Example

To start the **uuxqt** daemon for debugging, enter:

```
/usr/sbin/uucp/uuxqt  -svenus  -x7
```

This instructs the command to contact remote system `venus` and provide fairly detailed information about the contact.

Files

Item	Description
/usr/sbin/uucp/uuxqt	Contains the **uuxqt** daemon.
/etc/locks	Contains lock files that prevent multiple uses of devices and multiple calls to systems.
/etc/uucp/Maxuuxqts	Limits remote command executions.
/etc/uucp/Permissions	Describes access permissions for remote systems.
/var/spool/uucp/*	Contain the execute and data files.

v

The following AIX commands begin with the with the letter *v*.

vacation Command

Purpose

Returns a message to the sender that the mail recipient is on vacation.

Syntax

vacation [{ **-I** | *User* }] | [{ **-f** *Number* [*Unit*] | *User* }]

Description

The **vacation** command returns a message to the sender of a mail message to notify the sender that the recipient is on vacation. The intended use is in a **$HOME/.forward** file that allows messages to come to you while also sending a message back to the sender.

Note: Sendmail version 8.9.3 and subsequent releases have a security enhancement that will ignore the **.forward** file if *either* of the following conditions exist:

- The **.forward** file has group or world writeable permissions
- Any of **.forward** file's parent directories have group or world writable permissions

If you think that the vacation program is not working because the **.forward** file is being ignored, check the permissions. If you must have group or world writeable permissions on any of the parent directories of the **.forward** file, then set the DontBlameSendmail option in the sendmail configuration file with the appropriate values.

The **vacation** command expects a **$HOME/.vacation.msg** file containing a message to be sent back to each sender. If this file does not exist, the **vacation** command looks for **/usr/share/lib/vacation.def**, a systemwide default vacation message file. It should be an entire message, including any desired headers, such as `From` or `Subject`. By default, this message is sent only once a week to each person who sends mail to you. Use the **-f** flag to change the frequency intervals at which the message is sent. The names of the people who send messages are kept in the files **$HOME/.vacation.pag** and **$HOME/.vacation.dir**. These files are created when the **vacation** command is initialized for your user ID using the **-I** (uppercase i) flag.

If the **-I** flag is not specified, the **vacation** command reads the first line from the standard input for a `From` line to determine the sender. If no text is available from standard input, the command returns an error message. All properly formatted incoming mail should have a `From` line. No message is sent if the `From` header line indicates that the message is from `Postmaster`, MAILER-DAEMON, or if the initial `From` line includes the string-`REQUEST@` or if a `Precedence: bulk` or `Precedence: junk` line is included in the header.

Flags

Item	Description
-I	Initializes the **$HOME/.vacation.pag** and **$HOME/.vacation.dir** files. Execute the **vacation** command using this flag before you modify your **$HOME/.forward** file.

Item	Description
-f*Number* **[***Unit* **]**	Specifies the frequency interval at which the vacation message is sent. The *Number* parameter is an integer value and the *Unit* parameter specifies a time unit. The *Unit* parameter can be one of the following: **s** Seconds **m** Minutes **h** Hours **d** Days **w** Weeks

Note: The **-f** flag cannot be used with the **-I** flag.

Examples

1. Before you use the **vacation** command to return a message to the sender saying that you are on vacation, you must initialize the **$HOME/.vacation.pag** and **$HOME/.vacation.dir** files. To initialize these files, type:

   ```
   vacation -I
   ```

2. Modify the **.forward** file. For example, Mark types the following statement in the **.forward** file:

   ```
   mark,|"/usr/bin/vacation mark"
   ```

   The sender receives the message that is in the **$HOME/.vacation.msg** file, or if the file does not exist, the default message found in the **/usr/share/lib/vacation.def** file. If neither of these files exist, no automatic replies are sent to the sender of the mail message and no error message is generated. If either of these files exist, the sender receives one vacation message from `mark` per week, regardless of how many messages are sent to `mark` from the sender.

3. If the following entry is contained in your **.forward** file,

   ```
   mark, |"/usr/bin/vacation -f10d mark"
   ```

   The sender receives one vacation message from `mark` every ten days, regardless of how many messages are sent to `mark` from the sender.

4. To create a vacation message that is different from the default vacation message, create the file **$HOME/.vacation.msg** and add your message to this file. The following is an example of a vacation message:

   ```
   From: mark@odin.valhalla (Mark Smith)
   Subject: I am on vacation.
   Delivered-By-The-Graces-Of: the Vacation program
   I am on vacation until October 1. If you have something urgent,
   please contact Jim Terry <terry@zeus.valhalla>.
      --mark
   ```

5. To cancel the vacation message, remove the **.forward** file, **.vacation.dir** file, **.vacation.pag** file, and **.vacation.msg** file from your **$HOME** (login) directory:

   ```
   rm .forward .vacation.dir .vacation.pag .vacation.msg
   ```

Files

Item	Description
$HOME/.forward	Contains the names of people who you want your mail to be forwarded to.
/usr/share/lib/vacation.def	Contains the systemwide default **vacation** message.
$HOME/.vacation.dir	Contains the names of people who have sent mail to you while the **vacation** command was being used.
$HOME/.vacation.msg	Contains your personalized **vacation** message.
$HOME/.vacation.pag	Contains the names of people who have sent mail to you while the **vacation** command was being used.
/usr/bin/vacation	Contains the **vacation** command.

val Command (SCCS)

Purpose

Validates SCCS files.

Syntax

val [**-s**] [**-r***SID*] [**-m***Name*] [**-y***Type*] *File* ...

Description

The **val** command reads the specified file to determine if it is a Source Code Control System (SCCS) file meeting the characteristics specified by the accompanying flags. If you specify a **-** (minus) for the *File* value, the **val** program reads standard input and interprets each line of standard input as **val** flags and the name of an SCCS file. An end-of-file character terminates input.

The **val** command displays messages to standard output for each file processed.

Flags

Each flag or group of flags applies independently to each named file. The flags can appear in any order.

Item	Description
-m*Name*	Compares the *Name* value with the SCCS **31** identification keyword in the specified file.
-r *SID*	Specifies the SID of the file to be validated. The SID must be valid and unambiguous.
-s	Suppresses the error message normally written to standard output.
-y*Type*	Specifies a type to compare with the SCCS identification keyword in the specified file.

Exit Status

The **val** command returns 0 if successful for all files; otherwise, it returns an 8-bit code that is a disjunction of the possible errors. It is interpreted as a bit string in which set bits (from left to right) are interpreted as follows:

Item	Description
0x80	Missing file argument.
0x40	Unknown or duplicate option.

Item	Description
0x20	Corrupted SCCS file.
0x10	Cannot open file or file not SCCS.
0x08	SID is invalid or ambiguous.
0x04	SID does not exist.
0x02	, y mismatch.
0x01	31, m mismatch.

Note: The **val** command can process two or more files on a given command line and can process multiple command lines (when reading standard input). In these cases, an aggregate code is returned; a logical OR of the codes generated for each command line and file processes.

Example

To determine if file `s.test.c` is an SCCS text file, enter:

```
val -ytext s.test.c
```

varyoffvg Command

Purpose

Deactivates a volume group.

Syntax

varyoffvg [**-s**] *VolumeGroup*

Description

The **varyoffvg** command deactivates the volume group specified by the *VolumeGroup* parameter along with its associated logical volumes. The logical volumes first must be closed. For example, if the logical volume contains a file system, it must be unmounted.

To activate the volume group, use the **varyonvg** command.

Note: To use this command, you must either have root user authority or be a member of the **system** group.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit varyoffvg
```

Note:

- A volume group that has a paging space volume on it cannot be varied off while the paging space is active. Before you deactivate a volume group with an active paging space volume, ensure that the paging space is not activated automatically at system initialization, and then reboot the system.
- The `varyoffvg` command discards any background space reclamation process that is running for the volume group. To identify whether a space reclamation is running, you can use the `lvmstat` command with **`-r`** option.

Flag

Item	Description
-s	Puts the volume group into System Management mode, so that only logical volume commands can be used on the volume group. In this mode, no logical volume can be opened or accessed by users.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To deactivate volume group vg03, enter:

   ```
   varyoffvg vg03
   ```

2. To deactivate volume group vg02, but allow logical volume commands to continue to take effect, enter:

   ```
   varyoffvg  -s vg02
   ```

   Logical volumes within the volume group cannot be opened, but logical volume commands continue to take effect.

File

Item	Description
/usr/sbin/varyoffvg	Contains the **varyoffvg** command.

varyonvg Command

Purpose

Activates a volume group.

Syntax

varyonvg [**-b**] [**-c**] [**-f**] [**-M** *ltgsize*] [**-n**] [**-p**] [**-r**] [**-s**] [**-t**] [**-u**] [**-k loc**|**rem**] [**-d**] [**-o**] [**-O**] *volumegroup*

Description

The **varyonvg** command activates the volume group specified by the *volumegroup* parameter and all associated logical volumes. A volume group that is activated is available for use. When a volume group

is activated, physical partitions are synchronized if they are not current. Physical volumes that are in the PVMISSING state and that have been replaced will be returned to the PVACTIVE state by the **varyonvg** command.

Note: If a physical volume is part of a dump device, the **varyonvg** command cannot return it to the PVACTIVE state. To run the command effectively, temporarily change the dump device.

A list of all physical volumes with their status is displayed to standard output whenever there is some discrepancy between the Device Configuration Database and the information stored in the Logical Volume Manager (LVM). The volume group may or may not be varied on. You must carefully examine the list and take proper action depending on each reported status to preserve your system integrity.

While varying on in concurrent mode, if the varyon process detects that there are logical volumes which are not previously known to the system, their definitions are imported. The permissions and ownership of the new device special files are duplicated to those of the volume group special file. If you have changed the permissions and/or ownership of the device special files of the logical volume on the node it was created, you will need to perform the same changes on this node.

Restriction: Classic Concurrent mode is not supported in AIX 5.3.

If the *volume group* cannot be varied on due to a loss of the majority of physical volumes, a list of all physical volumes with their status is displayed. To vary on the *volume group* in this situation, you will need to use the force option.

The **varyonvg** command fails to vary on the volume group if a majority of the physical volumes are not accessible (no Quorum). This condition is true even if the quorum checking is disabled. Disabling the quorum checking will only ensure that the volume group stays varied on even in the case of loss of quorum.

The *volume group* will not vary on if any physical volumes are in the `PV_MISSING` state and the quorum checking is disabled. This condition is true even if a quorum of disks are available. To vary on in this situation either use the force option or set an environment variable `MISSINGPV_VARYON` to `TRUE` (set this value in **/etc/environment** if the volume group needs to be varied with missing disks at the boot time).

In the above cases (using the force vary on option and using the `MISSINGPV_VARYON` variable), you take full responsibility for the *volume group* integrity.

When you vary on a volume group that contains an encrypted logical volume, the **varyonvg** command attempts to unlock the encrypted logical volume. If the logical volume is configured with automated key protection methods, such as Platform keystore (PKS) or key server encryption methods, the **varyonvg** command communicates with the **hdcryptmgr** command to unlock the encrypted logical volume. If the unlock operation is successful, the logical volume is ready for I/O operations. Otherwise, any I/O requests to the encrypted logical volume is blocked and an error code of `EACCES` is returned. If a volume group contains an encrypted logical volume that is not configured with any automated key protection methods, the logical volume is blocked for any I/O operations until the unlock operation is performed successfully.

Requirement: To use this command, you must either have root user authority or be a member of the **system** group.

You could also use the System Management Interface Tool (SMIT) **smit varyonvg** fast path to run this command.

Flags

Item	Description
-b	Breaks disk reservations on disks locked as a result of a normal **varyonvg** command. Use this flag on a volume group that is already varied on. **Notes:** • This flag unlocks all disks in a given volume group. • The **-b** flag opens the disks in the volume group using **SC_FORCED_OPEN** flag. For SCSI and FC disks this forces open all LUNS on the target address that this disk resides on. Volume groups should therefore not share target addresses when using the **varyon -b** option. • The **-b** flag can cause a system to hang if used on a volume group that contains an active paging space.
-c	Varies the volume group on Enhanced Concurrent mode. This is only possible if the volume group is Concurrent Capable or Enhanced Concurrent Capable and the system has the PowerHA SystemMirror product loaded and available. If neither is true, the volume group fails the varyon operation. **Requirement:** Enhanced Concurrent volume groups use Group Services. Group Services must be configured prior to activating a volume group in this mode.
-d	Allows data divergence. The **-d** flag only takes effect when you try to bring the volume group online while the cache at the opposite site might contain nonmirrored data updates and that cache is not accessible. If the **varyonvg** command detects that you might use back-level data and you do not specify the **-d** flag, the command fails with a meaningful error message. For more information about asynchronous mirroring of geographic LVM, see *Geographic Logical Volume Manager for PowerHA SystemMirror Enterprise Edition*.
-f	Allows a volume group to be made active that does not currently have a quorum of available disks. All disks that cannot be brought to an active state will be put in a removed state. At least one disk must be available for use in the volume group. The **-f** flag (used to override quorum loss) is ignored if the volume group has not lost quorum. If a disk is put into removed state, use the **chpv -v a** *PVname* command to bring the disk back to active state.
-k loc | **rem**	Keeps data from the local mirror copy or remote mirror copy. You can specify the following attributes with the **-k** flag: **loc** Retains the local mirror copy data. Local means local physical volumes and not primary site **rem** Retains the remote mirror copy data. Remote means remote physical volumes and not remote site. For more information about asynchronous mirroring of geographic LVM, see *Geographic Logical Volume Manager for PowerHA SystemMirror Enterprise Edition*.
-M *ltgsize*	Statically sets the *ltgsize* of the volume group. Valid values for *ltgsize* include 128K, 256K, 512K, 1M, 2M, 4M, 8M, 16M, 32M, and 128M. If any disk in the volume group is not configured with a maximum transfer of *ltgsize* or greater, the varyonvg command will fail.
-n	Disables the synchronization of the stale physical partitions within the *volumegroup* parameter.

Item	Description
-o	Allows using data from partitions that are stale in the copy you selected but fresh in the other copy. The **varyonvg** command fails if you specify the **-k** flag to preserve either local copy or remote copy in the data divergence case and the **varyonvg** command cannot preserve the complete copy because some partitions are not fresh in the local or remote copy that you selected. You can override the failure by specifying the **-o** flag to use data from partitions that are stale in the copy that you selected but fresh in the other copy. The **-o** flag is only valid with the **-k** flag. For more information about asynchronous mirroring of geographic LVM, see *Geographic Logical Volume Manager for PowerHA SystemMirror Enterprise Edition*.
-p	All physical volumes must be available to use the **varyonvg** command.
-r	Varies on the volume group in read-only mode. This mode prevents: • Write operations to logical volumes • LVM metadata updates • Stale partitions synchronization **Restriction:** Mounting a JFS file system on a read-only logical volume is not supported. **Restriction:** All LVM high-level commands that require the LVM metadata update will fail the request in this mode.
-s	Makes the volume group available in System Management mode only. Logical volume commands can operate on the volume group, but no logical volumes can be opened for input or output. **Restriction:** Logical volume commands also cannot read or write to or from logical volumes in a volume group varied on with the **-s** flag. Logical volumes that attempt to write to a logical volume in a volume group varied on with the **-s** flag (such as **chvg** or **mklvcopy**) may display error messages indicating that they were unable to write to and/or read from the logical volume.
-t	Checks the timestamps in the Device Configuration Database and the Logical Volume Manager. If there is a discrepancy in the timestamps, the **synclvodm** command is issued to synchronize the database. **Tip:** This check is always done if the Volume Group is varied on in concurrent mode.
-u	Varies on a volume group, but leaves the disks that make up the volume group in an unlocked state. Use this flag as part of the initial varyon operation of a dormant volume group.
-O	Force a varyon operation of the volume group even if it is varied on in some other node. **Note:** In AIX 61 TL8 and later releases, if the volume group created is not allowed to varyon in non-concurrent mode in more than one node then the **varyonvg** command updates LVM metadata and ODM with the varyon state of the volume group. During varyon time, the **varyonvg** command reads this data and fails if the volume group is already varied on in another node. The **varyoffvg** command will reset the varyon state of the volume group during the varyoff time. If the system crashes before varying off the volume group or the volume group is forced off, the **varyonvg** command fails after you reboot the system. In this scenario, use **–O** flag to force a varyon operation of the volume group.

Attention: The base design of LVM assumes that only one initiator can access a volume group. The PowerHA SystemMirror product does work with LVM in order to synchronize multi-node accesses of a shared volume group. However, multi-initiator nodes can easily access a volume group with the **-b** and **-u** flags without the use of PowerHA SystemMirror. You must be aware that volume group status information might be compromised or inexplicably altered as a result of disk protect (locking) being bypassed with these two flags. If you use the **-b** and **-u** flags, data and status output cannot be guaranteed to be consistent.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To activate volume group vg03, enter:

```
varyonvg vg03
```

2. To activate volume group vg03 without synchronizing partitions that are not current, enter:

```
varyonvg -n vg03
```

Files

Item	Description
/usr/sbin	Contains the **varyonvg** command directory.
/tmp	Stores the temporary files while the command is running.

vc Command

Purpose

Substitutes assigned values for identification keywords.

Syntax

vc [**-a**] [**-t**] [**-s**] [**-c***Character*] [*Keyword*=*Value*]...

Description

The **vc** command copies lines from standard input to standard output. The flags and keywords on the command line and control statements in the input modify the resulting output. The **vc** command replaces user-declared keywords with the value assigned on the command line. Keywords can be replaced both in text and in control statements.

Control Statements

A control statement is a single line beginning with a control character (the default control character is a **:** (colon)). Control statements provide conditional processing of the input. The allowable types of control statements are:

:if *Condition*

Text

Item	Description
:end	Writes all the lines between the **:if** statement and the matching **:end** to standard output only if the condition is true. You can nest **:if** and **:end** statements. However, once a condition is false, all remaining nested **:if** and **:end** statements are ignored. See the Condition Syntax section for the syntax of conditions and allowable operators.
:dcl *Keyword***,** [*Keyword* . . .]	Declares specified keywords. All keywords must be declared.
:asg *Keyword***=***Value*	Assigns the specified value to the specified keyword. An **:asg** statement takes precedence over keyword assignment on the **vc** command line. A later **:asg** statement overrides all earlier assignments of the associated keyword. The keywords that are declared but not assigned *Values,* have null values.
:: *Text*	Removes the two leading control characters, replaces keywords with their respective values, and then copies the line to standard output.
:on or **:off**	Turns on or off keyword replacement on all lines.
:ctl *Character*	Changes the control character to the *Character* value.
:msg *Message*	Writes a message to standard error output in the form: Message(n): message `where n is number of the input line on which the message appeared.`
:err *Message*	Writes an error message to standard error. The **vc** command stops processing and returns an exit value of 1. The error message is in the form: `ERROR: message` `ERROR: err statement on line n (vc15)`

Condition Syntax

The items and statements allowed are:

```
condition             ::=OR statement
                      ::=NOR statement
OR statement          ::=AND statement
                      ::=AND statement | OR statement
AND statement         ::=expression
                      ::=expression & AND statement
expression            ::=( OR statement )
                      ::=value operator value
operator value        ::= = or != or < or >
                      ::= ASCII string
                      ::= numeric string
```

The available condition operators and their meanings are:

Item	Description
=	Equal

Item	Description
!=	Not equal
&	AND
&|	OR
>	Greater than
<	Less than
( )	Used for logical groupings
NOT	May only occur immediately after the *if*, and when present, inverts the value of the entire condition.

The > and < (greater-than and less-than) operate only on unsigned integer values; for example, 012 > 12 is false. All other operators take strings as modifiers; for example, 012 ! = 12 is true. The precedence of the operators, from highest to lowest precedence, is as follows:

- = ! = > < (all of equal precedence)
- &
- &|

Parentheses can be used to alter the order of precedence.

Values must be separated from operators or parentheses by at least one blank or tab.

Keyword Replacement

A keyword must begin and end with the same control character used in control statements. A keyword may be up to nine alphanumeric characters, where the first character must be alphabetic. Keyword values can be any ASCII string. A numeric keyword *Value* is an unsigned string of digits. Values cannot contain tabs or spaces.

Flags

Item	Description
-a	Replaces keywords surrounded by control characters with their assigned value in all text lines (not just those beginning with two control characters).
-c*Character*	Uses the *Character* value as the control character. The *Character* parameter must specify an ASCII character.
-s	Does not display the warning messages normally displayed to standard error.
-t	Ignores all characters from the beginning of a line up to and including the first tab character for detecting a control statement. If the **vc** command finds a control character, it ignores all characters up to and including the tab.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. Examples of *Keyword*=*Value* assignments are:

   ```
   numlines=4
   prog=acctg
   pass4=yes
   ```

   The **vc** command removes all control characters and keywords from input text lines marked with two control characters as it writes the text to standard output.
2. To prevent a control character from being interpreted, precede it with a backslash, as in the following example:

   ```
   ::the :prog: program includes several of the following\:
   ```

   The **:prog:** keyword is replaced by its value, but the \ **:** is passed to standard output as **:** (colon).

   Input lines beginning with a \ (backslash) followed by a control character are not control lines, and are copied to standard output without the backslash. However, the **vc** command writes lines beginning with a backslash and no following control character without any changes (including the initial backslash).

File

Item	Description
/usr/bin/vc	Contains the **vc** command.

vgrind Command

Purpose

Formats listings of programs that are easy to read.

Syntax

vgrind [**-f**] [**-n**] [**-t**] [**-x**] [**-P***Printdev*] [**-T***Name*] [**-**] [**-d***File*] [**-h** *Header*] [**-l***Language*] [**-s***Size*] [*File* ...]

Description

The **vgrind** command formats (grinds) the program sources specified by the *File* parameters in an easily readable style using the **troff** command. Comments are placed in italics, keywords in boldface, and the name of the current function is listed down the margin of each page as it is encountered.

The **vgrind** command runs in either filter mode or regular mode.

In filter mode, the **vgrind** command acts as a filter in a manner similar to the **tbl** command. Standard input is passed directly to standard output except for lines bracketed by the following **troff**-like macros:

Item	Description
.vS	Starts processing.
.vE	Ends processing.

The preceding lines are formatted according to the **vgrind** command conventions. The output from this filter can be passed to the **troff** command for output. There is no particular ordering with the **eqn** or **tbl** command.

In regular mode, the **vgrind** command accepts input files, processes them, and passes them in order to the **troff** command, the appropriate postprocessor, and then the printer.

In both modes, the **vgrind** command passes without converting lines, beginning with a decimal point.

The **vgrind** command supports only ASCII keywords defined in either the standard **/usr/share/lib/vgrindefs** language definitions file or any alternately specified file by the **-d** flag.

Flags

Item	Description
-f	Forces filter mode.
-n	Forces no keyword bolding.
-t	Causes formatted text to go to standard output.
-x	Outputs the index file in an easily readable format. The index file itself is produced whenever the **vgrind** command is run with the **index** file in the current directory. The index of function definitions can then be run off by running the **vgrind** command with the **-x** flag and the *File* parameter.
-P*PrintDev*	Sends the output to *Printdev* Printer using the **qprt** command. If this flag is not specified, the **PRINTER** environment variable is used. If the **PRINTER** environment variable is not set, the system default is used.
-T*Name*	Creates output for a **troff** device as specified by the *Name* parameter. The output is sent through the appropriate postprocessor. The default is the **ibm3816** postprocessor.
-	Forces input to be taken from standard input (default if the **-f** flag is specified).
-d*File*	Specifies an alternate language definitions file (default is the **/usr/share/lib/vgrindefs** file).
-h *Header*	Specifies a particular header to put on every output page (default is the file name). **Note:** A blank space is required after the **-h** flag before the *Header* variable.

Item	Description
-l*Language*	Specifies the language to use. Currently known languages are: **c** C (the default). Function names can be preceded on a line only by spaces, tabs, or an asterisk. The parenthetical options must also be on the same line. **csh** CSH. **p** PASCAL. Function names must be displayed on the same line as the **function** or **procedure** keywords. **m** MODEL. Function names must be displayed on the same line as the **isbeginproc** keyword phrase. **sh** SHELL. **r** RATFOR. **mod2** MODULA2. **yacc** YACC. **isp** ISP. **I** ICON. **-s *Size*** Specifies a point size to use on output (exactly the same as a **.ps** request).

Files

Item	Description
index	Contains the file the where source for the index is created.
/usr/bin/vgrind	Contains the **vgrind** command.
/usr/share/lib/tmac/tmac.vgrind	Contains the macro package.
/usr/share/lib/vfontedpr	Contains the preprocessor.
/usr/share/lib/vgrindefs	Contains the language descriptions.

vi or vedit Command

Purpose

Edits files with a full-screen display.

Syntax

{ **vi** | **vedit** } [**-l**] [**-R**] [**-t***Tag*] [**-v**] [**-w***Number*] [**-y***Number*] [**-r** [*File*]] [{ **+** | **-c** } { *Subcommand* }] [*File* ...]

Description

The **vi** command starts a full-screen editor based on the underlying ex editor. Therefore, ex subcommands can be used within the vi editor. The **vedit** command starts a version of the vi editor intended for beginners. In the vedit editor, the **report** option is set to 1, the **showmode** option is set, and the **novice** option is set, making it a line editor.

You start the vi editor by specifying the name of the file or files to be edited. If you supply more than one *File* parameter on the command line, the vi editor edits each file in the specified order. The vi editor on an existing file displays the name of the file, the number of lines, and the number of characters at the bottom of the screen. In case of multibyte locales the number of characters need to be interpreted as the number of bytes.

Since the vi editor is a full-screen editor, you can edit text on a screen-by-screen basis. The vi editor makes a copy of the file you are editing in an edit buffer, and the contents of the file are not changed until you save the changes. The position of the cursor on the display screen indicates its position within the file, and the subcommands affect the file at the cursor position.

vi Editor Limitations

The following list provides the maximum limits of the vi editor. These counts assume single-byte characters.

- 256 characters per global command list
- 2048 characters in a shell escape command
- 128 characters in a string-valued option
- 30 characters in a tag name
- 128 map macros with 2048 characters total
- 1,048,560 lines silently enforced
- The macro name and the macro text are limited to 100 characters.

Note: The vi editor supports a maximum of 2 GB edit buffer.

vi Editing Modes

The vi editor operates in the following modes:

Item	Description
command mode	When you start the vi editor, it is in command mode. You can enter any subcommand except those designated for use only in the text input mode. The vi editor returns to command mode when subcommands and other modes end. Press the Esc key to cancel a subcommand.
text-input mode	You use the vi editor in this mode to add text. Enter text input mode with any of the following subcommands: the **a** subcommand, **A** subcommand, **i** subcommand, **I** subcommand, **o** subcommand, **O** subcommand, **c***x* subcommands (where the *x* represents the scope of the subcommand), **C** subcommand, **s** subcommand, **S** subcommand, and **R** subcommand. After entering one of these subcommands, you can enter text into the editing buffer. To return to command mode, press the Esc key for normal exit or press Interrupt (the Ctrl-C key sequence) to end abnormally.

Item	Description
last-line mode	Subcommands with the prefix **:** (colon), **/** (slash), **?** (question mark), **!** (exclamation point), or **!!** (two exclamation points) read input on a line displayed at the bottom of the screen. When you enter the initial character, the vi editor places the cursor at the bottom of the screen, where you enter the remaining characters of the command. Press the Enter key to run the subcommand, or press Interrupt (the Ctrl-C key sequence) to cancel it. When the **!!** prefix is used, the cursor moves only after both exclamation points are entered. When you use the **:** prefix to enter the last-line mode, the vi editor gives special meaning to the following characters when they are used before commands that specify counts: **%** All lines regardless of cursor position **$** Last line **.** Current line **Note:** The history of last line mode subcommands can be navigated using the Up and Down Arrow keys.

Customizing the vi Editor

You can customize the vi editor by:

- Setting vi editor options
- Defining macros
- Mapping keys
- Setting abbreviations

Setting vi Editor Options

The following list describes the vi editor options you can change with the **set** command. The default setting for these options is **off**. If you turn on one of these toggle options, you can turn it off again by entering the word **no** before the option. If you want to discontinue the **autowrite** vi option, enter **noaw**, where **no** turns off the option and **aw** specifies the **autowrite** option.

Note: Do not include parentheses when entering vi options.

vi Option (Abbreviation)	Description
autoindent (ai)	Indents automatically in text input mode to the indentation of the previous line by using the spacing between tab stops specified by the **shiftwidth** option. The default is **noai**. To back the cursor up to the previous tab stop, press the Ctrl-D key sequence. This option is not in effect for global commands.
autoprin (ap)	Prints the current line after any command that changes the editing buffer. The default is **ap**. This option applies only to the last command in a sequence of commands on a single line and is not in effect for global commands.
autowrite (aw)	Writes the editing buffer to the file automatically before the **:n** subcommand, the **:ta** subcommand, the Ctrl-A, Ctrl -], and Ctrl -T key sequences, and the **!** subcommand if the editing buffer changed since the last **write** subcommand. The default is **noaw**.

vi Option (Abbreviation)	Description
backtags (bt)	Allows the Ctrl-T subcommand to return the file editing position to the location where the previous Ctrl-] subcommand was issued. If **nobacktags** is set, then Ctrl-T is the same as Ctrl-]. The default is **backtags**.
beautifying text (bf)	Prevents the user from entering control characters in the editing buffer during text entry (except for tab, new-line, and form-feed indicators). The default is **nobf**. This option applies to command input.
closepunct (cp=)	Handles a list of closing punctuation, especially when wrapping text (**wraptype** option). Precedes multicharacter punctuation with the number of characters; for example, `cp=3.;)}`. The **vi** command does not split closing punctuation when wrapping.
directory (dir=)	Displays the directory that contains the editing buffer. The default is **dir = /var/tmp**.
edcompatible (ed)	Retains **g** (global) and **c** (confirm) subcommand suffixes during multiple substitutions and causes the **r** (read) suffix to work like the **r** subcommand. The default is **noed.**
exrc (exrc)	If not set, ignores any **.exrc** file in the current directory during initialization, unless the current directory is that named by the **HOME** environment variable. The default is **noexrc**.
hardtabs (ht=)	Tells the vi editor the distance between the hardware tab stops on your display screen. (This option must match the tab setting of the underlying terminal or terminal emulator.) The default is **ht=8**.
history (hist=)	Sets the limit on last line mode history commands. The initial value is `hist=32`. The history size is zero (`hist=0`) for the `tvi` command.
ignorecase (ic)	Ignores distinction between uppercase and lowercase while searching for regular expressions. The default is **noic**.
linelimit (ll=)	Sets the maximum number of lines, as per the **-y** command-line option. This option only is effective if used with the **.exrc** file or the **EXINIT** environment variable.
lisp (lisp)	Removes the special meaning of ( ), { }, [[, and]] and enables the = (formatted print) operator for s-expressions, so you can edit list processing (LISP) programs. The default is **nolisp.**
list (list)	Displays text with tabs (^I) and the marked end of lines ($). The default is **nolist.**
magic (magic)	Treats the . (period), [(left bracket), and * (asterisk) characters as special characters when searching for a pattern. In off mode, only the ( ) (parentheses) and $ (dollar sign) retain special meanings. However, you can evoke special meaning in other characters by preceding them with a \ (backslash). The default is **magic**.

vi Option (Abbreviation)	Description
mesg (mesg)	Turns on write permission to the terminal if set while in visual mode. This option only is effective if used with the **.exrc** file or the **EXINIT** environment variable. The default is **on**.
modeline (modeline)	Runs a vi editor command line if found in the first five or the last five lines of the file. A vi editor command line can be anywhere in a line. For the vi editor to recognize a command line, the line must contain a space or a tab followed by the `ex:` or `vi:` string. The command line is ended by a second **:** (colon). The vi editor tries to interpret any data between the first and second colon as vi editor commands. The default is **nomodeline**.
novice	Indicates whether you are in **novice** mode. You cannot change the value by using the **set** command.
number (nu)	Displays lines prefixed with their line numbers. The default is **nonu**.
optimize (opt)	Speeds the operation of terminals that lack cursor addressing. The default is **noopt**.
paragraphs (para=)	Defines vi macro names that start paragraphs. The default is **para=IPLPPPQPP\ LIpplpipnpbp**. Single-letter **nroff** macros, such as the **.P** macro, must include the space as a quoted character if respecifying a paragraph.
partialchar (pc=)	Appears in the last display column where a double-wide character would not be displayed completely. The default character is **-** (minus sign).
prompt	Prompts for a new vi editor command when in command mode by printing a **:** (colon). The default is **on**.
readonly (ro)	Sets permanent read-only mode. The default is **noreadonly**.
redraw (redraw)	Simulates a smart workstation on a dumb workstation. The default is **nore**.
remap	Allows defining macros in terms of other macros. The default is **on**.
report (re=)	Sets the number of times you can repeat a command before a message is displayed. For subcommands that produce many messages, such as global subcommands, the messages are displayed when the command sequence completes. The default is **report=5.**
scroll (scr=)	Sets the number of lines to be scrolled when the user scrolls up or down. The default is 1/2 of the window size, rounded down.
sections (sect=)	Defines vi macro names that start sections. The default is **sect=NHSHHH\ HUuhsh+c**. Single-letter **nroff** macros, such as the **.P** macro, must include the space as a quoted character if respecifying a paragraph.
shell (sh=)	Defines the shell for the **!** subcommand or the **:!** subcommand. The default is the login shell.

vi Option (Abbreviation)	Description
shiftwidth (sw=)	Sets the distance for the software tab stops used by the **autoindent** option, the shift commands **(>** and **<)**, and the text input commands (the Ctrl-D and Ctrl-T key sequences). This vi option only affects the indentation at the beginning of a line. The default is **sw=8.**
showmatch (sm)	Shows the ((matching left parenthesis) or { (left bracket) as you type the) (right parenthesis) or } (right bracket). The default is **nosm**.
showmode (smd)	Displays a message to indicate when the vi editor is in input mode. The default is **nosmd**.
slowopen (slow)	Postpones updating the display screen during inserts. The default is **noslow.**
tabstop (ts=)	Sets the distance between tab stops in a displayed file. The default is **ts=8**.
tags (tags =)	Defines the search path for the database file of function names created using the **ctags** command. The default is **tags=tags\ /usr/lib/tags**.
term (term=)	Sets the type of workstation you are using. The default is **term=$TERM**, where **$TERM** is the value of the **TERM** shell variable.
terse (terse)	Allows the vi editor to display the short form of messages. The default is **noterse**.
timeout (to)	Sets a time limit of two seconds on an entry of characters. This limit allows the characters in a macro to be entered and processed as separate characters when the **timeout** option is set. To resume use of the macro, set the **notimeout** option. The default is **to**.
ttytype	Indicates the tty type for the terminal being used. You cannot change this value from the vi editor.
warn (warn)	Displays a warning message before the **!** subcommand executes a shell command if it is the first time you issued a shell command after changes were made in the editing buffer but not written to a file. The default is **warn**.
window (wi=)	Sets the number of lines displayed in one window of text. The default depends on the baud rate at which you are operating: 600 baud or less, 8 lines; 1200 baud, 16 lines; higher speeds, full screen minus 1 line.
wrapmargin (wm=)	Sets the margin for automatic word wrapping from one line to the next. The default is **wm=0**. A value of 0 turns off word wrapping.
wrapscan (ws)	Allows string searches to wrap from the end of the editing buffer to the beginning. The default is **ws**.

vi Option (Abbreviation)	Description
wraptype (wt=)	Indicates the method used to wrap words at the end of a line. The default value is **general**. You can specify one of the following four values: **general** Allows wraps on word breaks as white space between two characters. This setting is the default. **word** Allows wraps on words. **rigid** Allows wraps on columns and before closing punctuation. **flexible** Allows wraps on columns, but one character of punctuation can extend past the margin.
writeany (wa)	Turns off the checks usually made before a **write** subcommand. The default is **nowa**.

To see a list of the vi editor settings that have changed from the default settings, enter `set` and press the spacebar. Press the Enter key to return to the command mode.

To see a complete list of the vi editor settings, enter `set all`. Press the Enter key to return to the command mode.

To turn on a vi editor option, enter `set Option`. This command automatically returns you to the command mode.

To turn on multiple vi editor options, enter `set Option Option Option`. This command turns on the three designated vi editor options and returns you to the command mode.

To turn off a vi editor option, enter `set noOption`. This command automatically returns you to the command mode.

To change the value of a vi editor option, enter `set Option=Value`. This command automatically returns you to the command mode.

You can use the **:set** subcommand of the vi editor to set options for this editing session only, or to set options for this editing session and all future editing sessions.

To set or change vi editor options *for this editing session only*, enter the **:set** subcommand from the command line.

To set vi options for *all editing sessions*, put the **:set** subcommand in the **EXINIT** environment variable in the **.profile** file (read by the shell on login) or put the **set** subcommand into a **.exrc** file. The vi editor first looks for the **EXINIT** environment variable and runs its commands. If the **EXINIT** environment variable does not exist, the vi editor then looks for the **$HOME/.exrc** file and runs its commands. Last, and regardless of any previous results, the vi editor looks for the local **.exrc** file and runs its commands.

Note: This process is true except with the **tvi** command (trusted vi). In this instance, the vi editor looks for and runs only the **/etc/.exrc** file.

For information about changing an option by setting the **EXINIT** environment variable, see the description of environment variables in the **environment** file.

The **.exrc** file can contain subcommands of the form **set** *Option*=*Value*; for example:

```
set cp=3 . . ;
```

To include a comment in the **.exrc** file, use a **"** (double quotation mark) as the first character in the line.

Defining Macros

If you use a subcommand or sequence of subcommands frequently, you can use the vi editor to define a macro that issues that subcommand or sequence.

To define a macro, enter the sequence of subcommands into a buffer named with a letter of the alphabet. The lowercase letters a through z overlay the contents of the buffer, and the uppercase letters A through Z append text to the previous contents of the buffer, allowing you to build a macro piece by piece.

For example, to define a buffer macro named c that searches for the word corner and makes the third line after the word corner the current line, enter the following command:

```
o /corner/+3
```

Then press the Esc key and enter the following command:

```
"c
```

where c is the name of the buffer macro.

To add text to the previous contents of the defined buffer, enter the `o viSubcommand`, press the Esc key, and enter `"CapitalLetter`, where the *CapitalLetter* variable specifies an uppercase letter A through Z. For example, to build a buffer macro named T that searches for the word corner and allows you to add more commands, enter the following command:

```
o corner
```

Then press the Esc key and enter the following command:

```
"T
```

where T is the name of the buffer macro. You can repeat this process at any time to add more vi subcommands to the same buffer.

For example, to add commands that move the cursor to the previous line and delete that line, enter the following command:

```
o -dd
```

where - (minus sign) means to move the cursor up one line, and dd means to delete the current line. Press the Esc key and enter the following command:

```
"Tdd
```

To start the macro, enter `@Letter`, where the *Letter* variable specifies the letter name of the buffer macro you want to use. To use the same macro again, enter `@@` (two at symbols). For example, enter `@T` to start the T buffer macro and run the **search**, **move cursor**, and **delete line** commands. Enter `@@T` to start the T buffer macro again.

The character set used by your system is defined by the collation table. This table affects the performance of vi macros.

Mapping Keys

You can use the **:map**, **:map!**, and **:ab** subcommands to map a keystroke to a command or a sequence of commands. The **:map** subcommand is used in the command mode. The **:map!** and **:ab** subcommands are used in the text input mode. You can map keys for this editing session and all future editing sessions or only for the current editing session from either mode.

To map keys *for all future editing sessions*, put the subcommand into a **$HOME/.exrc** file. Each time you start the vi editor, it reads this file. The mapping remains in effect for every editing session.

To map keys *for the current editing session only* from the *command mode*, start the subcommand during the vi editor session. To map keys for the current editing session only from the *text input mode*, enter the subcommand on the command line during the vi editor session. The mapping remains in effect only for the current editing session.

Attention: If you use an IBM 3161 ASCII display station, IBM 3163 ASCII display station, or IBM 3101 ASCII display station, the default key-mapping of the vi editor can cause you to lose data. To see the default mapping, issue a **:map** subcommand. Specific problems arise with the Esc-J or Shift-J key sequence. These key sequences delete all information from the current position of the cursor to the end of the file. To avoid problems, change this key sequence using a **.exrc** file.

The **:map**, **:map!**, and **:ab** subcommands are defined and used as follows:

Item	Description
:map	Defines macros in the command mode. The **:map** subcommand allows you to run a specified command or sequence of commands by pressing a single key while in the vi editor. To map keys in the command mode, start the vi editor with an empty editing buffer and do not name a vi file using the **vi** command or type anything into the buffer after the vi editor starts. You can use the **:map** subcommand to do the following: • To map a character to a sequence of editing commands, enter: `:map Letter viSubcommand` • To unmap a character previously mapped in command mode, enter: `:unmap Letter` • To display a list of current mappings for the command mode, enter `:map` The following keys are not used by the vi editor, but are available for use with the **:map** subcommand in the command mode: • Letters g, K, q, V, and v • Control key sequences Ctrl-A, Ctrl-K, Ctrl-O, Ctrl-W, and Ctrl-X • Symbols _ (underscore), * (asterisk), \ (backslash), and = (equal sign) Although you can map a key that is already used by the vi editor, the key's usual function is not available as long as the map is in effect. Some terminals allow you to map command sequences to function keys. If you are in LISP mode, the = (equal sign) cannot be used because it is used by the vi editor. To map the letter v to the sequence of commands that would locate the next occurrence of the word `map` and change it to the word `MAP`, enter the following command: `:map v /map<Ctrl-V><Enter>cwMAP<Ctrl-V><Esc><Ctrl-V><Enter>` The previous example instructs the vi editor to locate the next occurrence of `map` (`/map<Ctrl-V><Enter>`), change `map` to `MAP` (`cwMAP`), end the change-word subcommand (`<Ctrl-V><Esc>`), and enter the command (`<Ctrl-V><Enter>`). **Requirement:** To prevent the vi editor from interpreting the Enter key, it must be preceded by the Ctrl-V key sequence when being mapped. This condition is also true of the Esc, Backspace, and Delete keys. To map the control characters Ctrl-A, Ctrl-K, and Ctrl-O, simultaneously press the Ctrl key and the letter. For example, to map the Ctrl-A key sequence to the sequence of commands that saves a file and edits the next one in a series, enter the following command: `:map <Ctrl-A> :w<Ctrl-V><Enter>:n<Ctrl-V><Enter>` To map the control characters Ctrl-T, Ctrl-W, and Ctrl-X, you must first escape them with the Ctrl-V key sequence.

Item **Description**

To map the | (pipe symbol), you must first escape it with the two Ctrl-V key sequences, as illustrated by the following example that maps the character g to the sequence of commands that escapes to the shell, concatenates the file **/etc/motd**, and pipes the output to the **wc** command:

```
:map g :!cat /etc/motd <Ctrl-V><Ctrl-V>| wc<Ctrl-V><Enter>
```

If your terminal permits you to map function keys, you must reference them with the #*number* key sequence to designate the number of the function key that you want to map. In the following example, the F1 function key is mapped to the sequence of commands that deletes a word and moves the cursor three words down:

```
:map #1 dwwww
```

In order for function key mapping to work, the output of the function key for your terminal type must match the output defined in the **terminfo** file. These definitions are denoted by the kf*number* entries, where kf1 represents the F1 function key, kf2 represents the F2 function key, and so on. If the output that you get when you press the function key does not match this entry, you must use the terminal's setup mode to correct the settings to match these terminal database entries before any mapping can occur.

You can also map certain keyboard special keys, such as the Home, End, Page Up, and Page Down keys. For most terminals, these keys are already mapped in the vi editor. You can verify this mapping by using the **:map** subcommand. If these keys are not already mapped, you can use the **:map** subcommand as follows:

```
:map <Ctrl-V><End> G
:map <Ctrl-V><Home> 1G
:map <Ctrl-V><PageUp> <Ctrl-F>
:map <Ctrl-V><PageDown> <Ctrl-B>
```

To get a listing of all current maps in the command mode, enter the **:map** subcommand. The preceding examples are then displayed as follows:

```
v       v          /map<Ctrl-M>cwMAP<Ctrl-[>Ctrl-M>
<Ctrl-A>  <Ctrl-A>   :w<Ctrl-M>:n<Ctrl-M>
g         g          :!cat /etc/motd | wc <Ctrl-M>
```

Tip: The Ctrl-V and Enter key sequence is displayed as the Ctrl-M key sequence, and the Ctrl-V and Esc key sequence is displayed as the Ctrl-[key sequence.

Item **Description**

:map! Maps character strings to single keys while in text input mode. To map keys in the text input mode, start the vi editor with an empty editing buffer and do not name a vi file using the **vi** command or type anything into the buffer after the vi editor starts. You can use the **:map!** subcommand to do the following:

- To map a letter to one or more vi strings in text input mode, enter:

```
:map! Letter String
```

- To unmap a letter previously mapped in text input mode, enter:

```
:unmap! Letter
```

- To display a list of existing strings that are mapped to specific keys in text input mode, enter:

```
:map!
```

Typing the mapped key while in text input mode produces the specified string. The Ctrl-V and Esc key sequence puts you into command mode, backs up to the beginning of the current word (**bbw**), and starts the **cw** (change-word) subcommand. For example:

```
:map! % <Ctrl-V><Esc>bbwcw
```

When typing text, if you realize that you have mistyped a word, you can change it by pressing the % (percent) key and retyping the word. You are automatically returned to insert mode.

Important: Be careful when choosing keys to be used for the **:map!** subcommand. Once keys have been mapped, they can no longer be input as text without first issuing the **:unmap!** subcommand.

:ab Maps a key or sequence of keys to a string of characters for use in the text input mode. The **:ab** subcommand is useful when inputting text that possesses several repetitive phrases, names, or titles.

The following example replaces the word `city` with the phrase `Austin, Texas 78759` whenever it is typed in text input mode and followed by a white space, period, or comma:

```
:ab city Austin, Texas 78759
```

For example, if while inputting text, you type the following:

```
My current residence is city.
```

Pressing the Tab key expands the word `city` to read:

```
My current residence is Austin, Texas 78759.
```

The abbreviation is not expanded within a word. For example, if you type `My current residence iscity`, the word iscity is not expanded.

If the **:map!** subcommand is used to map abbreviations for insert mode, then all occurrences of the abbreviations are expanded regardless of where it occurs. If you used the **:map!** subcommand for the preceding example (`:map! city Austin, Texas 78759`), then whenever you type the word `city`, regardless of what precedes or follows, the word will be expanded to `Austin, Texas 78759`. Therefore, the word `iscity` becomes `isAustin, Texas 78759`.

Important: Be careful when choosing the keys that are used for the **:ab** subcommand. Once keys are defined, they can no longer be input as text without first issuing the **:unab** subcommand.

Setting Abbreviations

The **set** command has behavior similar to the **map!** command except that the **set** command substitutes the string for the abbreviation only when the abbreviation is a separate word. You can use the **set** command of the vi editor to:

- List existing abbreviations
- Remove an abbreviation
- Set (define) an abbreviation

Tip: Start the vi editor with an empty editing buffer. Do not name a vi file using the **vi** command or type anything into the buffer after the vi editor starts. Press the Esc key to be sure you are in the command mode.

Item	Description
To list abbreviations	Enter the **:ab** command to list existing abbreviations. Press the Enter key to return to command mode.
To remove abbreviations	Enter the **:anab***Abbreviation* command to remove an abbreviation, where the *Abbreviation* variable specifies the character string you do not want abbreviated any more.
To set (define) an abbreviation	Enter the **:ab** *Abbreviation String* command to set an abbreviation, where the *Abbreviation* variable specifies the character string being defined as an abbreviation and the *String* variable specifies the character string being abbreviated. The abbreviation can be substituted for the string only when the abbreviation is a separate word. For example, if you enter the **:ab kn upper** command and then type `acknowledge` while in the text input mode, the set abbreviation string is not started because the kn string in the word acknowledge is not a separate word. However, if you type the **:ab kn upper** command and then type `make the kn line all kncase` while in the text input mode, the result is `make the upper line all uppercase`.

Flags

Item	Description
-c*Subcommand*	Carries out the ex editor subcommand before viewing with **vi** begins. The cursor moves to the line affected by the last subcommand to be carried out. When a null operand is entered, as in **-c' '**, the vi editor places the cursor on the first line of the file. The **-c** flag is incompatible with the **+** flag. Do not specify both flags at the same time.
-l	Enters the vi editor in LISP mode. In this mode, the vi editor creates indents appropriate for LISP code, and the **(**, **)**, **{**, **}**, **[[**, and **]]** subcommands are modified to act appropriately for LISP.
-r[*File*]	Recovers a file after a vi editor or system malfunction. If you do not specify the *File* variable, the vi editor displays a list of all saved files.

Item	Description
-R	Sets the **readonly** option to protect the file against overwriting.
-t*Tag*	Edits the file containing the *Tag* variable and positions the vi editor at its definition. To use this flag, you must first create a database of function names and their locations using the **ctags** command.
-v	Enters the vi editor in the verbose mode.
-w*Number*	Sets the default window size to the value specified by the *Number* variable. This flag is useful when you use the vi editor over a low-speed line.
-y*Number*	Overrides the maximum line setting of 1,048,560 with any value greater than 1024. You should request twice the number of lines that you require because the vi editor uses the extra lines for buffer manipulation.
+[*Subcommand*]	Carries out the ex editor subcommand before editing begins. If you do not specify the *Subcommand* variable, the cursor is placed on the first line of the file. This **+** flag is incompatible with the **-c** flag. Do not specify both flags at the same time.

vi General Subcommand Syntax

Use the following general syntax to enter subcommands:

[*Named_Buffer*] [*Operator*] [*Number*] *Object*

Tip: Square brackets indicate optional items.

Item	Description
[*Named_Buffer*]	Specifies a temporary text storage area.
[*Operator*]	Specifies the subcommand or action; instructs the vi editor.
[*Numbe*r]	Specifies either the extent of the action or a line address as a whole number.
Object	Specifies what to act on, such as a text object (a character, word, sentence, paragraph, section, character string) or a text position (a line, position in the current line, screen position).

Counts before Subcommands

You can put a number in front of many subcommands. The vi editor interprets this number in one of the following ways:

- Go to the line specified by the *Number* parameter:

```
5G
10Z
```

- Go to the column specified by the *Number* parameter:

```
25|
```

- Scroll the number of lines up or down specified by the *Number* parameter:

```
10Ctrl-U
10Ctrl-D
```

vi Editor Subcommands

Use the subcommands to perform these kinds of actions:

- Moving the cursor
- Editing text

- Manipulating files
- Other actions

Moving the Cursor

Use subcommands to move the cursor within a file in these ways:

- Moving within a line
- Moving within a line by character position
- Moving to words
- Moving by line position
- Moving to sentences, paragraphs, or sections
- Moving by redrawing the screen
- Paging and scrolling
- Searching for patterns
- Marking a specific location in a file and returning

Moving within a Line

Enter the following subcommands in command mode. You can cancel an incomplete command by pressing the Esc key. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
Left Arrow or **h** or **Ctrl-H**	Moves the cursor one character to the left.
Down Arrow or **j** or **Ctrl-J** or **Ctrl-N**	Moves the cursor down one line (it remains in the same column).
Up Arrow or **k** or **Ctrl-P**	Moves the cursor up one line (it remains in the same column).
Right Arrow or **l**	Moves the cursor one character to the right.

Moving within a Line by Character Position

Enter the following subcommands in command mode. You can cancel an incomplete command by pressing the Esc key. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
^	Moves the cursor to the first nonblank character.
0	Moves the cursor to the beginning of the line.
$	Moves the cursor to the end of the line.
f*x*	Moves the cursor to the next *x* character.
F*x*	Moves the cursor to the last *x* character.
t*x*	Moves the cursor to one column before the next *x* character.
T*x*	Moves the cursor to one column after the last *x* character.
;	Repeats the last **f**, **F**, **t**, or **T** subcommand.
,	Repeats the last **f**, **F**, **t**, or **T** subcommand in the opposite direction.
*Number***|**	Moves the cursor to the specified column.

Moving to Words

Enter the following subcommands in command mode. For more information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
w	Moves the cursor to the next small word.
b	Moves the cursor to the previous small word.
e	Moves the cursor to the next end of a small word.
W	Moves the cursor to the next big word.
B	Moves the cursor to the previous big word.
E	Moves the cursor to the next end of a big word.

Moving by Line Position

Enter the following subcommands in command mode. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
H	Moves the cursor to the top line on the screen.
L	Moves the cursor to the last line on the screen.
M	Moves the cursor to the middle line on the screen.
+	Moves the cursor to the next line at its first nonblank character.
-	Moves the cursor to the previous line at its first nonblank character.
Enter	Moves the cursor to the next line at its first nonblank character.

Moving to Sentences, Paragraphs, or Sections

Enter the following subcommands in command mode. You can cancel an incomplete subcommand by pressing the Esc key. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
(	Places the cursor at the beginning of the previous sentence, or the previous s-expression if you are in LISP mode.
)	Places the cursor at the beginning of the next sentence, or the next s-expression if you are in LISP mode.
{	Places the cursor at the beginning of the previous paragraph, or at the next list if you are in LISP mode.
}	Places the cursor at the beginning of the next paragraph, at the next section if you are in C mode, or at the next list if you are in LISP mode.
]]	Places the cursor at the next section, or function if you are in LISP mode.
[[	Places the cursor at the previous section, or function if you are in LISP mode.

Moving by Redrawing the Screen

Enter the following subcommands in command mode. You can cancel an incomplete subcommand by pressing the Esc key. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
z	Redraws the screen with the current line at the top of the screen.
z-	Redraws the screen with the current line at the bottom of the screen.
z.	Redraws the screen with the current line at the center of the screen.
/*Pattern***/z-**	Redraws the screen with the line containing the character string, specified by the *Pattern* parameter, at the bottom.

Paging and Scrolling

Enter the following subcommands in command mode. You can cancel an incomplete subcommand by pressing the Esc key. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
Ctrl-U	Scrolls up one-half screen.
Ctrl-D	Scrolls down one-half screen.
Ctrl-F	Scrolls forward one screen.
Ctrl-B	Scrolls backward one screen.
Ctrl-E	Scrolls the window down one line.
Ctrl-Y	Scrolls the window up one line.
z+	Pages up.
z^	Pages down.

Searching for Patterns

Enter the following subcommands in command mode. You can cancel an incomplete subcommand by pressing the Esc key. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
[*Number***]G**	Places the cursor at the line number specified by the *Number* parameter or at the last line if the *Number* parameter is not specified.
/*Pattern*	Places the cursor at the next line containing the character string specified by the *Pattern* parameter.
?*Pattern*	Places the cursor at the next previous line containing the character string specified by the *Pattern* parameter.
n	Repeats the last search for the text specified by the *Pattern* parameter in the same direction.
N	Repeats the last search for the text specified by the *Pattern* parameter in the opposite direction.
/*Pattern***/+***Number*	Places the cursor the specified number of lines after the line matching the character string specified by the *Pattern* parameter.
?*Pattern***?-***Number*	Places the cursor the specified number of lines before the line matching the character string specified by the *Pattern* parameter.
%	Finds the parenthesis or brace that matches the one at current cursor position.

Editing Text

The subcommands for editing enable you to perform the following tasks:

- Marking a specific location in a file and returning
- Adding text to a file
- Changing text while in input mode
- Changing text from command mode
- Copying and moving text
- Restoring and repeating changes

Marking a Specific Location in a File and Returning

Enter the following subcommands in command mode. You can cancel an incomplete subcommand by pressing the Esc key. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
"	Moves the cursor to the previous location of the current line.
"	Moves the cursor to the beginning of the line containing the previous location of the current line.
m*x*	Marks the current position with the letter specified by the *x* parameter.
`*x*	Moves the cursor to the mark specified by the *x* parameter.
'*x*	Moves the cursor to the beginning of the line containing the mark specified by the *x* parameter.

Adding Text to a File (Text Input Mode)

Enter the following subcommands in command mode to change the vi editor into text input mode. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
a*Text*	Inserts text specified by the *Text* parameter after the cursor. End text input mode by pressing the Esc key.
A*Text*	Adds text specified by the *Text* parameter to the end of the line. End text input mode by pressing the Esc key.
i*Text*	Inserts text specified by the *Text* parameter before the cursor. End text input mode by pressing the Esc key.
I*Text*	Inserts text specified by the *Text* parameter before the first nonblank character in the line. End text input mode by pressing the Esc key.
o	Adds an empty line below the current line. End text input mode by pressing the Esc key.
O	Adds an empty line above the current line. End text input mode by pressing the Esc key.

Changing Text While in Input Mode

Use the following subcommands only while in text input mode. These commands have different meanings in command mode. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
Ctrl-D	Goes back to previous autoindent stop.
^ Ctrl-D	Ends autoindent for this line only.

Item	Description
0Ctrl-D	Moves cursor back to left margin.
Esc	Ends insertion and returns to command state.
Ctrl-H	Erases the last character.
Ctrl-Q	Enters any character if xon is disabled.
Ctrl-V	Enters any character.
Ctrl-W	Erases the last small word.
****	Quotes the erase and kill characters.
Ctrl-?	Interrupts and ends insert or the Ctrl-D key sequence.

Changing Text from Command Mode

Use the following subcommands in command mode. An incomplete subcommand can be canceled by pressing the Esc key. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
C	Changes the rest of the line (same as **c$**).
cc	Changes a line.
cw	Changes a word.
cw*Text*	Changes a word to the text specified by the *Text* parameter.
D	Deletes the rest of the line (same as **d$**).
dd	Deletes a line.
dw	Deletes a word.
J	Joins lines.
r*x*	Replaces the current character with the character specified by *x*.
R*Text*	Overwrites characters with the text specified by the *Text* parameter.
s	Substitutes characters (same as **cl**).
S	Substitutes lines (same as **cc**).
u	Undoes the previous change.
x	Deletes a character at the cursor.
X	Deletes a character before the cursor (same as **dh**).
<<	Shifts one line to the left.
<L	Shifts all lines from the cursor to the end of the screen to the left.
>>	Shifts one line to the right.
>L	Shifts all lines from the cursor to the end of the screen to the right.
~	Changes letter at the cursor to the opposite case.
!	Indents for LISP.

Copying and Moving Text

Use the following subcommands in command mode. An incomplete subcommand can be canceled by pressing the Esc key. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
p	Puts back text from the undo buffer after the cursor.
P	Puts back text from the undo buffer before the cursor.
"*x***p**	Puts back text from the *x* buffer.
"*x***d**	Deletes text into the *x* buffer.
y	Places the object that follows (for example, **w** for word) into the undo buffer.
"*x***y**	Places the object that follows into the *x* buffer, where *x* is any letter.
Y	Places the line in the undo buffer.

Restoring and Repeating Changes

Use the following subcommands in command mode. An incomplete subcommand can be canceled by pressing the Esc key. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
u	Undoes the last change. **Tip:** After an undo, the cursor moves to the first non-blank character on the updated current line.
U	Restores the current line if the cursor has not left the line since the last change.
.	Repeats the last change or increments the **"***n***p** command. **Note:** 1. This subcommand will repeat the last change, including an undo. Therefore, after an undo, repeat performs an undo rather than repeat the last change. 2. This subcommand is not meant for use with a macro. Enter **@@** (two at signs) to repeat a macro.
"*n***p**	Retrieves the *n*th last delete of a complete line or block of lines.

Manipulating Files

The subcommands for manipulating files allow you to do the tasks outlined in the following sections:

- Saving changes to a file
- Editing a second file
- Editing a list of files
- Finding file information

Saving Changes to a File

Use the following subcommands in command mode. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
:w	Writes the edit buffer contents to the original file. If you are using this subcommand within the ex editor, you do not need to type the : (colon).

Item	Description
:w *File*	Writes the edit buffer contents to the file specified by the *File* parameter. If you are using this subcommand within the ex editor, you do not need to type the : (colon).
:w! *File*	Overwrites the file specified by the *File* parameter with the edit buffer contents. If you are using this subcommand within the ex editor, you do not need to type the : (colon).

Editing a Second File

Enter the following subcommands in command mode. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
:e *File*	Edits the specified file. If you are using this subcommand from the ex editor, you do not need to type the : (colon).
:e!	Re-edits the current file and discards all changes.
:e + *File*	Edits the specified file starting at the end.
:e + *Number File*	Edits the specified file starting at the specified line number.
:e #	Edits the alternate file. The alternate file is usually the previous file name before accessing another file with a **:e** command. However, if changes are pending on the current file when a new file is called, the new file becomes the alternate file. This subcommand is the same as the **Ctrl-A** subcommand.
:r *File*	Reads the file into the editing buffer by adding new lines below the current line. If you are using this subcommand from the ex editor, you do not need to type the **:** (colon).
:r !*Command*	Runs the specified command and places its output into the file by adding new lines below the current cursor position.
:ta *Tag*	Edits a file containing the *Tag* tag starting at the location of the tag. To use this subcommand, you must first create a database of function names and their locations using the **ctags** command. If you are using this subcommand from the ex editor, you do not need to type the **:** (colon).
Ctrl-]	Edits a file containing the tag associated with the current word starting at the location of the tag. To use this subcommand, you must first create a database of function names and their locations using the **ctags** command. Ctrl-T edits a file at the editing position where the previous Ctrl-] subcommand was issued. If multiple Ctrl-] subcommands have been issued, then multiple Ctrl-T subcommands can be used to return to previous editing positions where Ctrl-] subcommands were issued.
Ctrl-A	Edits the alternate file. The alternate file is usually the previous current file name. However, if changes are pending on the current file when a new file is called, the new file becomes the alternate file. This subcommand is the same as the **:e #** subcommand.

Editing a List of Files

Enter the following subcommands in command mode. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
:n	Edits the next file in the list entered on the command line. If you are using this subcommand from the ex editor, a : (colon) is not needed.

Item	Description
:n *Files*	Specifies a new list of files to edit. If you are using this subcommand from the ex editor, a **:** (colon) is not needed.

Finding File Information

Enter the following subcommand in command mode. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
Ctrl-G	Shows the current file name, current line number, number of lines in the file, and percentage of the way through the file where the cursor is located.

Other Actions

The vi editor provides the subcommands described in the following sections:

- Adjusting the screen
- Entering shell commands
- Interrupting and ending the vi editor

Adjusting the Screen

Enter the following subcommands in command mode. An incomplete subcommand can be canceled by pressing the Esc key. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
Ctrl-L	Clears and redraws the screen.
Ctrl-R	Redraws the screen and eliminates blank lines marked with @ (at sign).
z*Number*	Makes the window the specified number of lines long.

Entering Shell Commands

The following subcommands allow you to run a command within the vi editor. Enter these subcommands in command mode. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
:sh	Enters the shell to allow you to run more than one command. You can return to the vi editor by pressing the Ctrl-D key sequence. If you are using this subcommand within the ex editor, a : (colon) is not needed.
:!*Command*	Runs the specified command and then returns to the vi editor. If you are using this subcommand within the ex editor, a : (colon) is not needed. **Tip:** The # (alternate file), **%** (current file), and **!** (previous command) special characters are expanded when following a **:!** subcommand. To prevent any of these characters from being expanded, use the **** (backslash).
:!!	Repeats the last **:!***Command* subcommand**.**

Item	Description
*Number***!!***Command*	Runs the specified command and replaces the lines specified by *Number* with the output of the command**.** If a number is not specified, the default value is 1. If the command expects standard input, the specified lines are used as input.
!*Object Command*	Runs the specified command and replaces the object specified by the *Object* parameter with the output of the command**.** If the command expects standard input, the specified object is used as input.

Interrupting and Ending the vi Editor

Enter the following subcommands in command mode. If you need information about the format of vi subcommands, see vi General Subcommand Syntax.

Item	Description
Q	Enters the ex editor in command mode.
ZZ	Exits the vi editor, saving changes.
:q	Quits the vi editor. If you have changed the contents of the editing buffer, the vi editor displays a warning message and does not quit. If you are using this subcommand from the ex editor, a : (colon) is not needed.
:q!	Quits the vi editor, discarding the editing buffer. If you are using this subcommand from the ex editor, a : (colon) is not needed.
Esc	Ends text input or ends an incomplete subcommand.
Ctrl-?	Interrupts a subcommand.

Exit Status

The following exit values are returned:

Item	Description
0	Indicates successful completion.
>0	Indicates an error occurred.

Input Files

Input files must be text files or files that are similar to text files except for an incomplete last line that contains no null characters.

The **.exrc** files must be text files consisting of **ex** commands.

The `$HOME/.vi_history` file is an auto-generated text file that records the last line mode command history.

By default, the vi editor reads lines from the files to be edited without interpreting any of those lines as any form of vi editor command.

view Command

Purpose

Starts the vi editor in read-only mode.

Syntax

view [**-c***Subcommand*] [**-l**] [**-t** *Tag*] [**-w***Number*] [**-y**] [**-r** [*File*]] [**+** [*Subcommand*]] [*File ...*]

Description

The **view** command starts the vi full-screen editor in read-only mode. The read-only mode is only advisory to prevent accidental changes to the file. To override read-only mode, use the ! (exclamation point) when executing a command. The *File* parameter specifies the name of the file you want to browse. Use vi subcommands for moving within the file. Use the **:q** subcommand to exit the **view** command. If you modify the file you can save your modifications by pressing the Esc key and `wq!`.

Flags

Item	Description
-c*Subcommand*	Carries out the ex editor subcommand before viewing with vi begins. When a null operand is entered, as in `-c ''`, the editor places the cursor on the last line of the file.
-l	Enters a version of the vi editor with specialized features designed for writing programs in the LISP language. In this mode, the vi editor indents appropriately for LISP programming, and the **(**, **)**, **{**, **}**, **[[**, and **]]** subcommands are modified to act appropriately for LISP.
-r [*File*]	Recovers a file after an editor or system crash. If you do not specify a *File* parameter, the editor displays a list of all saved files.
-t*Tag*	Edits the file containing the tag specified by the *Tag* parameter and positions the editor at its definition. To use this flag, you must first create a database of function names and their locations using the ctags command.
-w*Number*	Sets the default window size to the value specified by the *Number* parameter. This is useful when your terminal communicates with the system running the editor over a slow communications line.
-y	Overrides the maximum line setting of 1,048,560 with any value greater than 1024.
+[*Subcommand*]	Carries out the ex editor subcommand specified by the *Subcommand* parameter before viewing with vi begins. If you do not specify a subcommand, the cursor is placed on the last line of the file.

viosupgrade command

Purpose

Performs the operations of backing up the virtual and logical configuration data, installing the specified image, and restoring the virtual and logical configuration data of the Virtual I/O Server (VIOS).

Syntax

To perform the **bosinst** type of upgrade operation, use the following syntax:

```
viosupgrade -t bosinst -n hostname -m ios_mksysbname
-p spotname  {-a RootVGCloneddisk: ... | -r RootVGInstallDisk: ...| -s}
[-b BackupFileResource][-c][-e resources: ...] [-F skipclusterstate] [-v]
```

To perform the **altdisk** type of upgrade operation, use the following syntax:

```
viosupgrade -t altdisk -n hostname -m ios_mksysbname
-a RootVGInstallDisk: ... [-b BackupFileResource] [-c] [-e
resources: ...] [-F skipclusterstate] [-v]
```

To perform a **bosinst** or **altdisk** type of upgrade operation across multiple nodes, use the following syntax:

```
viosupgrade -t {bosinst | altdisk} -f filename [-v]
```

To check the status of the triggered upgrade operation, use the following syntax:

```
viosupgrade -q { [-n hostname | -f filename] }
```

To create the `ios_mksysb` image file from the International Organization for Standardization (ISO) image files, use the following syntax:

```
viosupgrade -I ISOImage1:ISOImage2 -w directoryPath
  -x iosmksysbResourceName [-y spotResourceName]
```

Description

When the **viosupgrade** command is run, the following operations are performed in the background:

Backup
The virtual and logical configuration data is backed up to ensure that the VIOS partition can be recovered after a new installation.

Installation
Performs a new and complete installation of the VIOS partition from the provided VIOS image.

Restore
The virtual and logical configuration data of the VIOS partition is restored.

Flags

Flag name	Description
-a	Specifies one or more alternative disks for the installation. One of the following actions is taken based on the type of installation: • *bosinst* installation type: The provided disks are used to clone the current `rootvg`. After the completion of the migration process, the current `rootvg` disk is installed with the provided image. The provided disks are at the VIOS level before the migration process. • *altdisk* installation type: The provided disks are used to install the provided image. The current `rootvg` disk on the VIOS partition is not impacted during the installation process. The VIOS partition remains in the running state during the installation of the alternative disk.
-b	Specifies the resource name of the VIOS configuration backup file.
-c	Specifies that cluster-level backup and restore operations are performed. **Note:** The *-c* flag is mandatory for the VIOS that is part of an SSP cluster.

Flag name	Description
-e	Specifies the configuration resources to be applied after the installation. The valid values are **resolv_conf**, **script**, **fb_script**, **file_res**, **image_data**, and **log**. **Note:** • The **file_res** option is applicable only to *bosinst* type of installations. This option is not supported for *altdisk* type installations. • If you do not specify the **-e** flag or if the `/etc/resolv.conf` and `/etc/hosts` files are not included in the specified configuration resources, the upgrade process backs up these files from the current system before starting the installation operation. The backed-up files are restored during the migration operation.
-f	Specifies the file name that contains the list of VIOS nodes. The values and fields in the file must be specified in a particular sequence and format. The details of the format are specified in the `/usr/samples/nim/viosupgrade.inst` file and they are comma-separated. The maximum number of nodes that can be installed through the *-f* option is `30`. The VIOS images are installed on the nodes simultaneously. The installation status is displayed for each node. For an SSP cluster, the **viosupgrade** command must be run on individual nodes. Out of the n number of nodes in the SSP `cluster`, maximum `n-1` nodes can be upgraded at the same time. Hence, you must ensure that at least one node is always active in the cluster and is not part of the upgrade process. **Note:** All of the information must be entered in the specified format. You must not specify any value for blank or optional fields. Blank fields for alternate `vg` and `rootvg` disk indicates SKIP option from the user.
-F	Overrides certain default parameters to proceed with the VIOS upgrade operation. The **skipclusterstate** option skips the verification of the SSP cluster state, so that the VIOS upgrade operation can be triggered on multiple VIOS nodes simultaneously. **Note:** You must ensure that all SSP cluster nodes are not in the **DOWN** state at the same time. Otherwise, the SSP cluster services might become inactive permanently.
-I	Specifies the ISO image files that must be used to create the `ios_mksysb` image file.
-m	Specifies the `IOS_MKSYSB` resource name on the NIM Master server for the specified VIOS installation.
-n	Specifies the target VIOS host name to perform the VIOS upgrade operation.
-p	Specifies the resource object name of the Shared Product Object Tree (SP0T) for NIM installation.
-q	Queries the status of the VIOS upgrade operation.
-r	Specifies a new `rootvg` disk where the specified image must be installed. If you specify this flag, the existing `rootvg` disks are not used and the new disk is used for the installation.
-s	Skips the cloning of current `rootvg` disks to alternative disks and continues with the VIOS installation on the current `rootvg` disk. If the storage disks are not available, you can specify the *-s* flag to continue with the installation.

Flag name	Description
-t	Specifies the type of installation from the NIM server. The supported types are *bosinst* and *altdisk*. **bosinst** Indicates new and fresh installation on the current `rootvg` disk. **altdisk** Indicates a new installation on the alternative disk. The current `rootvg` disk on the VIOS partition is not impacted by this installation. The VIOS partition that has the current `rootvg` disk, remains in the running state during the installation of the alternative disk.
-v	Validates whether VIOS hosts are ready for the installation. The *-v* flag must be specified only for validation and can be used for preview of the installation image only.
-w	Specifies the directory path to create new `ios_mksysb` image file.
-x	Specifies the `ios_mksysb` image file name that must be created.
-y	Specifies new SPOT resource that must be created from the `ios_mksysb` resource.

Exit Status

Return code	Description
0	Success
1	Failure

Requirements

Consider the following requirements for the **viosupgrade** command:

- Alternate disks that are used with the *-a* and *-r* options as part of the **viosupgrade** command must be completely free. You must be able to list the disks by running the `lspv -free` command on the VIO Server.
- Installations through the **viosupgrade** command are categorized as a `New & Complete` installation. Any customized configurations on the current system that are running before the installation, including the time zone, are not carried to the new installation image. If you need to copy a customized file to the new image, use the *-e* flag with the **file_res** option. The *-e* flag with the **file_res** option specifies the customized file that must be backed up after the installation. You can use the **viosupgrade** command with the *-e* flag and **file_res** option only for *bosinst* type installations. After the installation, the customized files are copied to the `/home/padmin/backup_files` directory.

  Note: You must follow the manual steps on the NIM master to define the **file_res** resources before you run the **viosupgrade** command. The source directory, `/export/nim/viosupgrade/copyfiles`, is where the customized files are stored on the NIM master. The destination directory, `/home/padmin/backup_files`, is where the customized files are copied to on the VIOS after the installation.

  For example, to restore the files in the `/etc/environment` and `/var/custom.conf` files, complete the following steps:

  1. Create a source directory.

     ```
     mkdir -p /export/nim/viosupgrade/copyfiles
     ```

2. Define the *file_resource* option.

```
nim -o define -t file_res -a location=/export/nim/viosupgrade/copyfiles -a
dest_dir=/home/padmin/ backup_files -a server=master file_res_user
```

3. Create the `/export/nim/viosupgrade/copyfiles/etc` and `/export/nim/copyfiles/var` directories under the source directory.

```
mkdir -p /export/nim/viosupgrade/copyfiles/etc
mkdir -p /export/nim/copyfiles/var.
```

4. Copy the files from VIOS to the NIM master source directories.

```
scp -r root jaguar13:/etc/environment /export/nim/viosupgrade/copyfiles/etc
scp -r root jaguar13:/var/custom.conf /export/nim/viosupgrade/copyfiles/var
```

- The `viosbr` restore process does not support virtual device mappings with *vscsi* disks that are created on the VIO Server's `rootvg` disks. Therefore, the **viosupgrade** command cannot restore *vscsi* mappings if LVs are created from the VIO Server's `rootvg` disk.
- To enable the VIOS to be remotely managed by the AIX NIM master, run the **remote_management** command from VIOS.
- The target `ios_mksysb` image level considerations follow:
  - The level of the target `ios_mksysb` image must be at 3.1.0.00 level, or later.
  - The target `ios_mksysb` image level must be higher than the current VIOS `rootvg` level.
- The **viosupgrade** command on NIM server is supported from IBM AIX 7.2 with Technology Level 3, or later.
- For the NIM *bosinst* method of installation, the following are the current VIOS levels that are supported:
  - 2.2.6.30, or later for a Shared Storage Pool Cluster environment
  - 2.2.x.x, or later for a non-Shared Storage Pool Cluster environment
- On the NIM master, VIOS must be defined along with the Network Adapter Hardware Address (MAC) or Network Adapter Logical Device Name (ent name) in the NIM object's `if1` definition. If not defined, the **viosupgrade** command displays a message that the network boot operation might be delayed or fail.
- For a NIM *bosinst* type of installation, the interface resource (MAC or ent name), defined on the NIM object's `if1` definition for any given VIOS, must adhere to the following considerations:
  - The interface resource can be any available Ethernet interface that is connected to the network.
  - The interface resource can be a physical interface that is part of a Shared Ethernet Adapter (SEA).
  - The interface resource cannot be a Shared Ethernet Adapter (SEA) interface.

  This is mandatory for the **viosupgrade** command to trigger the VIOS restore process after the installation.
- For a NIM *altdisk* type of installation, the current VIOS levels that are supported are 2.2.6.30, or later.
- The NIM *altdisk* type of installation is not supported for VIOS when an SEA is the only primary interface for communication.

  Note: To enable the NIM *altdisk* type of installation for the above case, you must configure an additional Ethernet interface as the primary interface for remote management. The NIM master must be able to establish the network connection with the VIOS using the IP address that is configured on this interface by default.
- If a node is part of a cluster, then the hostname of the node must be resolvable during the metadata restore process after the installation is complete, This can be achieved either by passing the `resolv.conf` file or a script to add an entry in the `/etc/hosts` file, along with the *-e* option.
- If the VIOS belongs to an SSP cluster and if the current VIOS version is older than 2.2.6.30, the following two-step upgrade process is necessary to upgrade to VIOS 3.1.0.00, or later.

1. Upgrade the VIOS to version 2.2.6.30 through the upgrade methods such as by using the **updateios** command.
2. Use the **viosupgrade** command to upgrade to the VIOS to version 3.1.0.00, or later.

- If you are using `altdisk` method for installing the VIOS, and if the VIOS has `altinst_rootvg` or `old_rootvg` disks, the **viosupgrade** command fails indicating the user to rename the disks.
- The NIM master might reboot the VIOS and re-initiate the restore process to restore multiple virtual IO mappings through the *viosbr* restore process after the installation.
- If the NIM master fails to restore all of the mappings, you must manually re-initiate the restore operation on the NIM master by using the following commands:
  - VIOS-non SSP:

    ```
    nim -o viosbr -a viosbr_action=restore -a ios_backup=<BackupObjectName> <VIOSObjectName>
    ```

  - VIOS-SSP Cluster:

    ```
    nim -o viosbr -a viosbr_action=restore -a ios_backup=<BackupObjectName> -a
    clustername=<clusterName> -a viosbr_flags="-curnode"  <VIOSObjectName>
    ```

    Note: The *BackupObjectName* would generally be *<VIOSName>_backup*. You can get the list of backup object names from the command, `lsnim -t ios_backup`.
- If you installed additional software on the VIOS that is not part of the base VIOS image, the **viosupgrade** command might fail to restore configurations that are related to that software. To handle this case, you must create a customized VIOS image with the software applications that you want to include and provide this customized VIOS image as an input to the **viosupgrade** command for installation.

  The **viosupgrade** command identifies the software applications that are not included in the `ios_mksysb` image file, but are installed in the current system, such as the Subsystem Device Driver (SDD), or the Subsystem Device Driver Path Control Module (SDDPCM). The **viosupgrade** command displays a warning prompt on the console output during software validation or before installing the software. You can choose to continue or terminate the upgrade process by selecting the corresponding option. For example, if the software application that is installed on the VIOS is SDDPCM, you must customize the `ios_mksysb` image file to include the SDDPCM application and to provide the customized VIOS image file as an input to the **viosupgrade** command.

  Note: For more information about creating a customized VIOS image, refer to the **backupios** command.
- To create the `ios_mksysb` image file from the ISO image files, you require approximately 4 GB memory in the directory that you specify by using the **-w** flag in the **viosupgrade** command.

Examples

1. To validate the VIOS upgrade operation by using the *bosinst* method, type the following command:

   ```
   viosupgrade -v -t bosinst -n systemA -m mksysbA -p spotA -s
   ```

   Where, target VIOS node is **systemA**, mksysb image name is **mksysbA**, spot name is **spotA** and the type of install is *bosinst*.
2. To validate the VIOS upgrade operation by using the *bosinst* method, type the following command:

   ```
   viosupgrade -v -t bosinst -n clusternodeA -m mksysbA -p spotA -r hdisk1:hdisk2 -c -b clusterbackup -e resolv_conf:script
   ```

   Where, target cluster VIOS node is **clusternodeA**, mksysb image name is **mksysbA**, spot name is **spotA**, VIOS configuration backup resource is clusterbackup, NIM resources are resolv_conf and script, type of install is *bosinst* and provided new `rootvg` disks are hdisk1 and hdisk2.

3. To perform the VIOS upgrade operation by using the *bosinst* method by skipping the current `rootvg` cloning, type the following command

```
viosupgrade -t bosinst -n systemA -m mksysbA -p spotA -s
```

Where, target VIOS node is **systemA**, mksysb image name is **mksysbA**, spot name is **spotA** and type of install is *bosinst*.

Note: After the installation, the new `rootvg` will be the current `rootvg`.

4. To perform the VIOS upgrade operation by using the *bosinst* method by installing on provided disks, type the following command:

```
viosupgrade -t bosinst -n systemA -m mksysbA -p spotA -r hdisk1:hdisk2
```

Where, target VIOS node is **systemA**, mksysb image name is **mksysbA**, spot name is **spotA**, type of install is *bosinst* and provided new `rootvg` disks are hdisk1 and hdisk2.

Note: After reinstall new `rootvg` will be on the provided disks.

5. To perform the VIOS upgrade operation by using the bosinst method by doing the backup of current `rootvg` on provided alternate disks, type the following command:

```
viosupgrade -t bosinst -n systemA -m mksysbA -p spotA -a hdisk3:hdisk4
```

6. To perform the VIOS upgrade operation through *bosinst* method by using the provided VIOS configuration backup file, type the following command:

```
viosupgrade -t bosinst -n systemA -m mksysbA -p spotA -s -b backup
```

Where the target VIOS node is **systemA**, mksysb image name is **mksysbA**, spot name is **spotA**, VIOS configuration resource is backup, and the type of install is *bosinst*.

7. To perform the SSP cluster VIOS upgrade operation by using the *bosinst* method on the provided disks, type the following command:

```
viosupgrade -t bosinst -n clusternodeA -m mksysbA -p spotA -r hdisk1:hdisk2 -c
```

Where, target cluster VIOS node is **clusternodeA**, mksysb image name is **mksysbA**, spot name is **spotA**, type of install is *bosinst* and provided disks for new `rootvg` are hdisk1 and hdisk2.

8. You can perform the VIOS upgrade operation by using the *bosinst* method and the provided NIM resources. You can also create a backup of the current `rootvg` on a provided alternate disk. To perform all three of these operations, type the following command:

```
viosupgrade -t bosinst -n systemA -m mksysbA -p spotA -a hdisk3:hdisk4 -e resolv_conf:script:fb_script
```

Where, the target VIOS node is **systemA**, mksysb image name is **mksysbA**, spot name is **spotA**, type of install is *bosinst* and provided alternate `rootvg` disks are hdisk3 and hdisk4, provided NIM resources are resolv_conf, script, fb_script.

9. To validate the VIOS upgrade operation by using the *altdisk* method, type the following command:

```
viosupgrade -v -t altdisk -n systemA -m mksysbA -a hdisk3:hdisk4
```

Where, the target VIOS node is **systemA**, mksysb image name is **mksysbA**, alternate disks are hdisk3, hdisk4, and type of install is altdisk.

10. To perform the VIOS upgrade operation by using the *altinst* method, type the following command:

```
viosupgrade -t altdisk -n systemA -m mksysbA -a hdisk3:hdisk4
```

Where, the target VIOS node is **mysystemA**, mksysb image name is **mymksysbA**, type of install is *altdisk* and provided alternate `rootvg` disks are hdisk3 and hdisk4.

11. To perform the SSP cluster VIOS upgrade operation by using the *altinst* method, type the following command :

    ```
    viosupgrade -t altdisk -n clusternodeA -m mksysbA -a hdisk3:hdisk4:hdisk5 -c
    ```

    Where, the target cluster node is **clusternodeA**, mksysb image name is **mksysbA**, type of install is **altdisk** and provided alternate rootvg disks are hdisk3, hdisk4, hdisk5.

12. To validate the upgrade operation of one or more VIOS nodes in the provided file by using the *bosinst* method, type the following command:

    ```
    viosupgrade -v -t bosinst -f  "/usr/samples/nim/viosupgrade.inst"
    ```

    Note: Refer `/usr/samples/nim/viosupgrade.inst` for more information.

13. To perform the upgrade operation on a provided VIOS node in file by using the *bosinst* method, type the following command:

    ```
    viosupgrade -t bosinst -f  "/usr/samples/nim/viosupgrade.inst"
    ```

    If the target VIOS node is systemA, mksysb image name is mksysbA, spot name is spotA, provided alternate `rootvg` are hdisk3 and hdisk4, type of install is *bosinst* and file that contains VIOS node information is `/usr/samples/nim/viosupgrade.inst` then, file should contain the following data:

    ```
    systemA, mksysbA, spotA, ,hdisk3:hdisk4
    ```

14. To perform the upgrade operation on provided SSP cluster VIOS nodes in file through the *bosinst* method then, type the following command:

    ```
    viosupgrade -t bosinst -f  "/usr/samples/nim/viosupgrade.inst"
    ```

    If the target cluster VIOS node is **clusternodeA**, mksysb image name is **mksysbA**, spot name is **spotA**, type of install is *bosinst* and file that contains VIOS nodes information is `/usr/samples/nim/viosupgrade.inst` then, file must contains the following data:

    ```
    clusternodeA, mksysbA, spotA, , , , c,
    ```

15. To perform the upgrade operation on the provided VIOS nodes in the file by using the *altdisk* method then, type the following command:

    ```
    viosupgrade -v -t altdisk -f  "/usr/samples/nim/viosupgrade.inst"
    ```

16. To perform the VIOS upgrade operation by using the *bosinst* method and to restore files from the current `rootvg` to a newly installed image, type the following command:

    ```
    viosupgrade -t bosinst -n systemA -m mksysbA -p spotA -s -e file_res
    ```

    Where, the target VIOS node is **systemA**, mksysb image name is **mksysbA**, spot name is **spotA**, file_res is **file_res_user** and type of install is *bosinst*.

17. To create the `ios_mksysb` image file and SPOT resource from ISO image files, enter the following command:

    ```
    viosupgrade -I /home/padmin/dvdimage.v1.iso:/home/padmin/dvdimage.v2.iso -w /home/
    myNewIosMksysbImageDir
    -x myIosMkysysbRes -y mySPOTResource
    ```

    Where, `/home/padmin/dvdimage.v1.iso` and `/home/padmin/dvdimage.v2.iso` are the ISO image files, `/home/myNewIosMksysbImageDir` is the path where the `ios_mksysb` image file is created, `myIosMkysysbRes` is the name of the `ios_mksysb` image file that is created, and `mySPOTResource` is the name of the SPOT resource that is created.

18. To perform the VIOS upgrade operation by using the *bosinst* method on specified disks and to override the verification of the SSP cluster state, type the following command:

```
viosupgrade -t bosinst -n systemA -m mksysbA -p spotA -r hdisk1:hdisk2 -F skipclusterstate
-c
```

Where, **systemA** is the target VIOS node, **mksysbA** is the name of the mksysb image, **spotA** is the name of the SPOT resource, the type of installation is *bosinst*, and the specified new rootvg disks are hdisk1 and hdisk2.

vmh Command

Purpose

Starts a visual interface for use with MH commands.

Syntax

vmh [**-prompt** *String*] [**-vmhproc** *CommandString* | **-novmhproc**]

Description

The **vmh** command starts a visual interface for use with MH commands. The **vmh** command implements the server side of the MH window management protocol and maintains a split-screen interface to any program that implements the client side of the protocol.

The **vmh** command prompts for commands and sends them to the client side of the protocol. If the command produces a window with more than one screen of output, the **vmh** command prompts the user for a subcommand. The **vmh** subcommands enable you to display specific portions of the command output.

vmh Subcommands

Item	Description
Ctrl-L	Refreshes the screen.
Space	Advances to the next screen.
[*Number*] **Enter**	Advances the specified number of lines. The default is one line.
[*Number*] **d**	Advances 10 times the specified number of lines. The default for the *Number* variable is 1, for a total of 10 lines.
[*Number*] **g**	Goes to the specified line.
[*Number*] **G**	Goes to the end of the window. If the *Number* variable is specified, this command acts like the **g** flag.
[*Number*] **u**	Goes back 10 times the specified number of lines. The default for the *Number* variable is 1, for a total of 10 lines.
[*Number*] **y**	Goes back the specified number of lines. The default is one line.
h	Displays a help message.
q	Ends output.

Flags

Item	Description
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-novmhproc	Runs the default **vmproc** without the window management protocol.
-prompt *String*	Uses the specified string as the prompt.
-vmhproc *CommandString*	Specifies the program that implements the client side of the window management protocol. The default is the **msh** program.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Path:`	Specifies the user's MH directory.
`mshproc:`	Specifies the program used for the MH shell.

Files

Item	Description
$HOME/.mh_profile	Contains the MH user profile.
/usr/bin/vmh	Contains the **vmh** command.

vmo Command

Purpose

Manages Virtual Memory Manager tunable parameters.

Syntax

vmo [**-p**| **-r**] [**-y**] { **-o** *Tunable* [**=** *Newvalue*] }

vmo [**-p** |**-r**] [**-y**] { **-d** *Tunable* }

vmo [**-p** |**-r**] [**-y**] **-D**

vmo [**-p** | **-r**] [**-F**] **-a**

vmo -h [*Tunable*]

vmo [**-F**] **-L** [*Tunable*]

vmo [**-F**] **-x** [*Tunable*]

Note: Multiple **-o**, **-d**, **-x** and **-L** are allowed.

Description

Note: The **vmo** command can only be executed by root. The **vmo** command is a self-documenting command. The information about some of the flags or tunable parameters might be missing or out-of-date. You can find an up-to-date list of all the flags and by using the **-h**,**-L**, or **-x** flag.

Use the **vmo** command to configure Virtual Memory Manager tuning parameters. This command sets or displays current or next boot values for all Virtual Memory Manager tuning parameters. This command can also make permanent changes or defer changes until the next reboot. Whether the command sets or displays a parameter is determined by the accompanying flag. The **-o** flag specifies both actions. It can either display the value of a parameter or set a new value for a parameter.

The Virtual Memory Manager (VMM) maintains a list of free real-memory page frames. The page frames are available to hold virtual-memory pages that are needed to satisfy a page fault. When the number of pages on the free list falls below the values that are specified by the `minfree` parameter, the VMM begins to steal pages to add to the free list. The VMM continues to steal pages until the free list has at least the number of pages that are specified by the `maxfree` parameter.

If the number of file pages (permanent pages) in memory is less than the number specified by the `minperm%` parameter, the VMM steals frames from either computational or file pages, regardless of repage rates. If the number of file pages is greater than the number specified by the `maxperm%` parameter, the VMM steals frames only from file pages. Between the two, the VMM normally only steals file pages, but if the repage rate for file pages is higher than the repage rate for computational pages, computational pages are stolen as well.

You can also modify the thresholds that are used to decide when the system is running out of paging space. The `npswarn` parameter specifies the number of paging-space pages available at which the system begins warning processes that paging space is low. The `npskill` parameter specifies the number of paging-space pages available at which the system begins stopping processes to release paging space.

Note: Options **-o**, **-d**, and **-D**, which attempt to change the value of a virtual memory manager tunable parameter, are not supported within a workload partition.

Understanding the Effect of Changing Tunable Parameters

Misuse of this command can cause performance degradation or operating-system failure. Before you experiment with the **vmo** command, familiarize yourself with both Performance overview of the Virtual Memory Manager and Enhanced JFS file system cache limit with the maxclient parameter.

Before modifying any tunable parameter, you should first carefully read about all its characteristics in the Tunable Parameters section below, and follow any Refer To pointer, in order to fully understand its purpose.

You must then make sure that the Diagnosis and Tuning sections for this parameter truly apply to your situation and that changing the value of this parameter could help improve the performance of your system.

If the Diagnosis and Tuning sections both contain only "N/A", you should probably never change this parameter unless specifically directed by AIX development.

Flags

Item	Description
-a	Displays current, reboot (when used with the **-r** option), or permanent (when used with the **-p** option) values for all tunable parameters, one per line in pairs *Tunable* **=** ***Value***. For the permanent option, a value is displayed only for a parameter if its reboot and current values are equal. Otherwise, `NONE` is displayed as the value.

Item	Description
-d *Tunable*	Resets the *Tunable* parameter to its default value. If a *Tunable* parameter, which must be changed because it is not set to its default value, meets one or more of the following sets of criteria, a warning message is displayed and no change is made to the parameter: • The tunable parameter is of type **Bosboot** or **Reboot**. • The tunable parameter is of type **Incremental** and was changed from its default value, and the **-r** flag is not used in combination.
-D	Resets all *Tunable* values to their default values. If *Tunables* that need to be changed because they are not set to their default values meet one or more of the following sets of criteria, a warning message is displayed and no change is made: • The tunable is of type **Bosboot** or **Reboot**. • The tunable is of type **Incremental** and was changed from its default value, and **-r** is not used in combination.
-F	Forces display of the restricted tunable parameters when the **-a**, **-L** or **-x** options are specified alone on the command line to list all tunables. When the **-F** flag is not specified, restricted tunables are not displayed, unless these restricted tunables are specifically named with a display option.
-h [*Tunable*]	Displays help about the *Tunable* parameter if one is specified. Otherwise, displays the **vmo** command usage statement.
-L [*Tunable*]	Lists the characteristics of one or all tunables, one per line, using the following format:

```
NAME                      CUR    DEF    BOOT   MIN    MAX
UNIT           TYPE
     DEPENDENCIES
------------------------------------------------------------------------------
-----
memory_frames             128K          128K                 4KB
pages          S
------------------------------------------------------------------------------
-----
maxfree                   1088  1088    130    16     200K   4KB
pages          D
     minfree
     memory_frames
------------------------------------------------------------------------------
-----
minfree                   960    960    122    8      200K   4KB
pages          D
     maxfree
     memory_frames
------------------------------------------------------------------------------
-----
...
where:
    CUR = current value
    DEF = default value
    BOOT = reboot value
    MIN = minimal value
    MAX = maximum value
    UNIT = tunable unit of measure
    TYPE = parameter type: D (for Dynamic), S (for Static), R for Reboot),
           B (for Bosboot), M (for Mount), I (for Incremental),
           C (for Connect), and d (for Deprecated)
    DEPENDENCIES = list of dependent tunable parameters, one per line
```

Item	Description
-o *Tunable*[**=***Newvalue*]	Displays the value or sets tunable to *Newvalue*. If a tunable must be changed (the specified value is different than the current value), is of type **Bosboot** or **Reboot**, or if it is of type **Incremental** and its current value is greater than the specified value, and **-r** is not used in combination, the tunable value is not changed but a warning is displayed. When the **-r** flag is used in combination without a new value, the nextboot value for tunable is displayed. When **-p** is used in combination without a new value, a value is displayed only if the current and next boot values for the tunable are the same. Otherwise, NONE is displayed as the value.
-p	When used in combination with **-o**, **-d** or **-D**, makes changes apply to both current and reboot values, that is, turns on the updating of the **/etc/tunables/nextboot** file in addition to the updating of the current value. These combinations cannot be used on **Reboot** and **Bosboot** type parameters because their current value can't be changed. When used with **-a** or **-o** without specifying a new value, values are displayed only if the current and next boot values for a parameter are the same. Otherwise NONE is displayed as the value.
-r	When the **-r** flag is used with the **-a** or **-o** options without specifying a new value, the values are displayed only if the current and next boot values for a parameter are the same. Otherwise, NONE is displayed as the value. The **-r** flag changes the reboot values when it is used with the **-o**, **-d**, or **-D** flags. For example, you can update the `/etc/tunables/nextboot` file when you use the **-r** flag. If any parameter of type **Bosboot** is changed, the user will be prompted to run the **bosboot** command. When used with the **-a** or the **-o** options without specifying a new value, next boot values for tunables are displayed instead of current values.
-x [*Tunable*]	Lists characteristics of one or all tunables, one per line, using the following (spreadsheet) format: `tunable,current,default,reboot,min,max,unit,type,{dtunable }` `where:` `  current = current value` `  default = default value` `  reboot = reboot value` `  min = minimal value` `  max = maximum value` `  unit = tunable unit of measure` `  type = parameter type: D (for Dynamic), S (for Static), R (for Reboot),` `  B (for Bosboot), M (for Mount), I (for Incremental),` `  C (for Connect), and d (for Deprecated)` `  dtunable = list of dependent tunable parameters`
-y	Suppresses the confirmation prompt before running the **bosboot** command.

If a restricted tunable parameter is changed, a warning message is displayed that indicates that a tunable of the restricted use type has been modified. If the **-r** or the **-p** option is specified, you are prompted to confirm the change. In addition, at system reboot, restricted tunables that are displayed in the **/etc/tunables/nextboot** file and are changed to values that are different from their default values (using a command line specifying the **-r** or **-p** option) causes an error log entry that identifies the list of these changed tunables.

When modifying tunable, the tunable value might be specified using abbreviations such as K, M, G, T, P and E to indicate units. See the following lists for abbreviations and their correspondent values:

- K=2^{10}

- M=2^{20}
- G=2^{30}
- T=2^{40}
- P=2^{50}
- E=2^{60}

Thus, a tunable value of 1024 might be specified as 1K.

Any change (with **-o**, **-d** or **-D**) to a parameter of type Mount will result in a message being displayed to warn the user that the change is only effective for future mountings.

Any change (with **-o**, **-d** or **-D** flags) to a parameter of type Connect will result in **inetd** being restarted, and a message displaying a warning to the user that the change is only effective for future socket connections.

Any attempt to change (with **-o**, **-d** or **-D**) a parameter of type **Bosboot** or **Reboot** without **-r**, will result in an error message.

Any attempt to change (with **-o**, **-d** or **-D** but without **-r**) the current value of a parameter of type Incremental with a new value smaller than the current value, will result in an error message.

Tunable Parameters Type

All the tunable parameters manipulated by the tuning commands (**no**, **nfso**, **vmo**, **ioo**, `raso`, and **schedo**) have been classified into these categories:

Item	Description
Dynamic	If the parameter can be changed at any time
Static	If the parameter can never be changed
Reboot	If the parameter can only be changed during reboot
Bosboot	If the parameter can only be changed by running bosboot and rebooting the machine
Mount	If changes to the parameter are only effective for future file systems or directory mounts
Incremental	If the parameter can only be incremented, except at boot time
Connect	If changes to the parameter are only effective for future socket connections
Deprecated	If changing this parameter is no longer supported by the current release of AIX.

For parameters of type Bosboot, whenever a change is performed, the tuning commands automatically prompt the user to ask if they want to execute the **bosboot** command. For parameters of type Connect, the tuning commands automatically restart the **inetd** daemon.

Note that the current set of parameters managed by the **vmo** command only includes Static, Dynamic, and Bosboot types.

Compatibility Mode

When running in compatibility mode (controlled by the **pre520tune** attribute of **sys**0), reboot values for parameters, except those of type Bosboot, are not really meaningful because in this mode they are not applied at boot time. For more information, see *Performance management*.

In compatibility mode, you can set reboot values to tuning parameters by imbedding calls to tuning commands in scripts called during the boot sequence. Parameters of type **Reboot** can be set without using the **-r** flag, so that existing scripts continue to work.

Tunable Parameters

To view the default and range of values allowed for the tunables, run the **vmo** command with the **-h** option as follows:

```
vmo -h <tunable_parameter_name>
```

Tunable	Description
ame_cpus_per_pool	**Purpose** Determines the ratio of CPUs per compressed memory pool. For every *ame_cpus_per_pool* CPUs, at least one compressed memory pool is created. **Tuning** Lower ratios are used to reduce contention on compressed memory pools. This ratio is not the only factor used to determine the number of compressed memory pools (amount of memory and the layout is also considered) so certain changes to this ratio may not result in any change to the number of compressed memory pools.
ame_maxfree_mem	**Purpose** Specifies the average amount of free memory in a compressed memory pool free list at which the VMM will shrink the compressed pool. **Tuning** Excessive shrink and grow operations can occur if compressed memory pool size tends to change significantly. This can occur if the workload working set size frequently changes. Increase this tunable to raise the threshold at which the VMM will shrink a compressed memory pool and thus reduce the number of overall shrink and grow operations.
ame_min_ucpool_size	**Purpose** Defines the minimum size of the uncompressed pool. **Tuning** If compressed memory pool grows too large, there may not be enough space in memory to house uncompressed memory which can slow down application performance due to excessive use of the compressed memory pool. Increase this value to limit the size of the compressed memory pool and make more uncompressed pages available.

Tunable	Description
ame_minfree_mem	**Purpose** Specifies the amount of free memory in a compressed memory pool free list at which the VMM will grow the compressed pool. **Tuning** If processes are being delayed waiting for compressed memory to become available, increase *ame_minfree_mem* to improve response time. Note, that this must be at least 64 KB less than *ame_maxfree_mem*.
ame_mpsize_support	**Purpose** Enables all supported page sizes in an Active Memory Expansion (AME) environment for POWER8 processor-based servers, or later, which supports the 64 KB accelerator. **Tuning** A value of 0 enables legacy behavior in an AME environment. In this case, only 4 KB and 16 MB page sizes are enabled. A value of 1 enables all supported page sizes in an AME environment. You can change this tunable parameter only in POWER8 processor-based servers, or later, which supports the 64 KB accelerator.
ams_loan_policy	**Purpose** This tunable toggles the loaning behavior when shared memory mode is enabled. **Tuning** When the tunable is set to 0, loaning is disabled. When set to 1, loaning of file cache is enabled. When set to 2, loaning of any type of data is enabled. In response to low memory in the AMS pool, the VMM will free memory and loan it to the hypervisor.
force_relalias_lite	**Purpose** If set to 0, a heuristic is used, when tearing down a **mmap** region, to determine when to avoid locking the source **mmapped** segment **Tuning** This is a scalability tradeoff, controlled by *relalias_percentage*, possibly costing more compute time used. If set to 1, the source segment lock is avoided whenever possible, regardless of the value of *relalias_percentage*.

Tunable	Description
kernel_heap_psize	**Purpose** Specifies the default page size to use for the kernel heap. **Tuning** This is an advisory setting. Support for 64 KB pages is provided by POWER5+ and later machines and used when *vmm_mpsize_support* is enabled. The 16 MB pages, provided by POWER4 and later machines, should only be used for the kernel heap under high performance environments. A value of 0 indicates that the kernel will use the preferred default value of 64 KB, if that page size is supported, else 4 KB pages are used.
lgpg_regions	**Purpose** Specifies the number of large pages to reserve for implementing with the **shmget()** system call with the SHM_LGPAGE flag **Tuning** The *lgpg_size* parameter must also be used in addition to this option. The application must be modified to specify the SHM_LGPAGE flag when calling **shmget()**. This will improve performance in the case where there are many TLB misses and large amounts of memory is being accessed. Although this parameter is Dynamic on DLPAR-capable systems, the **nextboot** value is written into the boot image when a **bosboot** command is run so that the setting is optimally restored at reboot.
lgpg_size	**Purpose** Specifies the size in bytes of the hardware-supported large pages used for the implementation for the **shmget()** system call with the SHM_LGPAGE flag. **Tuning** Supported on systems from POWER4 onwards. Although this parameter is Dynamic on DLPAR-capable systems, the **nextboot** value is written into the boot image when a **bosboot** command is issued so that the setting is optimally restored at reboot. The *lgpg_regions* parameter must be set to a non-zero value in addition to this parameter. The application must be modified to specify the SHM_LGPAGE flag when calling the **shmget()** subroutine. This will improve the performance in the case where there are many TLB misses and large amounts of memory is being accessed.

Tunable	Description
low_ps_handling	**Purpose** Specifies the action to change the system behavior in relation to process termination during low paging space conditions. **Tuning** A value of 1 indicates current behavior of process termination on low paging space. A value of 2 indicates a new behavior where processes with SIGDANGER handler will be killed, if no other processes were found earlier to recover from low paging space condition.
maxfree	**Purpose** Specifies the number of frames on the free list at which page-stealing is to stop. **Tuning** Observe free-list-size changes with **vmstat -n** command. If the **vmstat -n** command displays the free-list size frequently driven below *minfree* by application demands, increase the *maxfree* value to reduce calls to replenish the free list. Setting the value too high causes page replacement to run for a longer period of time. The difference between *maxfree* and *minfree* should be of the order of *maxpgahead*, and no less than 8.
maxpin%	**Purpose** Specifies the maximum percentage of real memory that can be pinned. **Tuning** Change if cannot pin memory, although free memory is available. If this value is changed, the new value should ensure that at least 4 MB of real memory will be left unpinned for use by the kernel. The **vmo** command converts **maxpin%** to the corresponding **maxpin** absolute value, which is the value used by the kernel. Change this parameter only in extreme situations, such as maximum-load benchmarking. This dynamic parameter will have its **nextboot** value written into the boot image if a **bosboot** command is issued.
memory_frames	**Purpose** Number of valid memory frames. **Tuning** N/A

Tunable	Description
memplace_data	**Purpose** Specifies the default memory placement policy for data. **Tuning** Refers to the data of the main executable (initialized data, BSS), heap, shared library and object modules loaded at run-time. Data placement can be set to first-touch (value of 1), round-robin across the system (value of 2) or automatic (value of 0), where the system decides the best placement for the memory.
memplace_mapped_file	**Purpose** Specifies the default memory placement policy for files that are mapped into the address space of a process (such as through **shmat()** and **mmap()**). **Tuning** Default placement of memory mapped files can be set to first-touch (value of 1) or round-robin across the system (value of 2) or automatic (value of 0), where the system decides the best placement for the memory.
memplace_shm_anonymous	**Purpose** Specifies the default memory placement policy for anonymous shared memory. **Tuning** Anonymous shared memory refers to working storage memory, created via **shmget()** or **mmap()**, that can be accessed only by the creating process or its descendants. This memory is not associated with a name (or key). Default placement of anonymous shared memory can be set to first-touch (value of 1) or round-robin across the system (value of 2) or automatic (value of 0), where the system decides the best placement for the memory.

Tunable	Description
memplace_shm_named	**Purpose** Specifies the default memory placement policy for named shared memory. **Tuning** Named shared memory refers to working storage memory, created via **shmget()** or **shm_open()**, which is associated with a name (or key) that allows more than one process to access it simultaneously. Default placement of named shared memory can be set to first-touch (value of 1) or round-robin across the system (value of 2) or automatic (value of 0), where the system decides the best placement for the memory.
memplace_stack	**Purpose** Specifies the default memory placement policies for the program stack. **Tuning** Stack placement can be set to first-touch (value of 1) or round-robin across the system (value of 2) or automatic (value of 0), where the system decides the best placement for the memory.
memplace_text	**Purpose** Specifies the default memory placement policy for the application text. **Tuning** This applies only to the text of the main executable and not to its dependencies. Text placement can be set to first-touch (value of 1) or round-robin across the system (value of 2) or automatic (value of 0), where the system decides the best placement for the memory.
memplace_unmapped_file	**Purpose** Specifies the default memory placement policy for unmapped file access, such as through **read()**/**write()**. **Tuning** Default placement of unmapped file access can be set to first-touch (value of 1) or round-robin across the system (value of 2) or automatic (value of 0), where the system decides the best placement for the memory.

Tunable	Description
minfree	**Purpose** Specifies the number of frames on the free list at which the VMM starts to steal pages to replenish the free list. **Tuning** Page replacement occurs when the number of free frames reaches *minfree*. If the processes are being delayed by page stealing, increase *minfree* to improve response time. The difference between *maxfree* and *minfree* should be of the order of *maxpgahead*, and no less than 8.
minperm%	**Purpose** Specifies the point below which the page-stealer will steal file or computational pages regardless of repaging rates. **Tuning** You can decrease this parameter if large number of file pages in memory is causing working storage pages to be replaced. On the other hand, if some files are known to be read repetitively, and I/O rates do not decrease with time from startup, *minperm* may be too low.
nokilluid	**Purpose** The user IDs lower than this value will be exempt from getting killed due to low page-space conditions. **Tuning** A value of 0 indicates off. Useful when system is out of paging space and the system administration processes are being killed. Either set this tunable to 1 in order to protect specific user ID processes from getting killed due to low page space or ensure there is sufficient paging space available.

Tunable	Description
npsassert	**Purpose** Asserts the LPAR when the system runs out of paging space. **Tuning** • A value of 0 for this tunable provides legacy behavior where processes are killed when free paging space reaches the *npskill* threshold. Vmo tunable *low_ps_handling* influences which process is killed. SIGDANGER signal is sent to processes when the free paging space reaches danger levels. • A value of 1 for this tunable asserts the LPAR when paging space is completely exhausted. Processes are not killed when free paging space reaches *npskill* levels. SIGDANGER signal mechanism is suppressed, and processes are not notified when free paging space reaches danger levels.
npskill	**Purpose** Specifies the number of free paging-space pages at which the operating system begins killing processes. **Tuning** The default value is the maximum of 64 and (number of paging space pages)/128. The *npskill* value must be greater than zero and less than the total number of paging space pages on the system.
npswarn	**Purpose** Specifies the number of free paging-space pages at which the operating system begins sending the SIGDANGER signal to processes. **Tuning** The default value is the maximum of 512 and (4**npskill*). The value of *npswarn* must be greater than zero and less than the total number of paging space pages on the system. Increase the value if you experience processes being killed because of low paging space.
numpsblks	**Purpose** Total number of paging-space blocks. **Tuning** N/A

Tunable	Description
pinnable_frames	**Purpose** Number of pages available for pinning **Tuning** N/A
relalias_percentage	**Purpose** If *force_relalias_lite* is set to 0, then this specifies the factor used in the heuristic to decide whether to avoid locking the source **mmapped** segment. **Tuning** This is used when tearing down an **mmapped** region and is a scalability statement, where avoiding the lock may help system throughput, but, in some cases, at the cost of more compute time used. If the number of pages being unmapped is less than this value divided by 100 and multiplied by the total number of pages in memory in the source **mmapped** segment, then the source lock is avoided. A value of 0 for *relalias_percentage*, with *force_relalias_lite* also set to 0, will cause the source segment lock to always be taken. Effective values for *relalias_percentage* will vary by workload, however, a suggested value is 200.
scrub	**Purpose** Enables or Disables freeing of paging space disk blocks from pages in memory for Deferred Page Space Allocation Policy pages. **Tuning** A value of 0 disables scrubbing completely. A value of 1 enables scrubbing of in memory paging space disk blocks when the number of system free paging space blocks is below **npsscrubmin**, and continues until above **npsscrubmax**.
v_pinshm	**Purpose** If set to 1, will allow pinning of shared memory segments. **Tuning** A value of 0 indicates off. Change this value when the overhead is high and in pinning or unpinning of AIO buffers from shared memory segments. Useful only if the application also sets the SHM_PIN flag when doing a **shmget()** call and if doing async I/O from shared memory segments.

Tunable	Description
vmm_default_pspa	**Purpose** This tunable controls the default aggressiveness of page size promotion. The value is an abstract aggressiveness weighting which is treated by the operating system as the inverse of the page promotion threshold. **Tuning** A value of 0 for the *vmm_default_pspa* setting is equivalent to a page promotion threshold of 100%, that is, a memory range must have 100% real memory occupancy in order to be promoted. A value of 100 for the *vmm_default_pspa* setting is equivalent to a page promotion threshold of 0%, that is, a memory range should be promoted immediately on first reference to memory in the range. A value of -1 for the *vmm_default_pspa* setting is equivalent to a page promotion threshold of -1, that is, never do page promotion for a memory range. Page size promotion thresholds are only considered at segment creation time. Thus, changing *vmm_default_pspa* will only affect the page size promotion thresholds for segments created after the tunable is adjusted.
wlm_memlimit_nonpg	**Purpose** Selects whether non-pageable page sizes (16M, 16G) are included in the WLM *realmem* and *virtmem* counts. If 1 is selected, then non-pageable page sizes are included in the *realmem* and *virtmem* limits count. If 0 is selected, then only pageable page sizes (4K, 64K) are included in the *realmem* and *virtmem* counts. This value can only be changed when WLM Memory Accounting is off, or the change will fail. **Tuning** When this tunable is set to 0, WLM virtual and real memory limits will only apply to pageable pages consumed by a WLM class. Because heavy use of pageable pages is what causes paging on a system, a value of 0 provides more granular control over how much a WLM class pages when non-pageable pages are in use. This tunable should only be adjusted when WLM real or virtual memory limits are being used on a system configured with non-pageable pages.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list the current and reboot value, range, unit, type and dependencies of all tunable parameters managed by the **vmo** command, enter:

   ```
   vmo -L
   ```

2. To turn on and reserve 16MB large pages on a POWER4 system, enter:

   ```
   vmo -o lgpg_regions=10 -o lgpg_size=16777216
   ```

   This command will propose **bosboot** to the user, and warn that a reboot is necessary before the change will be effective.

3. To display help on **nokilluid**, enter:

   ```
   vmo -h nokilluid
   ```

4. To turn on **v_pinshm** after the next reboot, enter:

   ```
   vmo -r -o v_pinshm=1
   ```

5. To permanently reset all **vmo** tunable parameters to default, enter:

   ```
   vmo -p -D
   ```

6. To list the reboot value for all virtual memory manager tuning parameters, enter:

   ```
   vmo -r -a
   ```

7. To list (spreadsheet format) the current and reboot value, range, unit, type and dependencies of all tunable parameters managed by the **vmo** command, enter:

   ```
   vmo -x
   ```

vmstat Command

Purpose

Reports virtual memory statistics.

Syntax

vmstat [**-f**] [**-i**] [**-s**] [**-I**[**-W**]] [**-t**] [**-v**] [**-h**] [**-w**] [**-l**] [**-c**] [**-@** *wparname* | **ALL**] [{ **-p** | **-P** } *physicalvolume* | **ALL] ALL]** [-S *power*] [*physicalvolume*] [*interval*] [*count*]

Note: Do not use the *wparname* parameter and the `-i` flag together inside workload partitions.

Description

The **vmstat** command reports statistics about kernel threads, virtual memory, disks, hypervisor pages, traps, and processor activity. Reports that are generated by the **vmstat** command can be used to balance system load activity. These system-wide statistics (among all processors) are calculated as averages for values that are expressed as percentages, and as sums otherwise. The **vmstat** command might return inconsistent statistics because the statistics are not read atomically.

If you run the **vmstat** command without flags, the report contains a summary of the virtual memory activity since system startup. If you specify the *-f* flag, the **vmstat** command reports the number of forks since system startup. The *physicalvolume* parameter specifies the name of the physical volume.

The *interval* parameter specifies the amount of time in seconds between each report. If you do not specify the *interval* parameter, the **vmstat** command generates a single report that contains statistics for the time since system startup and then exits. You can specify the *count* parameter only with the

interval parameter. If you specify the *count* parameter, its value determines the number of reports that are generated and the number of seconds apart. If you specify the *interval* parameter without the *count* parameter, reports are continuously generated. Do not specify a value of zero to the *count* parameter.

The kernel maintains statistics for kernel threads, paging, and interrupt activity, which the **vmstat** command accesses by using the *perfstat* kernel extension. The disk input/output statistics are maintained by device drivers. For disks, the average transfer rate is determined by using the active time and number of times information is being transferred. The percent active time is computed from the amount of time the drive is busy during the report.

The `vmstat` command reports the number of physical processors consumed (pc), and the percentage of entitlement consumed (ec), in Micro-Partitioning environments. These metrics display on the Micro-Partitioning environments.

The report that is generated by the **vmstat** command contains a system configuration row and column headings. If the -@ flag is specified, the report consists of system configuration and WPAR configuration. The system configuration row has the following values:

lcpu
Indicates the number of logical processors.

mem
Indicates the amount of memory.

tmem
Indicates the true memory size of the LPAR.

Note: This flag is available only when *-c* option is provided and Active Memory Expansion is enabled.

ent
Indicates the entitled capacity. Displays only when the partition is running with shared processor.

drives
Indicates the number of disks. Displays only when physical volume name is monitored.

WPARs
Indicates the number of active workload partitions. It is displayed only when the -@ flag is specified.

memlim
Indicates the limit of the memory resource of the workload partition. The limit is in megabytes (MB). This information is displayed only for the WPAR with enforced memory resource limit.

cpulim
Indicates the limit of processor resource of the workload partition in processor units. This information is displayed only for the WPAR with enforced processor resource limit.

rset
Indicates the type of the **rset** registry that is associated with a WPAR. The type can be regular or exclusive. This information is displayed only for the WPARs that are associated with a **rset** registry.

mmode
Indicates memory mode. This metric is displayed automatically in a system with Active Memory Sharing enabled. This metric is also displayed when **-c** option is used.

mpsz
Size of the memory pool in gigabytes. This metric is displayed only in shared-memory mode.

The column headings and their descriptions follow:

WPAR: Information about workload partitions. It displays only when the-@ flag is specified.

WPAR
Workload partition name.

Notes:

1. The *system* WPAR name indicates system-wide statistics. The *global* WPAR name indicates the statistics belong to Global only.

2. When the **vmstat** command is started with the *-@ ALL* option and the WPAR specific information is not available for a metric, then an en dash sign (-) is displayed instead of a value.
3. When the **vmstat** command is started with the *-@ wparname* or started inside a WPAR, if the WPAR information is not available for a metric, then that metric is marked with the at sign (@), and the system-wide value is displayed for that metric.
4. If a metric is not supported, then a en dash sign (-) is displayed instead of a value.

kthr: Information about kernel thread states.

r
Average number of runnable kernel threads over the sampling interval. Runnable threads consist of the threads that are ready but still waiting to run, and the threads that are already running.

b
Average number of kernel threads that are placed in the Virtual Memory Manager (VMM) wait queue (awaiting resource, awaiting input/output) over the sampling interval.

Memory: Information about the usage of virtual and real memory. Virtual pages are considered active if they are accessed. A page is 4096 bytes.

avm
Active virtual pages.

fre
Size of the free list.

Note: A large portion of real memory is used as a cache for file system data. It is not unusual for the size of the free list to remain small.

Page: Information about page faults and paging activity. This information is averaged over the interval and given in units per second.

re
Pager input/output list.

pi
Pages that are paged in from paging space.

po
Pages paged out to paging space.

fr
Pages freed (page replacement).

sr
Pages that are scanned by page-replacement algorithm.

cy
Clock cycles by page-replacement algorithm.

Faults: Trap and interrupt rate averages per second over the sampling interval.

in
Device interrupts.

sy
System calls.

cs
Kernel thread context switches.

CPU: Breakdown of percentage usage of processor time.

us
User time.

If the current physical processor consumption of the uncapped partitions exceeds the entitled capacity, the percentage becomes relative to the number of physical processor consumed (pc).

sy
System time.

If the current physical processor consumption of the uncapped partitions exceeds the entitled capacity, the percentage becomes relative to the number of physical processor consumed (pc).

id
Processor idle time.

If the current physical processor consumption of the uncapped partitions exceeds the entitled capacity, the percentage becomes relative to the number of physical processor consumed (pc).

wa
Processor idle time during which the system had outstanding disk/NFS I/O request.

If the current physical processor consumption of the uncapped partitions exceeds the entitled capacity, the percentage becomes relative to the number of physical processor consumed (pc).

pc
Number of physical processors used. Displayed only if the partition is running with shared processor.

ec
The percentage of entitled capacity that is consumed. Displayed only if the partition is running with shared processor. Because the time base over which this data is computed can vary, the entitled capacity percentage can sometimes exceed 100%. This excess is noticeable only with small sampling intervals.

rc
The percentage of processor resource that is used. This information is displayed only for the WPARs with enforced processor resource limit.

Disk: Provides the number of transfers per second to the specified physical volumes that occurred in the sample interval. The *physicalvolume* parameter can be used to specify one to four names. Transfer statistics are given for each specified drive in the order specified. This count represents requests to the physical device. It does not imply an amount of data that was read or written. Several logical requests can be combined into one physical request. If the *physicalvolume* parameter is used, the physical volume names are printed at the beginning of command execution.

If the **-I** flag is specified, an I/O oriented view is presented with the following column changes.

kthr
The **p** column is displayed in addition to the **r** and **b** columns.

p
Average number of threads waiting for I/O messages from raw devices. Raw devices are the devices that are directly attached to the system.

If the **-W** flag is specified along with the **-I** flag, an additional **w** column is also displayed along with the **r**, **b**, and **p** flags.

w
Number of threads per second of time that are waiting for the file system direct I/O event to occur. These events include the following types:

- Asynchronous I/O (AIO)
- Buffer cache subsystem
- Concurrent I/O (CIO)
- File system direct I/O
- NFS subsystem
- A thread is waiting for an action from the virtual memory manager (VMM) waiting list.

page
New **fi** and **fo** columns are displayed instead of the **re** and **cy** columns.

fi
File page-ins per second.

fo
File page-outs per second.

If the **-c** flag is specified, Active Memory Expansion view is presented with the following column changes.

memory
The columns **csz**, **cfr**, and **dxm** are displayed besides columns **avm** and **fre.**

csz
Current compressed pool size, in 4K page units.

cfr
Free pages available in compressed pool, in 4K page units.

dxm
Deficit in Expanded Memory Size, in 4K page units.

page
New columns **ci** and **co** are displayed instead of **re** and **cy** columns.

ci
Number of page-ins per second from compressed pool.

co
Number of page-outs per second to compressed pool.

If while the `vmstat` command is running, there is a change in system configuration that affects the output, `vmstat` prints a warning message about the configuration change. It then continues the output after printing the updated system configuration information and the header.

If the **-l** flag is specified, an additional "large-page" section is displayed with the following columns:

alp
Indicates the number of large pages currently in use.

flp
Indicates the number of large pages on the large page freelist.

If the **-p** option is specified, additional lines of VMM statistics are displayed for the specified page sizes. With **-I** and **-t** options, the **-p** option produces an additional line for the specified page size. This line contains the following VMM statistics relevant to the specified page size:

- **avm**
- **fre**
- **re**
- **fi**
- **fo**
- **pi**
- **po**
- **ci**
- **co**
- **fr**
- **sr**
- **cy**

Notes:

1. The display of the **re**, **fi**, **fo**, and **cy** options are affected by the **-I** option.
2. The display of the **re**, **ci**, **co**, and **cy** options are affected by the **-c** option.

3. If there is no resource control, then the **avm** and **fre** options are system-wide. Therefore, with the **-@** option set, both the **avm** and **fre** options will be marked with the at sign (@).

These VMM statistics are preceded by a **psz** column and followed by an **siz** column. The description of these two columns follows:

psz
Page size (for example, 4 KB, 64 KB).

siz
Number of frames of the specified page size that exist on the system.

With the **-s** option, the **-p** option produces a separate stanza of output that contains only the statistics relevant to the specified page size. This additional stanza is preceded by a page size header.

The **-P** option produces the following report for the specified page size:

pgsz
Indicates the page size (for example, 4 KB, 64 KB).

Memory
Indicates the memory statistics for the specified page sizes.

siz
The number of frames of the specified page size that exist on the system.

avm
Active virtual pages applicable to the specified page size.

fre
Size of the free list for the specified page size.

Page
Indicates the relevant page faults and paging activity for the specified page size. The page-related columns **re**, **pi**, **po**, **fr**, **sr**, **cy**, **fi**, **fo**, **ci**, and **co** are also applicable to this report.

Flags

Note: If the `-f` (or `-s`) flag is entered on the command line, then the system accepts the `-f` (or `-s`) flag and ignores other flags. If both the `-f` and `-s` flags are specified, the system accepts only the first flag and ignore the second flag.

Item	Description
-@ *wparname*	Reports the Virtual Memory activity of a workload partition: • The **-@ ALL** option indicates that the report pertains to the system and global environment, in addition to all of the workload partitions in the system. **Note:** The values that are system-wide statistics are marked with dash sign (-) against the WPAR section. • The **-@** *wparname* flag indicates that the activity is only for that workload partition. In a workload partition, if you specify the **-@** flag, system-wide statistics and workload partition statistics are displayed. The system-wide statistics are marked with the at sign (@). **Note:** Do not use the **-@** flag with any combination of the **-i** flag.
-c	Displays memory compression statistics with the new columns of output, **csz**, **cfr**, and **dxm** under the heading memory, and columns **ci** and **co** under the heading page instead of the columns **re** and **cy.** **Note:** This option is available only when Active Memory Expansion is enabled.
-f	Reports the number of forks since system startup.
-i	Displays the number of interrupts that are taken by each device since system startup. **Note:** The **-I**, **-t**, **-w**, and **-l** flags are ignored when they are specified with the **-i** flag.
-I	Displays I/O oriented view with the new columns of output, **p** under heading `kthr`,and columns **fi** and **fo** under heading page instead of the columns **re** and **cy** in the page heading.
-l	Displays an extra "large-page" section with the `alp` and `flp` columns.
-p *pagesize*	Appends the VMM statistics for the specified page size to the regular **`vmstat`** output.
-P *pagesize*	Displays only the VMM statistics, which are relevant for the specified page size.

Item	Description
-s	Writes to standard output the contents of the sum structure, which contains an absolute count of paging events since system initialization. The **-s** flag can only be used with the **-v** flag. These events are described as follows: **address translation faults** Incremented for each occurrence of an address translation page fault. I/O may or may not be required to resolve the page fault. Storage protection page faults (lock misses) are not included in this count.
-s	**backtracks** Incremented for each page fault that occurs while resolving a previous page fault. (The new page fault must be resolved first and then initial page faults can be *backtracked*.) **CPU context switches** Incremented for each processor context switch (dispatch of a new process). **decrementer interrupts** Incremented on each decrementer interrupt. **device interrupts** Incremented on each hardware interrupt. **executable-filled page faults** Incremented for each instruction page fault. **extend XPT waits** Incremented each time that a process is waited by VMM due to a commit in progress for the segments accessed. **free frame waits** Incremented each time that a process requests a page frame. The free list is empty, and the process is forced to wait while the free list is replenished. **iodones** Incremented at the completion of each VMM I/O request. **mpc send interrupts** Incremented on each mpc send interrupt. **mpc receive interrupts** Incremented on each mpc receive interrupt. **page ins** Incremented for each page read in by the virtual memory manager. The count is incremented for page ins from page space and file space. Along with the page-out statistic, this value represents the total amount of real I/O initiated by the virtual memory manager. **page outs** Incremented for each page that is written out by the virtual memory manager. The count is incremented for page outs to page space and for page outs to file space. Along with the page in statistic, this statistic represents the total amount of real I/O initiated by the virtual memory manager. **paging space page ins** Incremented for VMM initiated page ins from paging space only. **paging space page outs** Incremented for VMM initiated page outs to paging space only. **pages examined by the clock** VMM uses a clock-algorithm to implement a pseudo least recently used (lru) page replacement scheme. Pages are *aged* by being examined by the clock. This count is incremented for each page examined by the clock. **pages freed by the clock** Incremented for each page the clock algorithm selects to free from real memory. **pending I/O waits** Incremented each time that a process is waited by VMM for a page-in I/O to complete.

Item	Description
-s	**phantom interrupts** Incremented on each phantom interrupt **revolutions of the clock hand** Incremented for each VMM clock revolution (that is after each complete scan of memory). **start I/Os** Incremented for each read or write I/O request that is initiated by VMM. **syscalls** Incremented for each system call. **total reclaims** Incremented when an address translation fault can be satisfied without initiating a new I/O request. This can occur if the page has been previously requested by VMM, but the I/O has not yet completed; or if the page was pre-fetched by VMM's read-ahead algorithm, but was hidden from the faulting segment; or if the page has been put on the free list and has not yet been reused. **traps** Not maintained by the operating system. **zero-filled page faults** Incremented if the page fault is to working storage and can be satisfied by assigning a frame and zero-filling it. When the **-c** flag is specified along with the **-s** flag, the following additional metrics are displayed. **compressed pool page ins** Number of page-ins from Compressed Pool since system boot. **compressed pool page outs** Number of page-outs to Compressed Pool since system boot.
-s	When used with the **-p** *pagesize* option, the **-s** option appends the sum structure for the specified page size to the system-wide sum structure. This additional stanza is preceded by a page size header (for example, 4K pages). The following details are not be displayed in this pagesize-based stanza as these statistics are not related to page sizes: • Processor context switches • Device interrupts • Software interrupts • Decrementer interrupts • MPC-sent interrupts • MPC-received interrupts • Phantom interrupts • Traps • Syscalls **Notes:** 1. When the **-s** flag is used with the **-@ ALL** option, the system-wide statistics are repeated in the workload partition section. 2. When the **-s** flag is used with the *wparname* option, all metrics are reported and the system-wide statistics are marked with the at sign (@). 3. When the **-s** flag is used with the **-l** flag, the **vmstat** command displays the following metric: **large-page hi water count** Specifies the maximum value of the large-page inuse count.
-S *power*	Multiplies the statistics of the processor with a value of 10^{power}. The default value of the power is 0. The following statistics are scaled: • **us** • **sy** • **id** • **wa** • **pc** • **ec** **Notes:** 1. Do not use the **-S** flag with the **-f**, **-s**, **-i**, **-v**, or **-p** flags. 2. When the **-S** flag is specified, the **us**, **sy**, **id**, and **wa** statistics change. By default, the **us**, **sy**, **id**, and **wa** statistics are relative to the processor consumption of WPAR. When the **-S** flag is specified with a value of power that is not equal to zero, these statistics will be relative to system-wide processor consumption. 3. The value of power for **-S** flag can be only between 0 and 3.
-t	Prints the time-stamp next to each line of output of **vmstat**. The time-stamp is displayed in the HH:MM:SS format. **Note:** Time stamp is not be printed if **-f**, **-s**, or **-i** flags are specified.

Item	Description
-v	Writes to standard output various statistics maintained by the Virtual Memory Manager. The **-v** flag can only be used with the **-s** and the **-h** flags. If you specify the **-v** flag, the following statistics are displayed: **compressed percentage** Percentage of memory used by compressed pages. **client filesystem I/Os blocked with no fsbuf** Number of client filesystem I/O requests blocked because no fsbuf was available. NFS (Network File System) and VxFS (Veritas) are client filesystems. Fsbuf are pinned memory buffers used to hold I/O requests in the filesystem layer. **client pages** Number of client pages. **compressed pages** Number of compressed memory pages. **external pager filesystem I/Os blocked with no fsbuf** Number of external pager client filesystem I/O requests blocked because no fsbuf was available. JFS2 is an external pager client filesystem. Fsbuf are pinned memory buffers used to hold I/O requests in the filesystem layer. **file pages** Number of 4 KB pages that are currently used by the file cache. **free pages** Number of free 4 KB pages. **filesystem I/Os blocked with no fsbuf** Number of filesystem I/O requests blocked because no fsbuf was available. Fsbuf are pinned memory buffers used to hold I/O requests in the filesystem layer. **lruable pages** Number of 4 KB pages that are considered for replacement. This number excludes the pages that are used for VMM internal pages, and the pages that are used for the pinned part of the kernel text. **maxclient percentage** Tuning parameter (managed using vmo) specifying the maximum percentage of memory, which can be used for client pages. **maxperm percentage** Tuning parameter (managed using **vmo**) in percentage of real memory. **maxpin percentage** Tuning parameter (managed using **vmo**) specifying the percentage of real memory which can be pinned. **memory pages** Size of real memory in number of 4 KB pages. **memory pools** Tuning parameter (managed using **vmo**) specifying the number of memory pools. **minperm percentage** Tuning parameter (managed using **vmo**) in percentage of real memory. **numclient percentage** Percentage of memory that is occupied by client pages.

Item	Description
-v	*(Statistics that are displayed by **-v**, continued)*: **numperm percentage** Percentage of memory that is currently used by the file cache. **paging space I/Os blocked with no psbuf** Number of paging space I/O requests that are blocked because the psbuf space is not available. The psbufs space is pinned memory buffers that are used to hold I/O requests at the virtual memory manager layer. **pending disk I/Os blocked with no pbuf** Number of pending disk I/O requests blocked because no pbuf was available. Pbufs are pinned memory buffers used to hold I/O requests at the logical volume manager layer **pinned pages** Number of pinned 4 KB pages. **Note:** When the kernel locking feature (`vmm_klock_mode` parameter) is enabled, the pinned pages include the kernel locking (**klocked**) pages. For more information about the kernel locking feature, enter the following command: `vmo -h vmm_klock_mode`. **remote pageouts scheduled** Number of pageouts scheduled for client file systems. If you specify the **-h** flag with the **-v** flag, the following additional metrics are displayed: **Time resolving virtualized partition memory page faults** The total time that the virtual partition is blocked to wait for the resolution of its memory page fault. The time is measured in seconds, with millisecond granularity. **Virtualized partition memory page faults** The total number of virtual partition memory page faults that are recorded for the virtualized partition. **Number of 4 KB page frames loaned** The number of the 4 KB pages of partition's memory loaned to the hypervisor. **Percentage of partition memory loaned** The percentage of the partition's memory loaned to the hypervisor. **Notes:** 1. When the **-v** flag is used with the **-@ ALL** option, the system-wide statistics are not repeated in the workload partition section. 2. When the **-s** flag is used with the *wparname* option, all metrics are reported and the system-wide statistics are marked with the at sign (@). When the **-c** flag is specified along with the **-v** flag, the following additional metrics are displayed: **Compressed Pool Size** Size of Compressed Pool, in 4K page unit. **percentage of true memory that is used for compressed pool** Percentage of unexpanded memory that is used for compressed pool. **free pages in compressed pool (4K pages)** Number of free pages in compressed pool, in 4K page unit. **target memory expansion factor** Target memory expansion factor that is configured for the LPAR. **achieved memory expansion factor** Current memory expansion factor achieved.
-h	Displays the **hypv-page** section that includes the hypervisor page information. The **hypv-page** section contains the following metrics: **hpi** Number of hypervisor page-in per second. **hpit** Average time that is spent in milliseconds per hypervisor page-in. **pmem** Amount of physical memory that is backing the logical memory of partitions. The value is measured in gigabytes. If you specify the **-h** flag with the **-v** flag, the following metrics are displayed in addition to the metrics that are displayed using the **-v** flag: **Time resolving virtualized partition memory page faults** The total time that the virtual partition is blocked to wait for the resolution of its memory page fault. The time is measured in seconds, with millisecond granularity. **Virtualized partition memory page faults** The total number of virtual partition memory page faults that are recorded for the virtualized partition. **Number of 4 KB page frames loaned** The number of the 4 KB pages of the memory that is loaned to the hypervisor in the partition. **Percentage of partition memory loaned** The percentage of the memory loaned to the hypervisor in the partition.
-w	Displays the report in wide mode.
-W	Displays an extra field **w** in the **kthr** section. This option is allowed only with **-I** flag.

Notes:

1. If Active Memory Expansion is enabled, the **vmstat** reports memory statistics in the expanded view. However, if the environment variable AME_MEMVIEW is set to TRUE, the memory statistics represent the true view.
2. The AME_MEMVIEW environment variable has no impact on memory statistics reported using the **-c** option.

Examples

1. To display a summary of the statistics since boot, enter the following command:

   ```
   vmstat
   ```

2. To display five summaries at 2-second intervals, enter the following command:

   ```
   vmstat 2 5
   ```

3. To display a summary of the statistics since boot including statistics for logical disks scdisk13 and scdisk14, enter the following command:

   ```
   vmstat scdisk13 scdisk14
   ```

4. To display fork statistics, enter the following command:

   ```
   vmstat -f
   ```

5. To display the count of various events, enter the following command:

   ```
   vmstat -s
   ```

6. To display time-stamp next to each column of output of **vmstat**, enter the following command:

   ```
   vmstat -t
   ```

7. To display the I/O oriented view with an alternative set of columns, enter the following command:

   ```
   vmstat -I
   ```

8. To display all the VMM statistics available, enter the following command:

   ```
   vmstat -vs
   ```

9. To display the large-page section with the `alp` and `flp` columns at 8-second intervals, enter the following command:

   ```
   vmstat -l 8
   ```

10. To display the VMM statistics specific to a particular page size (in the example, 4 KB), enter the following command:

    ```
    vmstat -p 4K
    ```

11. To display the VMM statistics for all page sizes that are supported on the system, enter the following command:

    ```
    vmstat -p ALL
    ```

    Or enter the following command:

    ```
    vmstat -p all
    ```

12. To display only the VMM statistics for a particular page size (in this example, 4 KB), enter the following command:

```
vmstat -P 4K
```

13. To display only the per-page breakdown of VMM statistics for all supported page sizes, enter the following command:

    ```
    vmstat -P ALL
    ```

    Or enter the following command:

    ```
    vmstat -P all
    ```

14. To display a summary of the statistics for all of the workload partitions after boot, enter the following command:

    ```
    vmstat -@ ALL
    ```

15. To display all of the VMM statistics available for all of the workload partitions, enter the following command:

    ```
    vmstat –vs -@ ALL
    ```

16. To display both WPAR and system-wide VMM statistics from a workload partition, enter the following command:

    ```
    vmstat -@
    ```

17. To multiply the processor values with 10 and display the results, enter the following command:

    ```
    vmstat -S 1
    ```

18. To display the statistics for the hypervisor page, enter the following command:

    ```
    vmstat –h
    ```

19. To display the information about pages that are loaned to the hypervisor, enter the following command:

    ```
    vmstat -vh
    ```

20. To display memory compression statistics (in an LPAR with Active Memory Expansion enabled), enter the following command:

    ```
    vmstat -c
    ```

21. To display memory compression statistics specific to per-pagesize (in an LPAR with Active Memory Expansion enabled), enter the following command:

    ```
    vmstat –c –P ALL
    ```

22. To append memory compression information to the statistics displayed by **-s**option (in an LPAR with Active Memory Expansion enabled), enter the following command:

    ```
    vmstat –s -c
    ```

23. To append memory compression information to the statistics displayed by **-v**option (in an LPAR with Active Memory Expansion enabled), enter the following command:

    ```
    vmstat –v -c
    ```

Files

Item	Description
/**usr**/**bin**/**vmstat**	Contains the **vmstat** command.

vpdadd Command

Purpose

Adds entries to the product, lpp, history, and vendor databases.

Syntax

vpdadd { **-c** *Component* | **-p** *Product* | **-f** *Feature* } **-v** *v.r.m.f* [**-D** *Destdir*] [**-U** *Command*] [**-R** *Prereq*] [**-S** *Msg_Set*] [**-M** *Msg_Number*] [**-C** *Msg_Catalog*] [**-P** *Parent*] [**-I** *Description*]

Description

The **vpdadd** command is for use with or by installers that wish to be listed in Vital Product Database (VPD). The VPD consists of the product, lpp, and history databases. Entries to the inventory database must be added by the **sysck** command. A new vendor database is now included to track products that use destination directories and **non-installp** uninstallers.

The **vpdadd** command uses a tree structure of *Product* at the highest level, then *Feature*, and then *Component*.

The *Component* is the lowest installable unit, but in this hierarchy, a *Component* is not selectable for install or uninstall. Therefore, if an installer is using the **vpdadd** command to update the install database, they should look at their own tree representation and add entries based on their structure. If only adding one entry per install, then adding a *Product* type rather than *Component* type would allow that entry to be listed in the uninstall SMIT interfaces. All the entries are made in the VPD, but *Components* and *Features* are filtered out in the default **lslpp** listings (**-Lc**).

Flags

Item	Description
-C *Msg_Catalog*	Specifies the message catalog to search for a translated description of the *Component*. The default (English) description is specified with the **-I** flag. If the message catalog is not in the standard NLSPATH, then the full path name should be given.
-c *Component*	Specifies the *Component* name to add to the VPD. An entry is only added if it is unique. Uniqueness is described as having a different destination directory. If the same instance of a *Component* is already in the database, then no entry is added, and an error is returned. This allows a force install (that is, reinstall).
-D *Destdir*	Specifies the root (prefix) path that is added to all the files in a *Component* when being installed (and when being added to the inventory database by the **sysck** command). Files in a *Component* are listed with relative path names, so the root path is allowed to change. The default destination directory is **/opt**.
-f *Feature*	Specifies the *Feature* name to add to the VPD. An entry is only added if it is unique. Uniqueness is described as having a different VRMF or destination directory. If the same instance of a *Feature* is already in the database, then no entry is added, and an error is not returned. This allows for a force install (that is, reinstall).

Item	Description
-I *Description*	Specifies the default description of the *Component*, *Feature* or *Product*. The description must be specified in double quotation marks. Single quotation marks are allowed inside the description, and double quotation marks must be prepended with a ****.
-M *Msg_Number*	Specifies the message number for the description.
-P *Parent*	Specifies the parent software unit. A *Component* specifies either a *Feature* or a *Product* as its parent, depending on where it was in the tree.
-p *Product*	Specifies the *Product* name to add to the VPD. An entry is only added if it is unique. Uniqueness is described as having a different VRMF or destination directory. If the same instance of a *Product* is already in the database, then no entry is added, and an error is not returned. This allows a force install (that is, reinstall).
-R *Prereq*	Specifies a *Component* (fileset) that is a requisite of the installing *Component*. The argument must be specified in quotation marks. This flag can be used more than once to specify multiple prerequisites. Although these are treated as prerequisites at install time (by the installer), they are listed as corequisites in the *Product* database to avoid creating circular requisite chains.
-S *Msg_Set*	Specifies the message set (if more than one in the catalog).
-U *Command*	Specifies the *Command* to launch the uninstaller for this *Component*. This may be just a command path name, or it may include parameters if there is a global uninstaller. The **geninstall** command calls this uninstaller, and **installp** does not deinstall a fileset if this value is set in the VPD.
-v *v.r.m.f*	The VRMF of the *Component*, *Feature* or *Product* being added.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. The following example shows how the Registry service would call **vpdadd** to add a *Component* for the *Foo product*. This *Component* has two requisites, one that is specific to the operating system, and one that is listed as GUID.

```
vpdadd -c EPL2890198489F -v 1.2.3.0 -R "bos.rte.odm 4.3.3.0" -R "8KDEOKY90245686 1.1.0.0"  \
-U /usr/opt/foo/uninstaller.class -p KID892KYLIE25 -I "Foo Database Component"
```

2. To add a new product `devices.pci.cool.rte` to the VPD, enter:

```
vpdadd -p devices.pci.cool.rte -v 5.1.0.0 -U /usr/sbin/udisetup
```

Files

/usr/sbin/vpdadd

vpddel Command

Purpose

Removes entries from the product, lpp, history, and vendor databases.

Syntax

vpddel { **-c** *Component* | **-p** *Product* | **-f** *Feature* } **-v** *V.R.M.F* **-D** *Dest_dir*

Description

The **vpddel** command removes entries from the product, lpp, history, and vendor databases. The vrmf and destination directory must be specified so that the correct entries are removed.

Flags

Item	Description
-c *Component*	Removes the specified *Component*. The VRMF must also be included when removing a *Component*.
-D *Dest_dir*	Specifies the destination directory of the *Component* to remove. If a destination directory is not included, then the default **/opt** is used.
-f *Feature*	Specifies the *Feature* to remove from the vendor database.
-p *Product*	The *Product* to remove from the vendor database.
-v *V.R.M.F*	Specifies the version, release, modification and fix level of the *component* to delete from the VPD and vendor database.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Example

To remove the *Component* EPL2890198489F from the product, history, lpp, and vendor databases, type:

```
vpddel -c EPL2890198489F -v 1.2.3.0 -D /usr/lpp/Foo
```

Files

/usr/sbin/vpddel

vsdatalst Command

Purpose

vsdatalst – Displays virtual shared disk subsystem information.

Syntax

vsdatalst {**-g** | **-n** | **-v** | **-c**}

Description

Use this command to display one of several kinds of information to standard output.

You can use the System Management Interface Tool (SMIT) to run the **vsdatalst** command. To use SMIT, enter:

```
smit list_vsd
```

and select the option for the kind of virtual shared disk SDR information you want to see.

Flags

Only one of the following flags can be specified with each invocation of **vsdatalst**:

-g
Displays the following global volume group data:

global_group_name,
local_group_name,
primary_server_node,
secondary_server_node. (This is only enabled with the Recoverable virtual shared disk subsystem.)
eio_recovery
recovery
CVSD *server_list*

-n
Displays the following Node data:

node_number,
host_name,
adapter_name,
min_buddy_buffer_size,
max_buddy_buffer_size,
max_buddy_buffers.

-v
Displays the following definition data:

vsd_name,
logical_volume_name,
global_group_name,
minor_number.

-c
Displays the following cluster information:

node_number
cluster_name

Parameters

None.

Security

You must have root authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

1. To display global volume group date, enter:

```
vsdatalst -g
```

The system displays a message similar to the following:

```
VSD Global Volume Group Information
                                                Server Node Numbers
Global Volume Group name   Local VG name   primary   backup eio_recovery  recovery  server_list  vsd_type
-------------------------    ------------   -------   ------ ------------  --------  -----------  --------
gpfs0gvg                     gpfs0vg          1         2          0          0         0            VSD
gpfs1gvg                     gpfs1vg          2         1          0          0         0            VSD
gpfs3gvg                     gpfs3vg          1         0          0          0         1:2          CVSD
```

2. To display global volume group date, enter:

```
vsdatalst -n
```

The system displays a message similar to the following:

```
VSD Node Information
                                                 Buddy Buffer
  node                  VSD      IP packet minimum maximum # maxbufs
number host_name       adapter    size     size    size
------ --------------- -------- --------- ------- ------- ---------
     1 host1            ml0       61440    4096   262144    128
     2 host2            ml0       61440    4096   262144    128
```

3. To display global volume group date, enter:

```
vsdatalst -v
```

The system displays a message similar to the following:

```
VSD Table
VSD name                      logical volume  Global Volume Group             minor# size_in_MB
----------------------------- --------------- ------------------------------- ------ ----------
gpfs0vsd                      gpfs0lv         gpfs0gvg                             3 4096
gpfs1vsd                      gpfs1lv         gpfs1gvg                             1 4096
gpfs3vsd                      gpfs3lv         gpfs3gvg                             4 4096
```

Location

`/opt/rsct/vsd/bin/vsdatalst`

vsdchgserver Command

Purpose

`vsdchgserver` – Switches the server function for one or more virtual shared disks from the node that is currently acting as the server node to the other.

Syntax

vsdchgserver
-g *vsd_global_volume_group_name* -p *primary_node*

[-b *secondary_node*] [-o *EIO_recovery*]

Description

The `vsdchgserver` command allows the serving function for a global volume group defined on a primary node to be taken over by the secondary node, or to be taken over by the primary node from the secondary node. This allows an application to continue to use virtual shared disks in situations where the cable or adapter between the physical disks and one of the attached nodes is not working.

The Recoverable virtual shared disk subsystem automatically updates the virtual shared disk devices if, and only if, the `vsdchgserver` command is used to flip the currently-defined primary node and secondary node in the global volume group specified in the `-g` flag.

Flags

-g
Specifies the Global Volume Group name for the volume group that represents all the virtual shared disks defined on a particular node.

-p
Specifies the primary server node number for the global volume group.

-b
Specifies the secondary node number for the global volume group. If the **-b** flag is not specified, the secondary node definition will be removed.

-o
Specified as `0`, for no recovery on an EIO error, or `1`, for recovery on an EIO error.

Parameters

None.

Security

You must have root authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

To change the primary server node for the global volume group node12vg to node 1 and the secondary node to node 2, with EIO recovery, enter:

```
vsdchgserver -g node12vg -p 1 -b 2 -o 1
```

Location

`/opt/rsct/vsd/bin/vsdchgserver`

vsdelnode Command

Purpose

Removes virtual shared disk information for a node or series of nodes.

Syntax

vsdelnode *node_number* ...

Description

This command is used to remove virtual shared disk data for a node or series of nodes.

The **vsdelnode** command makes the listed nodes no longer virtual shared disk nodes so that no virtual shared disks can be accessed from them. This command is unsuccessful for any nodes that are servers for any global volume groups.

You can use the System Management Interface Tool (SMIT) to run the **vsdelnode** command. To use SMIT, enter:

```
smit delete_vsd
```

and select the **Delete Virtual Shared Disk Node** Information option.

Flags

-g
Specifies the Global Volume Group name for the volume group that represents all the virtual shared disks defined on a particular node.

-p
Specifies the primary server node number for the global volume group.

-b
Specifies the secondary node number for the global volume group. If the **-b** flag is not specified, the secondary node definition will be removed.

-o
Specified as `0`, for no recovery on an EIO error, or `1`, for recovery on an EIO error.

Parameters

node_number
Specifies the node number of the node whose virtual shared disk information you want to remove.

Security

You must have `root` authority to run this command.

Restrictions

The recoverable virtual shared disk subsystem must be stopped on the node(s) you are deleting. Otherwise, the results may be unpredictable. For more information, see *RSCT for AIX 5L Managing Shared Disks* .

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to *RSCT Administration Guide*.

Examples

To delete virtual shared disk node information for nodes **3** and **6**, enter:

```
vsdelnode 3 6
```

Location

`/opt/rsct/vsd/bin/vsdelnode`

vsdelvg Command

Purpose

vsdelvg – Removes virtual shared disk global volume group information.

Syntax

vsdelvg [**-f**] *global_group_name* ...

Description

Use this command to remove virtual shared disk global volume group information. If any virtual shared disks are defined on a global volume group, the **vsdelvg** command is unsuccessful unless **-f** is specified. If **-f** is specified, any such virtual shared disks must be unconfigured and in the defined state on all the virtual shared disk nodes to be deleted.

You can use the System Management Interface Tool (SMIT) to run the **vsdelvg** command. To use SMIT, enter:

```
smit delete_vsd
```

and select the **Delete Virtual Shared Disk Global Volume Group Information** option.

Flags

-f
Forces the removal of any virtual shared disks defined on this global volume group.

Parameters

global_group_name
Specifies the volume group that you no longer want to be global to the system.

Security

You must have root authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

To remove the global volume group **vg1n1**, enter:

```
vsdelvg vg1n1
```

Location

`/opt/rsct/vsd/bin/vsdelvg`

vsdnode Command

Purpose

Define virtual shared disk information for a node or series of nodes.

Syntax

vsdnode
node_number... adapter_name min_buddy_buffer_size

max_buddy_buffer_size max_buddy_buffers

vsd_max_ip_msg_size [*cluster_name*]

Description

Use this command to make the specified nodes virtual shared disk nodes and to assign their virtual shared disk operational parameters. If this information is the same for all nodes, run this command once. If the information is different for the nodes, run this command once for each block of nodes that should have the same virtual shared disk information.

You can use the System Management Interface Tool (SMIT) to run the **vsdnode** command. To use SMIT, enter:

```
smit vsd_data
```

and select the **virtual shared disk Node Information** option.

Flags

-f
Forces the removal of any virtual shared disks defined on this global volume group.

Parameters

node_number
Specifies the node or nodes whose virtual shared disk information is to be set. The value you specify for *node_number* must match a valid RSCT remote peer domain node number.

adapter_name
Specifies the adapter name to be used for virtual shared disk communications for the nodes specified. The adapter name must already be defined to the nodes. Note that the nodes involved in virtual shared disk support must be fully connected so that proper communications can take place. Use **ml0** to specify that the virtual shared disk device driver transmits data requests over the SP Switch. The **ml0** adapter will be used the next time the virtual shared disk device driver is loaded.

min_buddy_buffer_size
Specifies the smallest buddy buffer a server uses to satisfy a remote request to a virtual shared disk. This value must be a power of 2 and greater than or equal to 4096. The suggested value is 4096 (4 KB). For a 512 byte request, 4 KB is excessive. However, recall that a buddy buffer is only used for the short period of time while a remote request is being processed at the server node.

max_buddy_buffer_size
Specifies the largest buddy buffer a server uses to satisfy a remote noncached request. This value must be a power of 2 and greater than or equal to the *min_buddy_buffer_size*. The suggested value is 262144 (256 KB). This value depends on the I/O request size of applications using the virtual shared disks and the network used by the virtual shared disk software.

max_buddy_buffers
Specifies the number of *max_buddy_buffer_size* buffers to allocate. The virtual shared disk device driver will have an initial size when first loaded, and then will dynamically allocate and reclaim additional space as needed. The suggested value is 2000 256 KB buffers.

Buddy buffers are only used on the servers. On client nodes you may want to set *max_buddy_buffers* to 1.

Note: The `statvsd` command will indicate if remote requests are queueing waiting for buddy buffers.

vsd_max_ip_msg_size
Specifies the maximum message size in bytes for virtual shared disks. This value must not be greater than the maximum transmission unit (MTU) size of the network. The recommended values are:

- 61440 (60KB) for a switch
- 8192 (8KB) for jumbo frame Ethernet
- 1024 (1KB) for 1500-byte MTU Ethernet

cluster_name
A cluster name must be specified for server nodes that will be serving concurrently accessed shared disks. The cluster name can be any user provided name. A node can only belong to one cluster. For example, when you have a concurrent access environment, the two servers for the CVSD must both specify the same cluster name.

Note: The *cluster_name* is required only for SSA (Serial Storage Architecture) disks.

Security

You must have `root` authority to run this command.

Restrictions

The node specified on this command must already belong to a peer domain, and you must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to *RSCT Administration Guide*.

Examples

The following example defines information for a switch network and nodes 1 through 8.

```
vsdnode 1 2 3 4 5 6 7 8 ml0 4096 262144 128 61440
```

Location

`/opt/rsct/vsd/bin/vsdnode`

vsdsklst Command

Purpose

Produces output that shows you the disk resources used by the virtual shared disk subsystem across a peer domain.

Syntax

vsdsklst [**-v**] [**-d**] {**-a** | **-n** *node_number*[, node_number2, ...]}

Description

Use this command to check disk utilization across a peer domain.

Flags

-v
Displays only disk utilization information about volume groups and the virtual shared disks associated with them.

-d
Displays only disk utilization information about volume groups and the physical disks associated with them.

-a
Displays specified information for all nodes in the system or system partition.

-n *node_number*
Lists one or more node numbers for which information is to be displayed.

Parameters

node_number
Specifies the node or nodes whose virtual shared disk information is to be set. The value you specify for *node_number* must match a valid RSCT remote peer domain node number.

adapter_name
Specifies the adapter name to be used for virtual shared disk communications for the nodes specified. The adapter name must already be defined to the nodes. Note that the nodes involved in virtual shared disk support must be fully connected so that proper communications can take place. Use **ml0** to specify that the virtual shared disk device driver transmits data requests over the SP Switch. The **ml0** adapter will be used the next time the virtual shared disk device driver is loaded.

min_buddy_buffer_size
Specifies the smallest buddy buffer a server uses to satisfy a remote request to a virtual shared disk. This value must be a power of 2 and greater than or equal to 4096. The suggested value is 4096 (4 KB). For a 512 byte request, 4KB is excessive. However, recall that a buddy buffer is only used for the short period of time while a remote request is being processed at the server node.

max_buddy_buffer_size
Specifies the largest buddy buffer a server uses to satisfy a remote noncached request. This value must be a power of 2 and greater than or equal to the *min_buddy_buffer_size*. The suggested value is 262144 (256 KB). This value depends on the I/O request size of applications using the virtual shared disks and the network used by the virtual shared disk software.

max_buddy_buffers
Specifies the number of *max_buddy_buffer_size* buffers to allocate. The virtual shared disk device driver will have an initial size when first loaded, and then will dynamically allocate and reclaim additional space as needed. The suggested value is 2000 256KB buffers.

Buddy buffers are only used on the servers. On client nodes you may want to set *max_buddy_buffers* to 1.

Note: The `statvsd` command will indicate if remote requests are queueing waiting for buddy buffers.

vsd_max_ip_msg_size
Specifies the maximum message size in bytes for virtual shared disks. This value must not be greater than the maximum transmission unit (MTU) size of the network. The recommended values are:

- 61440 (60KB) for a switch
- 8192 (8KB) for jumbo frame Ethernet
- 1024 (1KB) for 1500-byte MTU Ethernet

cluster_name
A cluster name must be specified for server nodes that will be serving concurrently accessed shared disks. The cluster name can be any user provided name. A node can only belong to one cluster. For example, when you have a concurrent access environment, the two servers for the CVSD must both specify the same cluster name.

Note: The *cluster_name* is required only for SSA (Serial Storage Architecture) disks.

Security

You must have `root` authority to run this command.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to *RSCT Administration Guide*.

Examples

This command:

```
vsdsklst -dv -a
```

displays the following information on a system that has volume groups and virtual shared disks defined on nodes 1 and 2.

```
c164n12.ppd.pok.ibm.com: Node Number:2; Node Name:c164n12.ppd.pok.ibm.com
c164n12.ppd.pok.ibm.com: Volume group:rootvg; Partition Size:32; Total:271; Free:168
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk0; Total:271; Free:168
c164n12.ppd.pok.ibm.com:     Volume group:testvg is not varied on.
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk5;
c164n12.ppd.pok.ibm.com: Volume group:test1vg; Partition Size:4; Total:537; Free:534
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk2; Total:537; Free:534
c164n12.ppd.pok.ibm.com:         VSD Name:vsd1n2[testnewlv21n2]; Size:1
c164n12.ppd.pok.ibm.com:         VSD Name:vsd2n2[testlv1n2]; Size:346112.25
c164n12.ppd.pok.ibm.com:         VSD Name:vsd3n2[testlv2n2]; Size:346112.25
c164n12.ppd.pok.ibm.com:     Volume group:vg1 is not varied on.
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk9;
c164n12.ppd.pok.ibm.com:     Volume group:sharkvg is not varied on.
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk7;
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk10;
c164n12.ppd.pok.ibm.com: Volume group:bdhclvg; Partition Size:32; Total:134; Free:102
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk13; Total:134; Free:102
c164n12.ppd.pok.ibm.com: Volume group:gpfs0vg; Partition Size:8; Total:536; Free:0
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk12; Total:536; Free:0
c164n12.ppd.pok.ibm.com:         VSD Name:gpfs0vsd[gpfs0lv]; Size:352256.75
c164n12.ppd.pok.ibm.com:     Not allocated physical disks:
c164n12.ppd.pok.ibm.com:         Physical disk:hdisk1
c164n12.ppd.pok.ibm.com:         Physical disk:hdisk3
c164n12.ppd.pok.ibm.com:         Physical disk:hdisk4
c164n12.ppd.pok.ibm.com:         Physical disk:hdisk6
c164n12.ppd.pok.ibm.com:         Physical disk:hdisk11
c164n12.ppd.pok.ibm.com:         Physical disk:hdisk15
c164n11.ppd.pok.ibm.com: Node Number:1; Node Name:c164n11.ppd.pok.ibm.com
c164n11.ppd.pok.ibm.com: Volume group:rootvg; Partition Size:32; Total:271; Free:172
c164n11.ppd.pok.ibm.com:         Physical Disk:hdisk0; Total:271; Free:172
c164n11.ppd.pok.ibm.com: Volume group:bdhclvg; Partition Size:32; Total:134; Free:102
c164n11.ppd.pok.ibm.com:         Physical Disk:hdisk9; Total:134; Free:102
c164n11.ppd.pok.ibm.com:         VSD Name:bdhcvsd1n1[lvbdhcvsd1n1]; Size:45056
c164n11.ppd.pok.ibm.com: Volume group:testvg; Partition Size:16; Total:134; Free:70
c164n11.ppd.pok.ibm.com:         Physical Disk:hdisk13; Total:134; Free:70
c164n11.ppd.pok.ibm.com:     Not allocated physical disks:
c164n11.ppd.pok.ibm.com:         Physical disk:hdisk1
c164n11.ppd.pok.ibm.com:         Physical disk:hdisk2
c164n11.ppd.pok.ibm.com:         Physical disk:hdisk3
```

Location

`/opt/rsct/vsd/bin/vsdsklst`

vsdvg Command

Purpose

Defines a virtual shared disk global volume group.

Syntax

vsdvg
[**-g** *global_volume_group*] {`-l` *server_list local_group_name* | *local_group_name primary_node* [*secondary node* [*eio_recovery*]]}

Description

Use this command to define volume groups for use by the Virtual shared disk subsystem. This is done by specifying the local volume group name, the node on which it resides, and the name by which the volume group will be known throughout the cluster.

You can use the System Management Interface Tool (SMIT) to run the **vsdvg** command. To use SMIT, enter the following command and select the **Virtual Shared Disk Global Volume Group Information** option:

```
smit vsd_data
```

Flags

-g *global_volume_group*
Specifies a unique name for the new global volume group. This name must be unique across the system partition. It should be unique across the SP, to avoid any naming conflicts during future system partitioning operations. The suggested naming convention is **vg***xx***n***yy*, where *yy* is the node number, and *xx* uniquely numbers the volume groups on that node. If this is not specified, the local group name is used for the global name. The length of the name must be less than or equal to 31 characters.

-l *server_list*
Define the list of servers for CVSD. More than one server indicates the *global_volume_group* is a concurrent volume group.

Parameters

local_group_name
Specifies the name of a volume group that you want to indicate as being used for virtual shared disks. This name is local to the host upon which it resides. The length of the name must be less than or equal to 15 characters.

primary_node
Specifies the primary server node number on which the volume group resides. The length of the name must be less than or equal to 31 characters.

secondary_node
Specifies the secondary server node number on which the volume group resides. The length of the name must be less than or equal to 31 characters.

eio_recovery

Specifies how the Recoverable virtual shared disk subsystem will respond to EIO errors. If *eio_recovery* is set to the value 1 (the default), an EIO error will cause the Recoverable virtual shared disk system to flip the current primary node and the secondary node and perform one more retry on the new primary node.

Security

You must have root authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

nonzero
Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

1. The following example defines `gpfs1gvg` as a virtual shared disk global volume group with the local volume group `gpfs1vg` accessed from `node1` as the *primary_node* and `node2` as the *secondary_node*.

   ```
   vsdvg -g gpfs1gvg gpfs1vg 1 2
   ```

2. The following example defines `gpfs3gvg` as a virtual shared disk global volume group with the local volume group `gpfs3vg` concurrently accessed from `node1` and `node2`.

   ```
   vsdvg -g gpfs3gvg -l 1:2 gpfs3vg
   ```

Location

`/opt/rsct/vsd/bin/vsdvg`

vsdvgts Command

Purpose

Updates the timestamp used by the Recoverable virtual shared disk subsystem by reading the timestamp from the volume group descriptor area (VGDA) of the physical disks.

Syntax

vsdvgts [**-a**] [*volgrp*]

Description

Use this command to update the timestamp that the Recoverable virtual shared disk subsystem uses to determine if a twin-tailed volume group has changed. When the subsystem detects a change, the recovery scripts export the volume group and then import the volume group.

This command can be used to avoid exporting the volume group and then importing the volume group during recovery in situations where the export and import operations are not really necessary. This command should be used very carefully.

Flags

-a
: Specifies that the timestamps for this volume group for both primary and secondary nodes should be updated. If this flag is not specified, the timestamp is updated on the local node only.

Parameters

volgrp
Specifies a volume group. If this operand is not specified, the timestamps for all the volume groups on this node are updated.

Security

You must have root authority to run this command.

Exit Status

0
Indicates the successful completion of the command.

1
Indicates that the program was unable to read one or more timestamps.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

To update the timestamp associated with the virtual shared disk volume group vsdvg1 for just this node, enter:

```
vsdvgts vsdvg1
```

Location

/usr/lpp/vsd/bin/vsdvgts

w

The following AIX commands begin with the with the letter *w*.

w Command

Purpose

Prints a summary of current system activity.

Syntax

w [**-h**] [**-u**] [**-w**] [**-l** | **-s** [-X] [-@ [*WPAR*]][*User*]

Description

The **w** command prints a summary of the current activity on the system. The summary includes the following:

Item	Description
WPAR	Workload partition name.
User	Who is logged on.
tty	Name of the tty the user is on.
login@	Time of day the user logged on.
idle	Number of minutes since a program last attempted to read from the terminal. **Note:** The idle time is taken from the global terminal when you log into wpar using the **clogin** command.
JCPU	System unit time used by all processes and their children on that terminal.
PCPU	System unit time used by the currently active process.
What	Name and arguments of the current process.

The heading line of the summary shows the current time of day, how long the system has been up, the number of users logged into the system, and the load average. The load average is the number of runnable processes over the preceding 1-, 5-, 15-minute intervals.

The following examples show the different formats used for the login time field:

Item	Description
`10:25am`	The user logged in within the last 24 hours.
`Tue10am`	The user logged in between 24 hours and 7 days.
`12Mar91`	The user logged in more than 7 days ago.

If a user name is specified with the *User* parameter, the output is restricted to that user.

Flags

Item	Description
-@	Prints the System activity tagged with a workload partition name: • providing the **-@** option without a WPAR name indicates the global environment in addition to all WPARs active in the system, and the heading line indicates values for the global environment only • providing the **-@** option with a WPAR name indicates the activity, and the heading line indicates values for only that WPAR • providing **-@** Global indicates the activity, and the heading line indicates values for the global environment only. **Note:** Not providing the **-@** option indicates that the current WPAR or global environment, wherever the **w** command is running.
-h	Suppresses the heading.
-l	Prints the summary in long form. This is the default.
-s	Prints the summary in short form. In the short form, the tty is abbreviated, and the login time, system unit time, and command arguments are omitted.
-u	Prints the time of day, amount of time since last system startup, number of users logged on, and number of processes running. This is the default. Specifying the **-u** flag without specifying the **-w** or **-h** flag is equivalent to the uptime command.
-w	The equivalent of specifying the **-u** and **-l** flags, which is the default.
-X	Prints all available characters of each user name instead of truncating to the first 8 characters. The user name is also moved to the last column of the output.

Files

Item	Description
/etc/utmp	Contains the list of users.

wait Command

Purpose

Waits until the termination of a process ID.

Syntax

wait [*ProcessID* ...]

Description

The **wait** command waits (pauses execution) until the process ID specified by the *ProcessID* variable terminates. If the *ProcessID* variable is not specified, the **wait** command waits until all process IDs known to the invoking shell have terminated and exit with a 0 exit status. If a *ProcessID* variable represents an unknown process ID, the **wait** command treats them as known process IDs that exited with exit status 127. The **wait** command exits with the exitstatus of the last process ID specified by the *ProcessID* variable.

Flag

Item	Description
ProcessID	Specifies an unsigned decimal integer process ID of a command, which the **wait** command waits on until termination.

Exit Status

If one or more operands were specified, all of the operands terminated or were not known by the invoking shell, and the status of the last operand specified is known, then the exit status of the **wait** command is the same as the exit status information of the command indicated by the last operand specified. If the process terminated abnormally due to the receipt of a signal, then the exit status is greater than 128 and distinct from the exit status information generated by other signals, although the exact status value is unspecified (see the **kill -l** command option). Otherwise, the **wait** command exits with one of the following values:

Item	Description
0	The **wait** command was invoked with no operands and all process IDs known by the invoking shell have terminated.
1-126	The **wait** command detected an error.
127	The command identified by the last *ProcessID* operand specified is unknown.

File

Item	Description
/usr/bin/wait	Contains the **wait** command.

wall Command

Purpose

Writes a message to all users that are logged in.

Syntax

wall [**-a**] [**-g** *Group*][*Message*]

Description

The **wall** command writes a message to all users that are logged in. If the *Message* parameter is not specified, the **wall** command reads the message from standard input until it reaches an end-of-file character. The message is then sent to all logged in users. The following heading precedes the message:

```
Broadcast message from
user@node

 (tty) at hh:mm:ss ...
```

`hh:mm:ss` represents the hours, minutes, and seconds when the message was sent.

To override any protections set up by other users, you must operate with root user authority. Typically, the root user uses the **wall** command to warn all other users of an impending system shutdown.

Note:

- The **wall** command only sends messages to the local node.

- Messages can contain multibyte characters.

Flags

Item	Description
-a	Performs the default operation. This flag is provided for System V compatibility. It broadcast messages to the console and pseudo-terminals.
-g *Group*	Broadcasts to a specified group only.

Files

Item	Description
/dev/tty	Specifies a device.

wallevent Command

Purpose

Broadcasts an event or a rearm event to all users who are logged in.

Syntax

`wallevent [-c] [-h]`

Description

The `wallevent` script broadcasts a message on an event or a rearm event to all users who are currently logged in to the host when the event or the rearm event occurs. Event or rearm event information is captured and posted by the event response resource manager in environment variables that are generated by the event response resource manager when an event or a rearm event occurs. This script can be used as an action that is run by an event response resource. It can also be used as a template to create other user-defined actions. The language in which the messages of the `wallevent` script are returned depend on the locale settings.

Messages are displayed in this format at the consoles of all users who are logged in when an event or a rearm event occurs for which this script is a response action :

```
Broadcast message from user@host (tty) at hh:mm:ss...

severity event_type occurred for Condition condition_name
on the resource resource_name of resource_class_name at hh:mm:ss mm/dd/yy
The resource was monitored on node_name and resided on {node_names}.
```

Event information is returned about the ERRM environment variables, and also includes the following:

Local Time
Time when the event or rearm event is observed. The actual environment variable supplied by ERRM is ERRM_TIME. This value is localized and converted to readable form before being displayed.

This script captures the environment variable values and uses the `wall` command to write a message to the currently logged-in user consoles.

Flags

-c
Instructs `wallevent` to broadcast the ERRM_VALUE of an ERRM event. When the `-c` flag is specified, `wallevent` broadcasts the SNMP trap message.

-h
Writes the script's usage statement to standard output.

Parameters

log_file
Specifies the name of the file where event information is logged. An absolute path for the *log_file* parameter should be specified.

The *log_file* is treated as a circular log and has a fixed size of 64KB. When *log_file* is full, new entries are written over the oldest existing entries.

If *log_file* already exists, event information is appended to it. If *log_file* does not exist, it is created so that event information can be written to it.

Exit Status

0
Script has run successfully.

1
Error occurred when the script was run.

Restrictions

1. This script must be run on the node where the ERRM is running.
2. The `wall` command is used to write a message to currently logged-in user consoles. Refer to the `wall` man page for more information on the `wall` command.

Standard Output

When the `-h` flag is specified, the script's usage statement is written to standard output.

Examples

1. Suppose the `wallevent` script is a predefined action in the critical-notification response, which is associated with the `/var space used` condition on the resource `/var`. The threshold of the event expression defined for this condition is met, and an event occurs. The critical-notification response takes place, and `wallevent` is run. The following message is displayed on the consoles of all users who are logged in:

```
Broadcast message from joe@neverland.com (pts/6) at 18:42:03...

Critical event occurred for Condition /var space used
on the resource /var of filesys of IBM.FileSystem at 18:41:50 03/28/02
The resource was monitored on c174n05 and resided on {c174n05}.
```

2. When a rearm event occurs for the `/var space used` condition on the resource `/var`, the following message is displayed on the consoles of all users who are logged in:

```
Broadcast message from joe@neverland.com (pts/6) at 18:42:03...

Critical rearm event occurred for Condition /var space used
on the resource /var of filesys of IBM.FileSystem at 18:41:50 03/28/02
The resource was monitored on c174n05 and resided on {c174n05}.
```

Location

/opt/rsct/bin/wallevent

watch Command

Purpose

Observes a program that might be untrustworthy.

Syntax

watch [**-e** *Events]* [**-o** *File]* [**-X**] *Command* [Parameter ...]

Description

The **watch** command allows the root user or a member of the audit group to observe the actions of a program that are thought to be untrustworthy. The **watch** command starts the program you specify with the *Command* parameter, with or without any *Parameter* fields, and records all audit events or the audit events you specify with the **-e** flag.

The **watch** command observes all the processes that are created while the program runs, including any child process. The **watch** command continues until all processes exit, including the process it created, to observe all the events that occur.

The **watch** command formats the audit records and writes them to standard output or to a file you specify with the **-o** flag.

For the **watch** command to work, the auditing subsystem is not configured and enabled.

Flags

Item	Description
-e *Events*	Specifies the events to be audited. The *Events* parameter is a comma-separated list of audit events that are defined in the **/etc/security/audit/events** file. The default value is all events.
-o *File*	Specifies the path name of the output file. If the **-o** flag is not used, output is written to standard output.
-X	Prints long user names when used with other flags that display user names. The upper limit is determined by the `max_logname` object data manager (ODM) attribute in the predefined attribute (PdAt) and customized attributes (CuAt) object classes. If a user name is greater than the `max_logname` attribute, it is truncated to the number of characters as specified by the `max_logname` attribute, minus 1 character.

Security

Access Control: This command grants execute (x) access to the root user and members of the audit group. The **setuid** command is set for the root user. This setting allows access to other audit subsystem commands and files, and to the **trusted computing base** attribute.

Files Accessed:

Mode	File
r	**/dev/audit**
x	**/usr/sbin/auditstream**
x	**/usr/sbin/auditselect**
x	**/usr/sbin/auditpr**

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To watch all files opened by the **bar** command, enter:

   ```
   watch -e FILE_Open /usr/lpp/foo/bar -x
   ```

   This command opens the audit device and issues the **/usr/lpp/foo/bar** command. It then reads all records and selects and formats the files with the event type of FILE_Open.
2. To watch the installation of the xyzproduct program, that might be untrustworthy, enter:

   ```
   watch /usr/sbin/installp xyzproduct
   ```

   This command opens the audit device and issues the **/usr/sbin/installp** command. It then reads all records and formats them.

Files

Item	Description
/usr/sbin/watch	Contains the **watch** command.
/dev/audit	Specifies the audit device from which the audit records are read.

wc Command

Purpose

Counts the number of lines, words, bytes, or characters in a file.

Syntax

wc [**-c** | **-m**] [**-l**] [**-w**] [*File ...*]

wc **-k** [**-c**] [**-l**] [**-w**] [*File ...*]

Description

By default, the **wc** command counts the number of lines, words, and bytes in the files specified by the *File* parameter. The command writes the number of newline characters, words, and bytes to the standard output and keeps a total count for all named files.

When you use the *File* parameter, the **wc** command displays the file names as well as the requested counts. If you do not specify a file name for the *File* parameter, the **wc** command uses standard input.

The **wc** command is affected by the **LANG**, **LC_ALL**, **LC_CTYPE**, and **LC_MESSAGES** environment variables.

The **wc** command considers a word to be a string of characters of non-zero length which are delimited by a white space (for example SPACE , TAB).

Flags

Item	Description
-c	Counts bytes unless the **-k** flag is specified. If the **-k** flag is specified, the **wc** command counts characters.
-k	Counts characters. Specifying the **-k** flag is equivalent to specifying the **-klwc** flag. If you use the **-k** flag with other flags, then you must include the **-c** flag. Otherwise, the **-k** flag is ignored. For more information, see examples 4 and 5. **Note:** This flag is to be withdrawn in a future release.
-l	Counts lines.
-m	Counts characters. This flag cannot be used with the **-c** flag.
-w	Counts words. A word is defined as a string of characters delimited by spaces, tabs, or newline characters.

Note: If no flag is specified, **wc** by default counts the lines, words, bytes in a file or from standard input.

Exit Status

This command returns the following exit values:

Item	Description
0	The command ran successfully.
>0	An error occurred.

Examples

1. To display the line, word, and byte counts of a file, enter:

   ```
   wc chap1
   ```

   The **wc** command displays the number of lines, words, and bytes in the chap1 file.
2. To display only byte and word counts, enter:

   ```
   wc -cw chap*
   ```

   The **wc** command displays the number of bytes and words in each file that begins with chap. The command also displays the total number of bytes and words in these files.
3. To display the line, word, and character counts of a file, enter:

   ```
   wc -k chap1
   ```

   The **wc** command displays the number of lines, words, and characters in the chap1 file.
4. To display the word and character counts of a file, enter:

   ```
   wc -kcw chap1
   ```

   The **wc** command displays the number of characters and words in the chap1 file.
5. To use the **wc** command on standard input, enter:

   ```
   wc -klw
   ```

   The **wc** command displays the number of lines and words in standard input. The **-k** flag is ignored.
6. To display the character counts of a file, enter:

```
wc -m chap1
```

The **wc** command displays the number of characters in the chap1 file.

7. To use the **wc** command on standard input, enter:

```
wc -mlw
```

The **wc** command displays the number of lines, words, and characters in standard input.

Files

Item	Description
/usr/bin/wc, /bin/wc	Contains the **wc** command.
/usr/ucb/wc	Contains the symbolic link to the **wc** command.

what Command

Purpose

Displays identifying information in files.

Syntax

what [**-s**] *Pathname/File*.

Description

The **what** command searches specified files for all occurrences of the pattern that the **get** command substitutes for the **@(#)** keyletter (see the **get** or **prs** command for a description of identification keywords). By convention, the value substituted is "@(#)" (double quotation marks, at sign, left parenthesis, pound sign, right parenthesis, double quotation marks). If no file is specified, the **what** command reads from standard input.

The **what** command writes to standard output whatever follows the pattern, up to but not including the first double quotation mark ("), greater than symbol (>), new-line character, backslash (\), or null character.

The **what** command should be used in conjunction with the **get** command, which automatically inserts the identifying information. You can also use the **what** command on files where the information is inserted manually.

The **what** command accommodates the compiler inserted command line options in a binary file. The command line options saved in a binary file by AIX compilers may contain the backslash (\), the greater than symbol (>), or the double quotation mark ("), within the macro definitions. The **what** command behaves in the following manner to write the command line options saved by the compiler.

At the start of a line, if the pattern @(#) is followed by "opt" and is with or without a blank space (" ") in between, then the **what** command writes the character till "\n" that is the end of a line.

For example,

- For C and FORTRAN AIX compilers, use @(#) opt (...).
- For C++ AIX compilers, use @(#) opt (...).

The whole line is printed after "@(#)".

Note: The **what** command may fail to find SCCS identification strings in executable files.

Flags

Item	Description
-s	Searches for only the first occurrence of the **@(#)** pattern.

Exit Status

This command returns the following exit values:

Item	Description
0	Any matches were found.
1	Otherwise.

Examples

Suppose that the file `test.c` contains a C program that includes the line:

```
char ident[ ] = "@(#)Test Program";
```

If you compile `test.c` to produce `test.o`, then the command:

```
what test.c test.o
```

displays:

```
test.c:
Test Program
test.o:
Test Program
```

Note: The full file path names `usr/bin/test.c` and `user/bin/test.o` are required if the files are not in the current directory.

Files

Item	Description
/usr/bin/what	Contains the **what** command.

whatis Command

Purpose

Describes what function a command performs.

Syntax

whatis [**-M** *PathName*] *Command ...*

Description

The **whatis** command looks up a given command, system call, library function, or special file name, as specified by the *Command* parameter, from a database you create using the **catman -w** command. The **whatis** command displays the header line from the manual section. You can then issue the **man** command to obtain additional information.

The **whatis** command is equivalent to using the **man -f** command.

> **Note:** When the **/usr/share/man/whatis** database is built from the HTML library using the **catman -w** command, section 3 is equivalent to section 2 or 3. See the **man** command for further explanation of sections.

Flags

Item	Description
-M *PathName*	Specifies an alternative search path. The search path is specified by the *PathName* parameter, and is a colon-separated list of directories in which the **whatis** command expects to find the standard manual subdirectories.

Examples

To find out what the **ls** command does, enter:

```
whatis ls
```

This produces the following output:

```
ls(1)  -Displays the contents of a directory.
```

Files

Item	Description
/usr/share/man/whatis	Contains the **whatis** database.

whatnow Command

Purpose

Starts a prompting interface for draft disposition.

Syntax

whatnow [{ **-draftfolder** **+***folder* | **-nodraftfolder** | *file* } { **-draftmessage** *message* | *file* }]
[**-editor** *editor* | **-noedit**] [**-prompt** *string*]

Description

The **whatnow** command provides an interface for the disposition of messages. By default, the interface operates on the current draft message. When you enter the **whatnow** command, the system places you in the interface and returns the following prompt:

```
What now?
```

Within the interface you can manipulate message drafts using the **whatnow** subcommands. To see a listing of the subcommands, press the Enter key at the `What now?` prompt. To exit the interface, press q.

If you do not specify the **-draftfolder** flag or if the `Draft-Folder:` entry in the **$HOME/.mh_profile** file is undefined, the **whatnow** command searches your MH directory for a **draft** file. Specifying a message after the **-draftfolder +***folder* flag is the same as specifying the **-draftmessage** flag.

To change the default editor for the **whatnow** command, use the **-editor** flag or define the `Editor:` entry in the *UserMhDirectory***/.mh_profile** file.

Note: The **comp**, **dist**, **forw**, or **repl** commands use the same interface as the **whatnow** command.

Flags

Item	Description
-draftfolder +*folder*	Specifies the folder containing the message. By default, the system uses the *UserMhDirectory***/draft** file. Specifying a message after the **-draftfolder +***folder* is the same as using the **-draftmessage** flag.
-draftmessage *message*	Specifies the draft message.
-editor *editor*	Specifies that the value of the *editor* variable is the initial editor for composing or revising the message.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
file	User selected draft file.
message	Specifies the message. Use the following references to specify messages: ***Number*** Number of the message. **cur or . (period)** Current message. This is the default. **first** First message in a folder. **last** Last message in a folder. **next** Message following the current message. **prev** Message preceding the current message. **-nodraftfolder** Places the draft in the *UserMhDirectory***/draft** file. **-noedit** Suppresses the initial edit. **-prompt *string*** Uses the specified string as the prompt. The default string is `What now?`.

whatnow Subcommands

The **whatnow** subcommands enable you to edit the message, direct the disposition of the message, or end the processing of the **whatnow** command.

Item	Description
display [*flags*]	Displays the message being redistributed or replied to. You can specify any *flags* parameter that is valid for the listing program. (Use the `lproc:` entry in the **$HOME/.mh_profile** file to set a default listing program.) If you specify flags that are invalid for the listing program, the **whatnow** command does not pass the path name of the draft.
edit [*commandstring*]	Specifies with the *commandstring* parameter an editor for the message. You can specify the editor and any valid flags to that editor. If you do not specify an editor, the **whatnow** command uses the editor specified by the `Editor:` entry in your *UserMhDirectory***/.mh_profile** file. If your `Editor:` entry is undefined, the **whatnow** command starts the editor used in the previous editing session.
list [*flags*]	Displays the draft. You can specify any *flags* parameter that is valid for the listing program. (To specify a default listing program, set a default `lproc:` entry in the **$HOME/.mh_profile** file.) If you specify any flags that are invalid for the listing program, the **whatnow** command does not pass the path name of the draft.
push [*flags*]	Sends the message in the background. You can specify any valid flag for the **send** command.
quit [**-delete**]	Ends the **whatnow** session. If you specify the **-delete** flag, the **whatnow** command deletes the draft. Otherwise, the **whatnow** command stores the draft.
`refile` [*flags*] **+***folder*	Files the draft in the specified folder and supplies a new draft having the previously specified form. You can specify any *flags* parameter that is valid for the command serving as the **fileproc**. (You can set a default `fileproc:` entry in the **$HOME/.mh_profile** file.)
send [*flags*]	Sends the message. You can specify any valid flags for the **send** command.
whom [*flags*]	Displays the addresses to which the message would be sent. You can specify any valid flags for the **whom** command.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Draft-Folder:`	Sets the default folder for drafts.
`Editor:`	Sets the default editor.
`fileproc:`	Specifies the program used to refile messages.
`LastEditor-next:`	Specifies the editor used after exiting the editor specified by the *LastEditor* variable.
`lproc:`	Specifies the program used to list the contents of a message.
`Path:`	Specifies the *UserMhDirectory*.
`sendproc:`	Specifies the program used to send messages.

Item	Description
`whomproc:`	Specifies the program used to determine the users to whom a message would be sent.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the original message when you are replying to a message, enter the following at the `What now?` prompt:

   ```
   display
   ```

   The system displays the original message. If you enter the **display** subcommand from a command other than the **dist** or **repl** command, you will receive a system message stating that there is no alternate message to display.
2. To edit the draft message with the `vi` editor, enter the following at the `What now?` prompt:

   ```
   edit vi
   ```
3. To edit the draft message with the default editor specified in your **.mh_profile** file, enter the following at the `What now?` prompt:

   ```
   edit
   ```
4. To list the contents of the draft message you have composed, enter the following at the `What now?` prompt:

   ```
   list
   ```

   The draft message you are composing is displayed.
5. To send the draft message in the background and get a shell prompt immediately, enter the following at the `What now?` prompt:

   ```
   push
   ```

   The draft message is sent and you immediately receive the shell prompt.
6. To quit composing a draft message and save it to a file so that you can later finish composing the message, enter the following at the `What now?` prompt:

   ```
   quit
   ```

   The system responds with a message similar to the following.

   ```
   whatnow: draft left on /home/dale/Mail/draft
   ```

   In this example, user `dale`'s draft message is saved to the **/home/dale/Mail/draft** file.
7. To quit composing a draft message and delete the message, enter the following at the `What now?` prompt:

   ```
   quit -delete
   ```

   The shell prompt is displayed when the draft message is deleted.
8. To file the draft message you are composing before you send it, enter the following at the `What now?` prompt:

```
refile +tmp
```

The system responds with a message similar to the following:

```
Create folder "home/dale/Mail/tmp"?
```

In this example, if you answer `yes`, the draft message is filed in user `dale`'s folder `tmp`.

9. To send the draft message you have composed, enter the following at the `What now?` prompt:

```
send
```

The shell prompt is displayed when the message is sent.

10. To verify that all addresses in the draft message are recognized by the mail delivery system, enter the following at the `What now?` prompt:

```
whom
```

The system responds with a message similar to the following:

```
jeanne... User unknown
dale@venus... deliverable
```

In this example, the mail delivery system recognized `dale@venus` as a correct address, but did not recognize `jeanne` as a correct address.

Files

Item	Description
$HOME/.mh_profile	Specifies the MH user profile.
UserMhDirectory/**draft**	Contains the current message draft.
/usr/bin/whatnow	Contains the **whatnow** command.

whereis Command

Purpose

Locates source, binary, or manual for program.

Syntax

whereis [**-s**] [**-b**] [**-m**] [**-u**] [{ { **-S** | **-B** | **-M** } *Directory* ... }... **-f**] *File* ...

Description

The **whereis** command locates the source, binary, and manuals sections for specified files. The supplied names are first stripped of leading path name components and any (single) trailing extension of the form *.ext* (for example, **.c**). Prefixes of **s.** resulting from use of the Source Code Control System (see **SCCS**) are also dealt with. The command then attempts to find the desired program from a list of standard locations.

A usage message is returned if a bad option is entered. In other cases, no diagnostics are provided.

Flags

If any of the **-b, -s, -m** or **-u** flags are given, the **whereis** command searches only for binary, source, manual, or unusual sections respectively (or any two thereof).

Item	Description
-b	Searches for binary sections of a file.
-m	Searches for manual sections of a file.
-s	Searches for source sections of a file.
-u	Searches for unusual files. A file is said to be unusual if it does not have one entry of each requested type. Entering `whereis -m -u *` asks for those files in the current directory which have no documentation.

The **-B, -M,** and **-S** flags can be used to change or otherwise limit the places where the **whereis** command searches. Since the program uses the **chdir** subroutine to run faster, path names given with the **-M, -S** and **-B** flag directory list must be full; for example, they must begin with a **/** (slash).

Item	Description
-B	Like **-b**, but adds a directory to search. Change or limit the places where the **whereis** command searches for binaries.
-M	Like **-m**, but adds a directory to search. Change or limit the places where the **whereis** command searches for manual sections.
-S	Like **-s**, but adds a directory to search. Change or limit the places where the **whereis** command searches for sources
-f	Terminates the last **-M, -S** or **-B** directory list and signal the start of file names.

Examples

To find all of the files in the **/usr/ucb** directory that either are not documented in the **/usr/man/man1** directory or do not have source in the **/usr/src/cmd** directory, enter:

```
cd /usr/ucb
whereis -u -M /usr/man/man1 -S /usr/src/cmd -f *
```

Files

Item	Description
/usr/share/man/*	Directories containing manual files.
/sbin, **/etc**, **/usr/**{**lib**,**bin**,**ucb**,**lpp**}	Directories containing binary files.
/usr/src/*	Directories containing source code files.

which Command

Purpose

Locates a program file, including aliases and paths.

Syntax

which [*Name* ...]

Description

The **which** command takes a list of program names and looks for the files that run when these names are given as commands. The **which** command expands each argument, if it is aliased, and searches for it along the user's path. The aliases and paths are taken from the **.cshrc** file in the user's home directory. If the **.cshrc** file does not exist, or if the path is not defined in the **.cshrc** file, the **which** command uses the path defined in the user's environment.

A diagnostic is given if a name is aliased to more than a single word or if an executable file with the argument name is not found in the path.

In the Korn shell, you can use the **whence** command to produce a more verbose report.

Examples

To find the executable file associated with a command name of `lookup`:

```
which lookup
```

Files

Item	Description
$HOME/.cshrc	Contains the source of aliases and path values.

which_fileset Command

Purpose

Searches the **/usr/lpp/bos/AIX_file_list** file for a specified file name or command.

Syntax

which_fileset [*File*]

Description

The **which_fileset** command searches the **/usr/lpp/bos/AIX_file_list** file for a specified file name or command name, and prints out the name of the fileset that the file or command is shipped in.

The **/usr/lpp/bos/AIX_file_list** file is large and not installed automatically. You must install the **bos.content_list** fileset to receive this file.

The *File* parameter can be the command name, the full path name, or a regular expression search pattern.

Examples

1. To display which fileset the dbx command is shipped in, enter:

   ```
   which_fileset dbx
   ```

   The screen displays the following:

   ```
   /usr/bin/dbx > /usr/ccs/bin/dbx             bos.adt.debug 4.2.1.0
   /usr/ccs/bin/dbx                            bos.adt.debug 4.2.1.0
   ```

2. To display all commands and paths containing the *sendmail* string, enter:

   ```
   which_fileset sendmail.*
   ```

   The screen displays the following:

```
/usr/ucb/mailq > /usr/sbin/sendmail    bos.compat.links 4.2.0.0
/usr/ucb/newaliases > /usr/sbin/sendmail bos.compat.links 4.2.0.0
/usr/lib/nls/msg/Ca_ES/sendmail87.cat  bos.msg.Ca_Es.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/ca_ES/sendmail87.cat  bos.msg.ca_Es.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/cs_CZ/sendmail87.cat  bos.msg.cs_CZ.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/De_DE/sendmail87.cat  bos.msg.De_DE.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/de_DE/sendmail87.cat  bos.msg.de_DE.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/En_US/sendmail87.cat  bos.msg.En_US.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/en_US/sendmail87.cat  bos.msg.en_US.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/Es_ES/sendmail87.cat  bos.msg.Es_ES.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/es_ES/sendmail87.cat  bos.msg.es_ES.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/Fr_FR/sendmail87.cat  bos.msg.Fr_FR.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/fr_FR/sendmail87.cat  bos.msg.fr_FR.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/hu_HU/sendmail87.cat  bos.msg.hu_HU.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/It_IT/sendmail87.cat  bos.msg.It_IT.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/it_IT/sendmail87.cat  bos.msg.it_IT.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/Ja_JP/sendmail87.cat  bos.msg.Ja_JP.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/ja_JP/sendmail87.cat  bos.msg.ja_JP.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/ko_KR/sendmail87.cat  bos.msg.ko_KR.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/pl_PL/sendmail87.cat  bos.msg.pl_PL.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/ru_RU/sendmail87.cat  bos.msg.ru_RU.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/Sv_SE/sendmail87.cat  bos.msg.Sv_SE.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/sv_SE/sendmail87.cat  bos.msg.sv_SE.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/ZH_CN/sendmail87.cat  bos.msg.ZH_CN.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/zh_CN/sendmail87.cat  bos.msg.zh_CN.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/Zh_TW/sendmail87.cat  bos.msg.Zh_TW.net.tcp.client 4.2.0.0
/usr/lib/nls/msg/zh_TW/sendmail87.cat  bos.msg.zh_TW.net.tcp.client 4.2.0.0
/etc/sendmail.cf                       bos.net.tcp.client.4.2.1.0
/usr/lib/sendmail > /usr/sbin/sendmail bos.net.tcp.client.4.2.1.0
/usr/sbin/mailq > /usr/sbin/sendmail   bos.net.tcp.client.4.2.1.0
/usr/sbin/newaliases > /usr/sbin/sendmail bos.net.tcp.client.4.2.1.0
/usr/sbin/sendmail                     bos.net.tcp.client.4.2.1.0
```

3. To find where the **/usr/sbin/which_fileset** command is shipped, enter:

```
which_fileset /usr/bin/which_fileset
```

The screen displays:

```
/usr/sbin/which_fileset      bos.rte.install 4.2.1.0
```

who Command

Purpose

Identifies the users currently logged in.

Syntax

who [**-a** | **-b** **-d** **-i** **-l** **-m** **-p** **-q** **-r** **-s** **-t** **-u** **-w** **-A** **-H** **-T** **-X**] [*File*]

who am { **i** | **I** }

Description

The **who** command displays information about all users currently on the local system. The following information is displayed: login name, tty, date and time of login. Typing `who am i` or `who am I` displays your login name, tty, date and time you logged in. If the user is logged in from a remote machine, then the host name of that machine is displayed as well.

The **who** command can also display the elapsed time since line activity occurred, the process ID of the command interpreter (shell), logins, logoffs, restarts, and changes to the system clock, as well as other processes generated by the initialization process.

The general output format of the **who** command is as follows:

```
Name [State] Line Time [Activity] [Pid] [Exit] (Hostname)
```

where:

Item	Description
`Name`	Identifies the user's login name.
`State`	Indicates whether the line is writable by everyone (see the **-T** flag).
`Line`	Identifies the line name as found in the **/dev** directory.
`Time`	Represents the time when the user logged in.
`Activity`	Represents the hours and minutes since activity last occurred on that user's line. A . (dot) here indicates line activity within the last minute. If the line has been quiet more than 24 hours or has not been used since the last system startup, the entry is marked as old.
`Pid`	Identifies the process ID of the user's login shell.
`Term`	Identifies the process termination status (see the **-d** flag). For more information on the termination values, refer to the **wait** subroutine or to the **/usr/include/sys/signal.h** file.
`Exit`	Identifies the exit status of ended processes (see the **-d** flag).
`Hostname`	Indicates the name of the machine the user is logged in from.

To obtain information, the **who** command usually examines the **/etc/utmp** file. If you specify another file with the *File* parameter, the **who** command examines that file instead. This new file is usually the **/var/adm/wtmp** or **/etc/security/failedlogin** file.

If the *File* parameter specifies more than one file name, only the last file name will be used.

Note: This command only identifies users on the local node.

Flags

Item	Description
-a	Processes the **/etc/utmp** file or the named file with all information. Equivalent to specifying the **-bdlprtTu** flags.
-b	Indicates the most recent system startup time and date.
-d	Displays all processes that have expired without being regenerated by init. The exit field appears for dead processes and contains the termination and exit values (as returned by wait) of the dead process. (This flag is useful for determining why a process ended by looking at the error number returned by the application.)
-l	Lists any login process.
-m	Displays information about the current terminal only. The **who -m** command is equivalent to the **who am i** and **who am I** commands.
-p	Lists any active process that is currently active and has been previously generated by init.
-q	Prints a quick listing of users and the number of users on the local system.
-r	Indicates the current run-level of the process.
-s	Lists only the name, line, and time fields. This flag is the default; thus, the **who** and **who -s** commands are equivalent.
-t	Indicates the last change to the system clock by the root user using the **date** command. If the **date** command has not been run since system installation, the **who -t** command produces no output.
-u or **-i**	Displays the user name, tty, login time, line activity, and process ID of each current user.
-A	Displays all accounting entries in the **/etc/utmp** file. These entries are generated through the **acctwtmp** command.

Item	Description
-H	Displays a header (title).
-T or **-w**	Displays the state of the tty and indicates who can write to that tty as follows: **+** Writable by anyone. **-** Writable only by the root user or its owner. **?** Bad line encountered.
-X	Prints all available characters of each user name instead of truncating to the first 8 characters. The user name is also moved to the last column of the output.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. 1. To display information about who is using the local system node, type:

   ```
   who
   ```

   Information similar to the following is displayed:

   ```
   pts/1      Nov  9 00:20   long_username_greater_than_eight_characters  (localhost)
   ```

2. To display your user name, type:

   ```
   who am i
   ```

   Information similar to the following is displayed:

   ```
   george lft/0 Jun 8 08:34
   ```

3. To display a history of logins, logouts, system startups, and system shutdowns, type:

   ```
   who /var/adm/wtmp
   ```

   Information similar to the following is displayed:

   ```
   hank   lft/0   Jun   8   08:34  (ausnix5)
   john   lft/0   Jun   8   08:34  (JIKey)
   mary   lft/0   Jun   8   08:22  (machine.austin.ibm)
   jan    pts4    Jun   8   09:19  (puff.wisc.edu)
   ```

4. To display the run-level of the local system node, type:

   who -r

   Information similar to the following is displayed:

   ```
   . run-level 2 Jun 8 04:15 2 0 s
   ```

5. To display any active process that is currently actively and has been previously generated by init, type:

   ```
   who  -p
   ```

   Information similar to the following is displayed:

   ```
   srcmstr   .    Jun 8  04:15  old  2896
   cron      .    Jun 8  04:15  old  4809
   uprintfd  .    Jun 8  04:15  old  5158
   ```

6. To process the **/var/adm/wtmp** file with the **-bdlprtTu** flags specified, type:

   ```
   who  -a /var/adm/wtmp
   ```

   Information similar to the following is displayed:

   ```
      .        system boot Jun 19 10:13
      .        run-level 2 Jun 19 10:13
      .            .       Jun 19 10:14    old
      .            .       Jun 19 10:14    old
      .            .       Jun 19 10:14    old
   rc         -     .       Jun 19 10:13    old
      .            .       Jun 19 10:16    old
      .            .       Jun 19 10:14    old
   srcmstr    -     .       Jun 19 10:14    old
   rctcpip    -     .       Jun 19 10:14    old
   rcdce      -     .       Jun 19 10:14    old
   rccm       -     .       Jun 19 10:15    old
   dceupdt    -     .       Jun 19 10:15    old
   rcnfs      -     .       Jun 19 10:15    old
   cron       -     .       Jun 19 10:16    old
   piobe      -     .       Jun 19 10:16    old
   qdaemon    -     .       Jun 19 10:16    old
   writesrv   -     .       Jun 19 10:16    old
   uprintfd   -     .       Jun 19 10:16    old
      .            .       Jun 19 10:16    old
   LOGIN      - lft0        Jun 19 10:16    old
      .            .       Jun 19 10:16    old
      .            .       Jun 19 10:16    old
   ```

Files

Item	Description
/etc/utmp	Contains user and accounting information.
/etc/security/failedlogin	Contains the history of all invalid logins.
/var/adm/wtmp	Contains the history of all logins since the file was last created.
/usr/include/sys/signal.h	Contains a list of termination values.

whoami Command

Purpose

Displays your login name.

Syntax

whoami

Description

The **whoami** command displays your login name. Unlike using the command **who** and specifying **am i**, the **whoami** command also works when you have root authority since it does not examine the **/etc/utmp** file.

Files

Item	Description
/etc/passwd	Contains user IDs.

whodo Command

Purpose

Lists the jobs being performed by users on the system.

Syntax

whodo [**-h**] [**-l**] [-X] [*User*]

Description

Prints information on all processes for a terminal, as well as the child processes.

By default, the output generated by the command for each active logged user will contain name of the terminal, user ID, date login time. The output is headed by the date, time and machine name. This information is followed by a record of active processes associated with that user ID. Each record shows the terminal name, process-ID, CPU minutes and seconds used, and process name.

Flags

Item	Description
-h	Suppress the heading that is printed on the output.
-l	Produce a long form of output. A summary of the current activity on the system is printed. The summary includes the following: **User** Who is logged on. **tty** Name of the tty the user is on. **login@** Time of day the user logged on. **idle** Number of minutes since a program last attempted to read from the terminal. **JCPU** System unit time used by all processes and their children on that terminal. **PCPU** System unit time used by the currently active process. **what** Name and parameters of the current process. The heading line of the summary shows the current time of day, how long the system has been up, the number of users logged into the system.
-X	Prints all available characters of each user name instead of truncating to the first 8 characters. The user name is also moved to the last column of the output.

Parameters

Item	Description
User	Limits output to all sessions pertaining to the user specified with *User*. More than one user name cannot be specified at a time.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Examples

1. When the **whodo** command is invoked on host "linguist" without any flags or parameters, the output looks similar to the following:

```
Sun Jul 28 16:27:12 2002
linguist

lft0    jeffg    8:15
         ?    4136     0:00    dtlogin
         ?    3408     4:55    dtsession
         ?    2072     4:37    dtwm
         ?    17310     0:00    dtexec
         ?    20904     5:53    dtterm
     pts/0    22454     0:00    ksh
     pts/0    4360     0:07    ksh
     pts/0    25788     0:00    whodo
         ?    23672     0:00    dtexec
         ?    27536     0:00    dtterm
     pts/3    21508     0:00    ksh
         ?    23888     0:00    dtexec
         ?    24384     2:49    dtterm
     pts/2    24616     0:00    ksh
     pts/2    25002     0:04    ksh
     pts/2    26110     0:00    ksh
         ?    25276     0:00    dtexec
         ?    27090     0:31    dtterm
     pts/1    24232     0:00    ksh
     pts/1    23316     0:01    ksh
         ?    12566     4:23    dtfile
         ?    21458     1:35    dtfile

pts/0    jeffg    8:16
     pts/0    22454     0:00    ksh
     pts/0    4360     0:07    ksh
     pts/0    25788     0:00    whodo

pts/1    jeffg    17:8
     pts/1    24232     0:00    ksh
     pts/1    23316     0:01    ksh

pts/2    jeffg    17:20
     pts/2    24616     0:00    ksh
     pts/2    25002     0:04    ksh
     pts/2    26110     0:00    ksh

pts/3    root    16:26
     pts/3    21508     0:00    ksh
```

2. The command whodo -l on the host linguist produces the following output:

```
  04:33PM   up 20 day(s), 22 hr(s), 51 mins(s)  5 user(s)
User     tty          login@       idle      JCPU      PCPU what
jeffg    lft0        08Jul02   21day(s)                   /usr/sbin/getty /de
jeffg    pts/0       08Jul02                14:00         7 whodo -l
jeffg    pts/1       16Jul02   10day(s)        44         9 /usr/bin/ksh
jeffg    pts/2       12Jul02         11      8:39         4 /usr/bin/ksh
root     pts/3       04:26PM          7                     -ksh
```

3. The command whodo -lX on the host kq11 produces the following output:

```
  12:50AM   up 3 day(s),  1 hr(s), 41 mins(s)  4 user(s)
 tty          login@       idle      JCPU      PCPU what            User
 tty0        Wed11PM    2day(s)                    -ksh            root
 pts/0       12:12AM                               tn 0            root
 pts/1       12:20AM                               whodo -lX
long_username_greater_than_eight_characters
 pts/2       Fri05AM    2day(s)                    -ksh            root
```

Files

Item	Description
/usr/sbin/whodo	Contains the **whodo** command.
/etc/utmp	Contains the list of users.

whois Command

Purpose

Identifies a user by user ID or alias.

Syntax

whois [**-h** *HostName*] [**.** | **!**] [*****] *Name* [**...**]

whois ?

Description

The **/usr/bin/whois** command searches a user name directory and displays information about the user ID or nickname specified in the *Name* parameter. The **whois** command tries to reach ARPANET host `internic.net` where it examines a user-name database to obtain information. The **whois** command should be used only by users on ARPANET. Refer to RFC 812 for more complete information and recent changes to the **whois** command.

> **Note:** If your network is on a national network, such as ARPANET, the host name is hard-coded as `internic.net`.

The *Name* [**...**] parameter represents the user ID, host name, network address, or nickname on which to perform a directory search. The **whois** command performs a wildcard search for any name that matches the string preceding the optional ... (three periods).

Flags

Item	Description
.	Forces a name-only search for the name specified in the *Name* parameter.
!	Displays help information for the nickname or handle ID specified in the *Name* parameter.
*****	Displays the entire membership list of a group or organization. If there are many members, this can take some time.
?	Requests help from the ARPANET host.
-h *HostName*	Specifies an alternative host name. The default host name on the ARPANET is `internic.net`. You can contact the other major ARPANET user-name database, `nic.ddn.mil`, by specifying the **-h** *HostName* flag.

Examples

1. To display information about ARPANET registered users by the name of Smith, enter:

   ```
   whois Smith
   ```

2. To display information about ARPANET registered users that use the handle Hobo, enter:

   ```
   whois  !Hobo
   ```

3. To display information about ARPANET registered users with the name of John Smith, enter:

   ```
   whois  .Smith, John
   ```

4. To display information about ARPANET registered users whose names or handles begin with the letters HEN, enter:

   ```
   whois HEN ...
   ```

5. To get help information for the **whois** command, enter:

   ```
   whois  ?
   ```

whom Command

Purpose

Manipulates Message Handler (MH) addresses.

Syntax

whom [**-alias** *File* ...] [**-nocheck** | **-check**] [{ **-draftfolder** **+***Folder* | **-nodraftholder** | *File* }
{ **-draftmessage** *Message* | **-draft***File* }]

Description

The **whom** command does the following:

- Expands the headers of a message into a set of addresses.
- Lists the addresses of the proposed recipients of a message.
- Verifies that the addresses are deliverable to the transport service.

> **Note:** The **whom** command does not guarantee that addresses listed as being deliverable will actually be delivered.

A message can reside in a draft folder or in a file. To specify where a message resides, use the **-draft**, **-draftfolder**, or **-draftmessage** flag.

If you do not specify the **-draftfolder** flag or if the `Draft-Folder:` entry in the **$HOME/.mh_profile** file is undefined, the **whom** command searches your MH directory for a **draft** file. Specifying a message after the **-draftfolder** **+***Folder* flag is the same as specifying the **-draftmessage** flag.

Flags

Item	Description
-alias *File*	Specifies a file to search for mail aliases. By default, the system searches the **/etc/mh/MailAliases** file.
-draft	Uses the header information in the *UserMhDirectory***/draft** file if it exists.

Item	Description
-draftfolder +*Folder*	Uses the header information from the draft message in the specified folder. If you specify a draft folder that doesn't exist, the system creates one for you.
-draftmessage *Message*	Uses the header information from the specified draft message.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
Message	Specifies the message draft. Use the following to specify messages: ***Number*** Number of the message. **cur or . (period)** Current message. This is the default. **first** First message in a folder. **last** Last message in a folder. **next** Message following the current message. **prev** Message preceding the current message.
-nodraftfolder	Undoes the last occurrence of the **-draftfolder +***Folder* flag.

Note: Two other flags, **-check** and **-nocheck**, are also available. These flags have no effect on how the **whom** command performs verification. The **-check** and **-nocheck** flags are provided for compatibility only.

Profile Entries

The following entries are entered in the *UserMhDirectory***/.mh_profile** file:

Item	Description
`Draft-Folder:`	Sets your default folder for drafts.
`postproc:`	Specifies the program used to post messages.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To list and verify the addresses of the proposed recipients of a message, enter the addressees and subject of the message at the respective prompt, as follows:

```
To: d77@nostromo
Subject: a test
```

When prompted again, enter the text of the message:

```
-------Enter initial text
test
-------
```

After the whatnow prompt, enter the **whom** command:

```
whatnow>>> whom
```

The address of the proposed recipients of the message is then displayed:

```
lance...
d77@nostromo... deliverable
```

Files

Item	Description
$HOME/.mh_profile	Specifies the MH user profile.
/usr/bin/whom	Contains the **whom** command.

wlmassign command

Purpose

Manually assigns processes to a workload management class or cancels prior manual assignments for processes.

Syntax

wlmassign [**-s** | **-S**] [**-u** | *Class_Name*] [*pid_list*] [**-g** *pgid_list*]

wlmassign [**-t** { *tag* [**-i** *inheritance*] | **-r** } [*pid_list*] [**-g** *pgid_list*]

Description

The **wlmassign** command:

- Assigns a set of processes specified by a list of process identifiers (*PID*) or process group identifiers (*PGIDs*) to a specified superclass or subclass or both, thus overriding the automatic class assignment or a prior manual assignment.
- Cancels a previous manual assignment for the processes specified in *pid_list* or *pgid_list*.
- Assigns the Workload Manager (WLM) tag process attribute to a set of processes specified by a list of *pids* or *pgids*.
- Removes the WLM tag process attribute from a set of processes specified by a list of *pids* or *pgids*.

In addition to the tag, additional inheritance suboptions can be specified and these suboptions indicates to the WLM if a child process must inherit the tag from its parent after **fork** or **exec** subroutine.

The process requires at least the **SIGPRIV** privileges or higher for tagging another process.

The WLM tag assignment remains in effect until one or more of the following conditions are true:

- The tag is removed by using the **-r** flag.
- The tagged process is ended.
- The tag is overwritten with a new tag.

When a WLM tag is assigned to a process that belongs to a class that has the inheritance property turned off, it is automatically reclassified according to the current assignment rules and the new tag is taken into account during reclassification. The WLM tag is effective if class inheritance attribute is not specified for the current process class. To override the class inheritance attribute in favor of reclassification based on tag rules, the **/usr/samples/kernel/wlmtune** command available in the **bos.adt.samples** PTF, can be used to modify the behavior of WLM. The related tunables follow:

tag_override_super
Indicates to the WLM that superclass inheritance is bypassed in favor of the rule-based classification if a rule matches the process tag. The default value is **0**.

tag_override_sub
Indicates to the WLM that subclass inheritance is bypassed in favor of the rule-based classification if a rule matches the process tag. The default value is **0**.

For the tag assignment rules-based classification to be effective, the tunable values must be set prior to a WLM reclassification update.

The interactions between automatic assignment (inheritance and rules), inheritance, and manual assignment are detailed in the Workload management in *Operating system and device management*.

The **wlmassign** command allows to specify processes using a list of PIDs, a list of PGDIDs, or both. The formats of these lists follow:

```
pid[,pid[,pid[...]]]
```

```
pgid[,pgid[,pgid[...]]]
```

The name of a valid superclass or subclass must be specified to manually assign the target processes to a class. If the target class is a superclass, each process is assigned to one of the subclasses of the specified superclass according to the assignment rules for the subclasses of this superclass.

A manual assignment remains in effect (and a process remains in its manually assigned class) until:

- The process terminates
- Workload Management (WLM) is stopped. When WLM is restarted, the manual assignments in effect when WLM was stopped are lost.
- The class the process has been assigned to is deleted
- A new manual assignment overrides a prior one.
- The manual assignment for the process is canceled using the **-u** flag.
- The process calls the **exec()** routine.

The name of a valid superclass or subclass must be specified to manually assign the target processes to a class. The assignment can be done or canceled at the superclass level, the subclass level or both. When a manual assignment is canceled for a process, or the process calls **exec()**, the process is then subject to automatic classification; if inheritance is enabled for the class that the process is in, it will remain in that class, otherwise the process will be reclassified according to the assignment rules.

For a manual assignment:

- If the *Class_Name* is the name of a superclass, the processes in the list are assigned to the superclass. The subclass is then determined, for each process, using the assignment rules for the subclasses of the target superclass.
- If the class name is a subclass name (*supername.subname*), the processes by default are assigned to both the superclass and the subclass. The processes can be assigned to the superclass only by specifying the **-S** flag or the subclass only by specifying the **-s** flag.

```
wlmassign   super1.sub2  -S  pid1
```

is equivalent to:

```
wlmassign super1 pid1
```

To assign a process to a class or cancel a prior manual assignment, the user must have authority both on the process and on the target class. These constraints translate into the following:

- The root user can assign any process to any class.
- A user with administration privileges on the subclasses of a given superclass (that is, the user or group name matches the user or group names specified in the attributes **adminuser** and **admingroup** of the superclass) can manually reassign any process from one of the subclasses of this superclass to another subclass of the superclass.
- Users can manually assign their own processes (same real or effective user ID) to a class, for which they have manual assignment privileges (that is, the user or group name matches the user or group names specified in the attributes **authuser** and **authgroup** of the superclass or subclass).

This defines 3 levels of privilege among the persons who can manually assign processes to classes, root being the highest. For a user to modify or terminate a manual assignment, they must have at least the same level of privilege as the person who issued the last manual assignment.

Note: The **wlmassign** command works with currently loaded WLM configuration. If the current configuration is a set, and the assignment is made to a class which does not exist in all configurations in the set, the assignment will be lost when a configuration that does not contain the class becomes active (class is deleted).

Flags

Item	Description
-g *pgid_list*	Indicates that the following list is a list of PGID.
-S	Specifies that the assignment is to be done or canceled at the superclass level only. This flag is used with a subclass name of the form *supername***.***subname*.
-s	Specifies that the assignment is to be done or canceled at the subclass level only. This flag is used with a subclass name of the form *supername***.***subname*.
-u	Cancel any manual assignment in effect for the processes in the **pid_list** or the **pgid_list**. If none of the **-s** or **-S** flags are used, this cancels the manual assignments for both the superclass and the subclass level.
-r	Removes a WLM tag from the specified process or the process group list.
-t *tag*	Sets a WLM tag for the specified process or the process group list.
-i *inheritance*	Specifies one or both tag inheritance sub-options in a comma-separated list. The following tag inheritance sub-options can be specified: **fork** Specifies that the children of this process should inherit the parent tag across fork. **exec** Specifies that the process retains its tag after a call to exec.

wlmcheck command

Purpose

Check automatic assignment rules and/or determines the Workload Manager class a process with a specified set of attributes would be classified in.

Syntax

wlmcheck [**-d** *Config*] [**-a** *Attributes*] [**-q**]

Description

The **wlmcheck** command with no arguments, gives the status of Workload Manager (WLM) and makes some coherency checks:

- Displays the current status of WLM (running/non running, active/passive, rsets bindings active, total limits enabled).
- Displays the status files that report the last loading errors, if any. If 'current' configuration is a set, this applies to all configurations in the set, and messages logged by the WLM daemon are reported.
- Checks the coherency of the attributes and assignment rules file(s) (such as, the existence of the classes, validity of user and group names, existence of application file names, etc).

If the **-d** *Config* flag is not specified, the checks are performed on the 'current' configuration.

The **wlmcheck** command can apply to a configuration set. In this case, the checks mentioned above are performed on all configurations of the set, after checking the set itself. Superclass names are reported in the form 'config/superclass' to indicate the regular configuration which they belong to.

Specifying a configuration with **-d** *Config* performs the checks on the *Config* configuration or set instead of 'current'. This does not change the reporting of status files and of the WLM daemon log, which only applies to the active configuration.

With the **-a** flag, **wlmcheck** displays the class that the process with attributes specified by *Attributes* would be assigned to, according to the rules for the current or specified configuration or configuration set. The format of the *Attributes* string is similar to an entry in the *rules* file, with the following differences:

- The class field is omitted (it is actually an output of **wlmcheck**)
- Each field can have at most one value. Exclusion (!), attribute groupings ($), comma separated lists, and wild cards are not allowed. For the *type* field, the AND operator "+" is allowed, since a process can have several of the possible values for the type attribute at the same time. For instance a process can be a 32 bit process and call plock, or be a 64 bit fixed priority process.
- At least one field must be specified (have a value different from a hyphen (-).

In addition, the first 2 fields are mandatory. The other fields, if not present default to a hyphen (-) which mean that any value in the corresponding field of an assignment rule is a match. When one or more of the fields in the attribute string are either not present or specified as a hyphen (-), the string is likely to match more than one rule. In this case, **wlmcheck** displays all the classes corresponding to all the possible matches.

Example of valid attribute strings:

```
$ wlmcheck -a "- root system /usr/lib/frame/framemaker - -"
$ wlmcheck -a "- - staff - 32bit+fixed"
$ wlmcheck -a "- bob"
```

Flags

Item	Description
-a *Attributes*	Passes a set of values for the classification attributes of the process in order to determine which class the process would be put into. This is a way to check that the assignment rules are correct and classify processes as expected.
-d *Config*	Uses the WLM property files in **/etc/wlm/Config** (which may indicate a set of time-based configurations) instead of **/etc/wlm/current**.
-q	Suppresses the output of the status of the latest activation/update of WLM and of messages logged by the WLM daemon (quiet mode).

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
classes	Contains the names and definitions of the classes.
limits	Contains the resource limits enforced on the classes.
rules	Contains the automatic assignment rules.
shares	Contains the resource shares allocated to the classes.

wlmcntrl Command

Purpose

Starts or stops the Workload Manager.

Syntax

wlmcntrl [[**-a** | **-c** | **-p**] [**-T** [**class** | **proc**] [**-g**] [**-d** *Config_Dir*] [**-o** | **-q**]

wlmcntrl -u [**-S** *Superclass* | **-d** *Config_Dir*]

Description

The **wlmcntrl** command stops, starts, updates or queries the state of Workload Manager (WLM). When starting or updating WLM, the WLM property files for the target configuration are pre-processed, and the data is loaded into the kernel. WLM can be started in two different modes:

- An active mode where WLM monitors and regulates the processor, memory and disk I/O utilization of the processes in the various classes.
- A passive mode where WLM only monitors the resource utilization without interfering with the standard operating system resource allocation mechanisms.

The active mode is the usual operating mode of WLM.

The classes, their limits and shares are described respectively in the **classes**, **limits**, and **shares** files. The automatic assignment rules are taken from the **rules** file. The class properties files for the superclasses of the WLM configuration **Config** are located in the subdirectory **/etc/wlm/Config**. The class properties files for the subclasses of the superclass Super of the configuration Config are located in **/etc/wlm/Config/Super**. The standard configuration shipped with the operating system is in **/etc/wlm/standard**. The current configuration is the one in the directory pointed to by the symbolic link **/etc/wlm/current**.

When the **-d** *Config_dir* flag is not used, **wlmcntrl** uses the configuration files in the directory pointed to by the symbolic link **/etc/wlm/current**.

When the **-d** *Config_dir* flag is used, **wlmcntrl** uses the configuration files in **/etc/wlm/Config_dir** and updates the **/etc/wlm/current** symbolic link to point to **/etc/wlm/Config_dir**, making **/etc/wlm/Config_dir** the current configuration. This is the recommended way to make **/etc/wlm/Config_dir** the current configuration.

When updating WLM using the **-u** flag, an empty string can be passed as *Config_dir* with the **-d** flag:

```
wlmcntrl -u -d ""
```

will simply refresh (reload) the assignment rules of the current configuration into the kernel without reloading the class definitions. This can be useful when a prior activation of WLM detected that some application files could not be accessed. After the system sdministrator has fixed the problems with either the rules or the files, this command can be used to reload only the rules.

The WLM configuration **Config** may also be a set of time-based configurations, in which case the subdirectory **/etc/wlm/Config** does not contain the properties files, but a list of configurations and the times of the week when they apply. The properties files are still in the subdirectory of each regular configuration of the set. When WLM is started or updated which such a set, a daemon is responsible for switching regular configurations of the set when the applicable one changes.

Note: This command is not supported when executed within a workload partition.

Flags

Item	Description
-a	Starts WLM in active mode or switches from passive to active mode. This is the default when no flag other than **-d**, **-g**, or **-T** is specified.
-c	Starts WLM in processor-only mode or switches from any mode to processor-only mode. In this mode, the WLM accounts for all resources, but only processor resource is regulated.
-d *Config_dir*	Uses **/etc/wlm/Config_dir** as an alternate directory for the WLM configuration (containing the classes, limits, shares and rules files) or configuration set (containing the list of configurations and the time tanges when they apply). This makes **/etc/wlm/Config_dir** the current configuration. This flag is effective when starting the WLM in active, processor-only or passive mode, or when updating the WLM. This flag cannot be used in conjunction with the **-o** and **-q** flags or when switching from a mode (among active, processor-only and passive) to another.
-g	Instructs WLM to ignore any potential resource set bindings. This means that all classes have access to the whole resource set of the system, regardless of whether or not they use a restricted resource set.
-o	Stops Workload Manager.
-p	Start WLM in passive mode or switches from any mode to passive mode. In this mode, the WLM accounts for all resources, but no resource is regulated.

Item	Description
-q	Queries the current state of WLM. Returns: **0** WLM is running in active mode. **1** WLM is not started. **2** WLM is running in passive mode. **3** WLM is running in active mode with no rset bindings. **4** WLM is running in passive mode with no rset bindings. **5** WLM is running in active mode for processor only **6** WLM is running in active mode for processor only with no rset bindings. **16** WLM is running in active mode, process total accounting is off. **18** WLM is running in passive mode, process total accounting is off. **19** WLM is running in active mode with no rset bindings, process total accounting is off. **20** WLM is running in passive mode with no rset bindings, process total accounting is off. **21** WLM is running in active mode for processor only, process total accounting is off. **22** WLM is running in active mode for processor only with no rset bindings, process total accounting is off. **32** WLM is running in active mode, class total accounting is off. **34** WLM is running in passive mode, class total accounting is off. **35** WLM is running in active mode with no rset bindings, class total accounting is off. **36** WLM is running in passive mode with no rset bindings, class total accounting is off. **37** WLM is running in active mode for processor only, class total accounting is off. **38** WLM is running in active mode for processor only with no rset bindings, class total accounting is off. **48** WLM is running in active mode, class and process total accounting are off. **50** WLM is running in passive mode, class and process total accounting are off. **51** WLM is running in active mode with no rset bindings, class and process total accounting are off. **52** WLM is running in passive mode with no rset bindings, class and process total accounting are off. **53** WLM is running in active mode for processor only, class and process total accounting are off. **54** WLM is running in active mode for processor only with no rset bindings, class and process total accounting are off. A message indicating the current state of WLM is printed on STDOUT.
-S *Superclass*	Requests an update of WLM that is limited to the subclasses of the Superclass. Use this flag with the **-u** flag. If the running configuration is a set of time-based configurations, Superclass must be given in the form "config/Superclass" where "config" is the regular configuration of the set which the Superclass belongs to. If "config" is the currently active configuration of the set, the changes will take effect immediately, else they will take effect at the next time "config" will be made active.
-T	Disables both class and process total limits accounting and regulation.
-T class	Disables only class total limits accounting and regulation.
-T proc	Disables only process total limits accounting and regulation.

Item	Description
-u	Updates the WLM. A single update operation can change the attributes, limits and shares of existing classes and/or add or remove classes. If the running configuration is a set, this operation refreshes the set description along with the content of all configurations of the set. Update can be used by a user with root authority to switch to an alternate configuration or configuration set. Update can also be used by a superclass administrator to update only the subclasses of the superclass he has administrative access to (using the **-S** flag).

Security

Access Control: Starting, stopping, switching from one mode to another, and updating superclasses or a configuration set requires root privileges. Updating the subclasses of a given superclass requires only admin user or admin group privileges (superclass administrator). Any user can query the state of WLM.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
classes	Contains the names and definitions of the classes.
limits	Contains the resource limits enforced on the classes.
rules	Contains the automatic assignment rules.
shares	Contains the resource shares allocated to the classes.
description	Contains the description text for each configuration.
groupings	Contains attribute value groupings for the configuration

wlmstat Command

Purpose

Shows Workload Manager (WLM) per class resource utilization statistics.

Syntax

wlmstat [**-l** *Class* | **-t** *Tier*] [**-S** | **-s**] [**-@**] [**-c**] [**-m**] [**-b**] [**-B** *Device*] [**-T**] [**-a**] [**-w**] [**-v**] [*Interval*] [*Count*]

wlmstat [**-l** *Class* | **-t** *Tier*] [-@] [**-c**] [**-m**] [**-b**] [**-u**] [*Interval*] [*Count*]

wlmstat [-l *Class* | -t *Tier*] [**-@**] [-M] [-S | -s] [-w] [-v] [*Interval*] [*Count*]

Description

The **wlmstat** command symbolically displays the contents of WLM data structures retrieved from the kernel. If a *Count* is specified, **wlmstat** loops *Count* times and sleeps *Interval* seconds after each block is displayed. If *Interval* and *Count* are not specified, one output report is produced. If *Interval* is specified but no *Count* is given, **wlmstat** outputs results continuously at the given interval until stopped by a signal (SIGINTR, SIGQUIT, and SIGKILL). By default, **wlmstat** displays the statistics for all the resources for every superclass and subclass. You can specify flags to narrow the focus of the statistics to a type of resource, tier, superclass, or subclass and alter the output format.

Note: The following should be considered when viewing the **wlmstat** output:

1. Starting with AIX 5.3, the WLM processor usage values and process priority adjustments are updated 10 times per second by default.
2. The value displayed for processor usage is not the current instantaneous usage from the last second, but is instead an average of the last *N* readings (starting with AIX 5.3, the default value for *N* is 15).

3. The Unmanaged class is used to report system interrupt time and for tracking memory usage for all of the pinned pages in the system that are not managed by the WLM. No processes are assigned to this class.

It is possible for a process with a hard limit of 50 percent to use more than 50 percent of the processor between two consecutive WLM usage updates. Each tenth of a second, every process is assigned a priority, and the scheduler then schedules all processes based on their assigned priorities. A process might receive more of the processor resources than the process hard limit between WLM updates.

By default, each instantaneous value of processor usage from each update is kept for the following 15 readings and is averaged with the other 14 readings before being displayed by **wlmstat**. This can potentially result in a value of greater than 50 percent due to a single instance of more than 50 percent usage between WLM updates.

The priority of a process will be greatly reduced and the process will be unable to run if the process consistently reaches or exceeds its hard limit. Over the long term, the resource utilization of the process must be at or under the process hard maximum. Over a short time interval, **wlmstat** may show the process using more than the process hard limit. The **/usr/samples/kernel/wlmtune** command that is available in the bos.adt.samples PTF can be used to modify the behavior of WLM in such an instance. The related tunables are:

schedhz
The frequency at which the WLM scheduler recalculates class consumption and priority for processor. The default is 10. Modifying this value changes the responsiveness of WLM. Increasing this value causes WLM to update more frequently, thereby reducing the possibility of a process exceeding its hard limit during a short time interval. The trade-off for this is increased overhead, since more WLM processing occurs. This can potentially affect overall system performance.

cpuhist
The number of consecutive processor consumption values used in the average calculation. The default is 15. Increasing this value further smooths the reported processor usage values by averaging over a longer period.

To make WLM more responsive so that classes do not exceed their maximums over long periods, it is recommended that you first try modifying **schedhz** until the **wlmstat** output displays the desired results. You may want to also modify **cpuhist** so that **wlmstat** averages over the same time interval. For example, if **schedhz** is 20 and **cpuhist** is 15, **wlmstat** will average over a period of 0.75 seconds (15/20), so you may want to change **cpuhist** to 30 so that **wlmstat** still averages over 1.5 seconds.

On systems with no contention for processor, an *Interval* of 5 for **wlmstat** is recommended in order to adhere to WLM limits.

Flags

Item	Description
-@	Displays workload partition resource information.
-a	Displays subclass consumption in absolute terms. By default, the subclass consumption percentages are shown relative to the superclass consumption. With this option, subclass consumption is displayed relative to the total amount of resource available on the system (as is done for superclasses). All values are displayed with 1% precision. For instance, if a superclass has a processor target of 20% and the processor percentage shown by **wlmstat** without **-a** for a subclass is 10%, **wlmstat** with **-a** shows the processor percentage for the subclass as 2%.
-b	Displays only disk I/O statistics.
-B *Device*	Displays disk I/O device statistics. Passing an empty string (-B "") displays the statistics for all the disks accessed by the class.
-c	Shows only processor statistics.

Item	Description
-l *Class*	Displays statistics for *Class* name. If not specified, all classes display along with a summary for appropriate fields.
-m	Shows only physical memory statistics.
-M	Displays the Real/Virtual Memory statistics. Use of the `-M` option adds the following columns in the output: **RMSIZ** Utilized real memory size for the class **VMSIZ** Utilized virtual memory size for the class **RMLIM** Real memory limit for the class **VMLIM** Virtual memory limit for the class **LGPGSIZ** Utilized large pages in the class **LGPGLIM** Large page limit for the class **Note:** A `-` will be displayed for the `RMLIM`, `VMLIM`, and `LGPGLIM` fields if the limit is undefined. When the `-M` and `-w` options are used together, `RMSIZ` and `VMSIZ` fields contain the high watermarks for these attributes instead of the actual utilized values. In addition, the `LGPGSIZ` and `LGPGLIM` fields is turned off.
-s	Displays only subclass statistics.
-S	Displays only superclasses statistics.
-t *Tier*	Displays statistics only for the specified *Tier*.
-T	Displays the total numbers for resource utilization since WLM was started or the class was created, whichever is the latter. The units are: **CPU** The total processor time, in milliseconds, consumed by a class **MEM** Unused **DKIO** The total number of 512 byte blocks sent/received by a class for all the disk devices accessed.

Item	Description
-v	Specifies verbose mode. This flag, intended for trouble shooting, also displays some class attributes, resource shares and limits and other WLM parameters, including internal parameter values intended for AIX support personnel. The following information can be of interest for users: **Column Header** Description **CLASS** Class name. **tr** tier number (0 to 9) **i** Value of the inheritance attribute: 0 = no, 1 = yes. **#pr** Number of processes in the class. If a class has no (0) process assigned to it, the values shown in the other columns might not be significant. **CPU** Processor utilization of the class (%). **MEM** Physical memory utilization of the class (%). **DKIO** Disk IO bandwidth utilization for the classs (%). **sha** Number of shares ('-' is represented as -1) **min** Resource minimum limit (%) **smx** Resource soft maximum limit (%) **hmx** Resource hard maximum limit (%) **des** (desired): percentage goal (target) calculated by WLM using the shares numbers (%) **npg** Number of memory pages owned by the class. The other columns are for internal use only and bear no meaning for administrators and end users. This format is better used with a resource selector (**-c**, **-m**, or **-b**), otherwise the lines might be too long to fit into a line of a display terminal.
-w	Displays the memory *high water mark*, that is the maximum number of pages that a class had in memory at any given time since WLM was started or the class was created (whichever happened last).
-u	Displays per-tier and total unused resources.

Display

Results are tabulated, with the following fields:

Name	Class name
CPU	Percentage of total processor time consumed by the class.

Name	Class name
MEM	Percentage of physical memory consumed by the class.
DKIO	Percentage of the disk IO bandwidth consumed by the class. This number is the average of the disk bandwidth on all the disk devices accessed by the class, and is usually not significant. For instance if a class consumes 80% of the bandwidth of one disk and 5% of the bandwidth of two other disks, the DKIO column shows 30%. For details on the per device utilization, use the -B device option.

Examples

1. To get a printout of WLM activity right now, enter:

```
wlmstat
```

   This produces the following output:

```
                CLASS CPU MEM DKIO
          Unclassified   0   0   0
             Unmanaged   0   0   0
               Default   0   0   0
                Shared   0   0   0
                System   0   0   0
                class1  12   0   0
    class1.Default     4   0   0
     class1.Shared     0   0   0
  class1.subclass1     4   0   0
  class1.subclass2     4   0   0
                class2  12   0   0
    class2.Default     4   0   0
     class2.Shared     0   0   0
  class2.subclass1     4   0   0
  class2.subclass2     4   0   0
```

2. To get a report for superclass **class1**, enter:

```
wlmstat -l class1
```

   This produces the following output:

```
                CLASS CPU MEM DKIO

               class1  12   0   0
    class1.Default     4   0   0
     class1.Shared     0   0   0
  class1.subclass1     4   0   0
  class1.subclass2     4   0   0
```

3. To get a report for subclass **sclass1.subclass2** updated every 10 seconds, for one minute, enter:

```
wlmstat -l class1.subclass2 10 6
```

   This produces the following output:

```
                CLASS CPU MEM DKIO
  class1.subclass2     4   0   0
  class1.subclass2     4   0   0
  class1.subclass2     4   0   0
  class1.subclass2     4   0   0
  class1.subclass2     4   0   0
  class1.subclass2     4   0   0
```

4. To display virtual/real memory statistics, enter:

```
wlmstat -M
```

This produces the following output:

```
CLASS          RMSIZ    RMLIM   VMSIZ    VMLIM  LGPGSIZ LGPGLIM
Unmanaged       1024      4096   4096     8192     0       -
Default         0           -      0       -       0       -
Shared          0           -      0       -       0       -
System          23567    50000  819234   1000000   0       -
```

5. To display the memory high water mark, enter:

```
wlmstat -M -w
```

This produces the following output:

```
CLASS      RMSIZ RMLIM  VMSIZ   VMLIM
Unmanaged  1024  4096   4096    8192
Default    0     -      0       -
Shared     0     -      0       -
System     23567 50000  819234  1000000
```

Errors

A warning message is issued by **wlmstat** if WLM is not started.

wol command

Purpose

Wakes up one or more hosts that are connected to a network in suspend mode by sending a Magic Packet.

Syntax

To send a Magic Packet to a subnet-directed broadcast address:

wol { [**-m** *MACAddress* [[**-h** *Host* **-s** *SubnetMask*] | **-i** *Interface*] | **-f** *File* } [**-v**]

To send a Magic Packet to a multicast address:

wol { **-m** *MACAddress* **-M** *MulticastAddress* [**-p** *Port*] [**-i** *Interface*] | **-f** *File*} [**-v**]

Description

The **wol** command wakes up one or more hosts that are connected to a network in suspend mode by sending a Magic Packet to the specified address or addresses on the specified subnet.

If the user doesn't specify either the **-h**, nor **-s** flag, the **wol** manager will broadcast the Magic Packet as follows:

- If the user specifies the interface name (**-i** *Interface*), the Magic Packet will be broadcast from the specified interface.
- If the user doesn't specify the interface name, then the **wol** manager will loop through each network interface installed on the machine. If an interface is up, it will broadcast the Magic Packet from that interface, and then continue to the next interface until it goes through the entire interface list on the machine.

The file specified with **-f** *File* contains the list of hosts which need to be awakened. This file consists of one or more lines, each line containing the following information in this format:

MacAddress; Hostname/IPaddress; SubnetMask; Multicast; Port; Interface

For example, the file might look like this:

```
00:20:35:7a:7:89a;        9.41.86.19;      255.255.255.0  ; ; ;
00:04:ac:17:c0:9f ;        obiwan.aoot.austin.ibm.com;       255.255.255. 224; ; ;
00:07:be:4a:2:394; ; ; ; en0
00:06:38:6b:7e:8f ;         ; ; 234.5.6.7;   12345 ;
```

A line starting with a "#" character is a comment and is ignored. Each line contains 6 tokens separated by ";" character. The MAC address is mandatory. The other tokens are optional, but the ";" character must be used to separate unused tokens.

Flags

Item	Description
-i *Interface*	Specifies the interface to use on the host where the **wol** command is being run
-f *File*	Specifies the name of a file containing a group list. This allows the user to wake a specified group of hosts.
-h *Host*	Specifies a host to wake, either as a hostname or as an IPv4 address in dot string representation (for example, 10.0.0.3).
-m *MACAddress*	Specifies the a 48 bits MAC address of the host in hex representation (for example, 00:20:35:7a:78:9a).
-M *MulticastAddress*	Specifies an IPv4 multicast address.
-p *Port*	Specifies the port to use on the multicast machine.
-s *SubnetMask*	Specifies an IPv4 subnet mask in dot string representation (for example, 255.255.255.0).
-v	Specifies verbose mode.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Location

/usr/sbin/wol

wparerr Command

Purpose

Logs an error message for a specific Workload partition (WPAR).

Syntax

From global

```
wparerr –w  WparName [{-c <cat file> -s <set no> -n <msg no>}]  -S default_string args
```

From WPAR

```
wparerr  [ -c  <cat file> -s <set no> -n <msg no>}]  -S default_string args
```

Description

The **wparerr** command provides a mechanism to log error messages for a given WPAR. Each WPAR can hold up to 1 KB of error messages. If there is enough space to log a new message, the command logs the message; otherwise, it fails. The *-w* option should not be used inside a WPAR. Everything after *–S* flag is treated as arguments for the message.

Flags

Item	Description
-w	Specifies the name of the workload partition for which the message should be logged.
-c	Specifies the catalog file name to be used for translation.
-s	Specifies the message set number of the error message in the catalog file.
-n	Specifies the message number of the error message.
-S	Specifies the default message string. Follows the same syntax as the **printf** subroutine *Format* parameter. Floating point is not supported.
args	Specifies the arguments to the message if any.

Security

Attention RBAC users and Trusted AIX users

This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see **Privileged Command Database** in the security. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** sub commands.

Example

- To log a message for the WPAR *mywpar* from global:

```
wparerr –w mywpar –c wparerrs.msg –s 1 –n 12 –S “ %s failed with return value %d\n” lswpar -1
```

- To log a command failure from inside a WPAR:

```
wparerr –c wparerrs.msg –s 1 –n 5 –S “ %s application does not allow checkpoint\n”
myapplication
```

wparexec Command

Purpose

Creates an application workload partition or specification file.

Syntax

wparexec [**-a**] [**-c**] [**-1**][-F] [-h *hostname*] [-H *architecture*] [**-i**] [**-I** *attribute=value* ...] ... [-M *attribute=value*...] ... [-N *attribute=value*...] ... [-R *attribute=value*...] [-u *userscript*] [-v] [-x] { -n *wparname* [-e *existingwpar* | -f *infile*] [-o *outfile* [-w]] | -f *infile* [-n *wparname*] [-o *outfile* [-w]] | -w -o *outfile* [-n *wparname*] [-e *existingwpar* | -f *infile*] } [[--] [*var*=*value* ...] */path/to/command* [*arg* ...]]

Note:

Regardless of locale, only ASCII characters are allowed as arguments to the **wparexec** command.

In addition to this, there are more restrictions for a WPARs name:

- May not be more than 25 bytes.
- May not contain white space or any of the following symbols:

  = : / ! ; ` ' " < > ~ & ( ) * + [ ] , . ^ 0 { } | \
- May not start with '-' or '0'.

Description

The wparexec command builds and starts an application workload partition, or creates a specification file to simplify the creation of future application workload partitions.

An application workload partition is an isolated execution environment that might have its own network configuration and resource control profile. Although the partition shares the global file system space, the processes that are running therein are only visible to other processes in the same partition. This isolated environment allows process monitoring, gathering of resource, accounting, and auditing data for a predetermined cluster of applications.

The wparexec command starts and monitors a single application within this isolated environment. The wparexec command returns synchronously with the return code of this tracked process only when all of the processes in the workload partition terminate. For example, if the tracked process creates a daemon and exits with the 0 return code, the wparexec command will block until the daemon and all of its children terminate, and then exit with the 0 return code, regardless of the return code of the daemon or its children.

Flags

Item	Description
-1	Creates the configuration only. Causes the **wparexec** command to stop after creating the configuration of the application WPAR. The **startwpar** command must then be used to start the WPAR. Only advanced users can use the **-1** option.
-a	Automatically resolves conflicting static settings if required. Resolvable settings are name, hostname, and network configuration.
-c	Enables this workload partition to be checkpointed. This option is only valid when additional checkpoint-restart software is installed and configured. When this option is used, any file systems associated with this workload partition (for example, with the **-M** option) must be remote (for example, **vfs**=**nfs**).
-e *existingwpar*	Uses an existing application workload partition as the source for specification data. Do not use the **-e** flag with the -f flag. Any values specified by other wparexec flags override those values from the existing workload partition.
-f *infile*	Indicates the specification file to read default values from. Do not use with the -e flag. Any values specified by other wparexec flags override those values from the loaded specification file.

Item	Description
`-F`	Suppresses or overrides most error conditions. With the `-F` flag, the `wparexec` command continues with a warning.
`-h` *hostname*	Specifies a host name for this workload partition. If not specified, the `wparexec` command uses the workload partition name as host name.
`-H` *architecture*	Creates an architecture-compatible workload partition. Valid architecture values are `pwr4, ppc970, pwr5, pwr6,` and `pwr7`. The architecture value must be earlier, or equal to, the system hardware version. The applications in the workload partition are presented with the lowest common denominator of the specified architecture. If the workload partition can create a checkpoint, the workload partition is able to migrate between systems with hardware levels greater than, or equal to, the workload partition architecture. **Note:** The POWER5 processor-based systems and BladeCenter JS21 Express systems are not compatible with each other. You cannot create a JS21-compatible (ppc970) WPAR on a POWER5 processor-based system even though the JS21 using an earlier processor than a POWER5 processor-based systems.
`-i`	Enables WPAR-specific routing for the workload partition. By default, outgoing network traffic from a workload partition is routed like it is being sent from the global environment, notably in the following ways: • Traffic between addresses that were hosted on the same global system is sent through the loopback interface. • Routing table entries that are configured in the global system, including the default route, are used to transmit workload partition traffic. If you enable WPAR-specific routing by specifying the **-i** flag, the workload partition creates and uses its own routing table for outgoing traffic. Routing entries are created automatically for each of the network addresses of the workload partition to accommodate broadcast, loopback, and subnet routes. For more information about the network attributes, see the **-N** flag. You can create explicit additions to the routing table of the workload partition using the **-I** flag. In particular, you can use the **-I** flag to configure the default route, as no default route is created automatically.
`-I` *attribute=value ...*	Adds routing table entries to the entries that are automatically created when WPAR-specific routing is in effect. You can specify more than one **-I** flag to configure multiple routes. Using the **-I** flag automatically enables WPAR-specific routing as described under the **-i** flag. You can specify the following attributes with the **-I** flag. The **rtdest** attribute and the **rtgateway** attribute are required to be specified.
rtdest=*destination*	Identifies the host or network to which you are directing the route. You can specify the value using either a symbolic name or a numeric address. You can use the keyword **default** to specify a default route. For more information about the route **rtdest** attribute, see the *Destination* parameter of the **route** command.
rtgateway=*gateway*	Identifies the gateway to which packets are addressed. You can specify the value using either a symbolic name or a numeric address.
rtnetmask=*A.B.C.D*	Specifies the network mask to the destination address.
rtprefixlen=*n*	Specifies the length of a destination prefix, which is the number of bits in the netmask. The value must be a positive integer.
rttype={*net*|*host*}	Forces the **rtdest** attribute to be interpreted as the specified type.
rtinterface=*if*	Specifies the interface, for example, `en0`, to associate with the route so that packets are sent using the interface when the route is chosen.
61V**rtfamily =***{inet|inet6}*	Specifies the address family. For information about the parameters of the **rtfamily** flag, see the parameter section of the route command.

Item	Description
`-M directory=`*dir* [`vfs=`*type*] [`dev=`*devicepath*] [`host=`*remotehost*] [`mountopts=`*mountopts*]	Specifies file system dependencies only. Attributes must be space-separated. By default, an application workload partition has the same level of access to all of the global file systems and mounts as the user who created the workload partition. Use the `-M` flag with the directory attribute that is set to the file system name to specify additional file systems. More than one `-M` flag can be specified. **Note:** All of the mounts and all of the directories are created and available at global level. File systems that are based on disk, such as the `vfs=`*jfs* and the `vfs=`*jfs2*, will not be created for application workload partitions. A local file system dependency can be added by defining only the directory attribute. However, the directory specified must exist in the `/etc/filesystems`. If an error occurs during the process of creating the workload partition, any file systems mounted by the `wparexec` command are unmounted. After the creation succeeds, the file systems are not unmounted, regardless of the return status of the user application. The following are the valid values for the *vfs* attribute for application workload partitions: **nos** The directory specified by the `dev` attribute on the system specified by the host attribute is mounted at the location that is specified by the directory attribute. If the mount point does not exist, it will be created. The only other attributes that are applicable to an `nfs` mount are the `mountopt` attributes, corresponding to the **-o** option of the **mount** command or the `options` attribute in an `/etc/filesystems` stanza. If not specified, no mount options are used by default. Acceptable option values correspond to the `-o` options to the **mount** command. **namefs** The global directory specified by the `dev` attribute is mounted over the directory specified by the `directory` attribute. The only other attributes that are applicable to a `namefs` mount are the `mountopt` attributes. **directory** The global directory specified by the `directory` attribute is created if it does not exist. No mounting is performed.
`-n` *wparname*	Specifies the name for the workload partition to be created. If no name is supplied, a name is generated.
`-N` *attribute=value*	Allows specification of the following network configuration attributes: • **interface**= *if* or **interface**=*namemappedif* • **address**=*A.B.C.D* • **netmask**=*A.B.C.D* • **broadcast**=*A.B.C.D* • **address6**=*S*:*T*:*U*:*V*:*W*:*X*:*Y*:*Z* • **prefixlen**=*n* The name-mapped interface is defined in the **/etc/wpars/devmap** file. You can specify the mapping between the name-mapped interface and the system interface as follows: `# The comments start with '#'` `# Each line contains a pair of name-mapped interface` `# and real interface separated by tab or blank spaces.` `foo en0` `goo en1` `soo en2` The *attribute=value* pairs must be separated by spaces. More than one `-N` flag can be used to configure multiple IP addresses. At minimum, the `address` or the **address6** attribute must be specified. The **wparexec** command collects any other values that are not specified from the global system settings. If no `-N` flag is specified, the `wparexec` command will attempt to discover an appropriate IP address for the workload partition by running the `gethostbyname` subroutine on the workload partition name specified with the `-n` flag. If an address is found on the same subnet as any global interface, the settings of that interface will be used with the resolved IP address to create the default network entry. To define an IPv6 network configuration, specify the **-N** flag with the **address6** attribute, the **prefixlen** attribute, and the **interface** attribute: • The **address6** attribute is a 128-bit address. The address is represented by eight 16-bit integers that are separated by colons. Each integer is represented by four hex digits. Leading zeros can be skipped, and consecutive null 16-bit integers can be replaced by two colons (one time per address). • The **prefixlen** attribute is the number of high-order bits that are used to mask the IPv6 address and to comprise the prefix. The value of the **prefixlen** attribute ranges from 0 through 128. Each **-N** flag can accept either IPv4 attributes, or IPv6 attributes, but not both.
`-o` *outfile*	Indicates an output path and file name to write specification data to. This specification file can be used to create an application workload partition later with the `-f` flag.

Item	Description
-R *attribute=value*	Allows specification of resource control attributes. Only one -R flag can be specified. Most resource controls are similar to those supported by the Workload Manager (WLM). See the listed WLM pages for descriptions of these attributes. Valid attributes are as follows: **active** Allows resource control definitions to be retained, but rendered inactive. This attribute can take the *yes* or *no* values. **rset** Configures this workload partition to use a resource set created by the **mkrset** command. **shares_CPU** The number of processor shares available to this workload partition. For more information about processor shares, see Workload Manage shares File in in *Files Reference*. **CPU** The percentage of processor limits for this workload partition's processes. This attribute uses the following format to definite the limits values: `CPU=<m>%-<SM>%,<HM>%` The *m* value represents the minimum limit. The *SM* value represents the soft maximum limit. The *HM* value represents the hard maximum limit. For more information about limited values, see the Workload Manager limits File in in *Files Reference*. **shares_memory** The number of memory shares available to this workload partition. For more information about memory shares, see Workload Manager shares File in in *Files Reference*. **memory** The percentage of memory limits for this workload partition's processes. For more information about memory limit, see the Workload Manager limits File in in *Files Reference*. **procVirtMem** The maximum amount of virtual memory that a single process can use. Processes that exceed the specified limit are terminated. The valid units are megabytes (M or MB), gigabytes (G or GB), and terabytes (T or TB). The minimum limit allowed is 1M. The maximum limit that can be specified is 8796093022207M, 8589934591G, or 8388607T. If the value is set to -1 (no units), the limit is disabled. For more information about limit values, see Workload Manager limits File in in *Files Reference*. **totalVirtMem** The maximum amount of virtual memory that can be used by the WPAR as a whole. Processes that cause the specified limit to be exceeded will be terminated. The valid range and units are the same as for procVirtMem. If the value is set to '-1' (no units), the limit is disabled. See Workload Manager limits File in *Files Reference*. **totalProcesses** The total number of processes that are allowed in this workload partition. For more information about allowed processor number, see Workload Manager limits File in in *Files Reference*. (*Attributes for the* **-R** *flag, continued*): **totalPTYs=*n*** The total number of pseudo terminals that are allowed in the workload partition. For more information about the allowed pseudo terminals, see pty Special File. **totalLargePages=*n*** The number of large pages that can be allowed for the workload partition. For more information about the allowed large pages, see Large Pages. **pct_msgIDs=*n*%** The percentage of the maximum number of message queue IDs of the system that are allowed in the workload partition. For more information about the allowed number of message queue IDs, see Message Queue Kernel Services. **pct_semIDs=*n*%** The percentage of the maximum number of semaphore IDs of the system that are allowed in the workload partition. **pct_shmIDs=*n*%** The percentage of the maximum number of shared memory IDs of the system that are allowed in the workload partition. For more information about the allowed number of shared memory IDs, see Shared Memory. **pct_pinMem=*n*%** The percentage of the maximum pinned memory of the system that can be allocated to the workload partition. For more information about pinned memory, see Support for pinned memory. **totalThreads** The total number of threads that are allowed in this workload partition. For more information about allowed number of threads, see Workload Manager limits File in in *Files Reference*.

Item	Description
-u *userscript*	Specifies the path to a user script to be run by the workload partition commands at various administration points. The parameter of the **-u** flag can be a quoted string including additional arguments to be passed to the script. In all cases, the first component of the parameter of the **-u** flag must be an absolute path to an existing executable file. The script is invoked as follows: `/path/to/userScript<action><WAPR>` The *action* argument indicates the administrative action being performed, as follows: • WPAR_LOAD: A script runs in the global environment after kernel configuration, before the tracked process is created. If the script returns a value of non-zero, the workload partition will not be started. • WPAR_START: A script runs in the global environment after the workload partition becomes active. For application workload partitions, the script runs once the tracked process is started. **Note:** This code path can be run asynchronously by a dissociated process with its standard I/O streams closed or redirected. Internal messaging must be handled accordingly, and the script must account for the fact that short-lived workload partitions might be stopped or stopping at any point during the execution of the script. If the script returns a value of nonzero, a warning is logged, but no other behavior changes. • WPAR_STOP: A script runs in the global environment after all of the workload partition processes finish, and before the kernel is unconfigured. **Note:** This code path can be executed by a dissociated process with its standard I/O streams closed. Internal messaging should be handled accordingly. If the script returns a value of non-zero, a warning will be logged, but no other behavior will change.
-v	Specifies the command to run in the verbose mode.
-x	Allows access to cross-WPAR semaphores and shared memory segments.
-w	Writes the specification file. When it is used with the -o flag, the -w flag causes the wparexec command to quit after writing the new specification file without actually creating the workload partition.
[--] [*var*=*value*] */path/to/command* [arg ...]	Specifies the application (tracked process) to be run within the workload partition, along with any necessary environment variable settings and arguments. The command is required, either by this command-line syntax or the general.application attribute in the specification file, unless the command is only creating a specification file (with -w flag). When it is started, the command line provided is always shell-expanded within the workload partition. When using the command line, shell metacharacters should be escaped appropriately to prevent premature expansion. The special double-minus separator (--) is used to signify that all subsequent command-line arguments comprise the tracked process. For example, use this separator to remove ambiguity between attributes to the -N flag and assignment of environment variables to the tracked process. Only one tracked process per workload partition is supported, but this application might create other processes. The workload partition is automatically stopped and removed when all of the processes therein terminate. A workload partition might be stopped and removed prematurely by the stopwpar command.

Security

Access Control: Only the root user can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To create an application workload partition that is running a benchmark program, enter:

   ```
   wparexec -n tpcc -N address=192.168.0.51 /u/tpcc/benchmark -f /tmp/logfile
   ```

   Note: The -f flag is passed to the /u/tpcc/benchmark file and is not processed by the wparexec command as a flag.
2. To create a workload partition based on an existing specification file, enter:

   ```
   wparexec -f /tmp/wparexec1.spec
   ```
3. To override the default minimal *PATH* variable provided by the wparexec command, enter:

```
wparexec PATH=/usr/opt/bin:/usr/bin:/usr/sbin /home/joe/runapp
```

Files

Item	Description
`/usr/samples/wpars/sample.spec`	An annotated workload partition specification file.

wpar_reg_script Command

Purpose

Allows a user to register scripts to be run during different phases of a system WPAR live application mobility. Currently, only the WPAR restart phase is supported.

Syntax

/usr/lib/wpars/wpar_reg_script [`-r` | `-u`] [`-p` *phase*] [`-s` */path/to/script*]

Description

The `wpar_reg_script` command allows a user to register scripts to be run during specific phases of WPAR live application mobility.

Registering a script will place an entry in a new ODM class called `CuWscr` that contains the full path to the script and its arguments.

This script can be used with kernel extensions and mobility to load kernel extensions on the arrival node before other processes start.

Restrictions

The `wpar_reg_script` command can only be run from inside a WPAR.

Only the file systems that are part of the WPARs config are available at the time of execution (not privately mounted file systems). These file systems are only available read-only.

No processes can be left running when the script exits.

Loading kernel extensions with this script for use with mobility is restricted to `SYS_SINGLELOAD.`

Flags

Item	Description
`-r`	Register an entry including its script and phase to the `CuWscr` class.
-u	Unregister an entry from the `CuWscr` class.
`-p`	The phase that the command will be executed during. Value of 1 corresponds to WPAR restart. More may be added in the future. If no value is specified, it will use a default value of 1.
`-s`	The path to the script and its arguments that will be executed. Surround the full command with "" to include command line parameters.

Examples

1. To register the script /usr/sbin/foo/ to execute:

```
/usr/lib/wpars/wpar_reg_script -r -s /usr/sbin/foo
```

2. To register the script /usr/sbin/foo/ with arguments:

```
/usr/lib/wpars/wpar_reg_script -r -s "/usr/sbin/foo args"
```

3. To Unregister all instances of the script /usr/sbin/foo/ foo:

```
/usr/lib/wpars/wpar_reg_script -u -s /usr/sbin/foo
```

4. To unregister all scripts that run during phase 1:

```
/usr/lib/wpars/wpar_reg_script -u -p 1
```

wparprnterr Command

Purpose

Displays error messages specific to a Workload partition (WPAR).

Syntax

From global

```
wparprnterr  WparName
```

Description

The **wparprnterr** command prints all the error messages that are logged for a WPAR by using the **wparerr** command, **wpar_log_err**, and **kwpar_err** subroutines on the standard output.

Security

Attention RBAC users and Trusted AIX users

This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see **Privileged Command Database** in Security. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To display messages of the WPAR *mywpar*:

```
wparprnterr mywpar
```

write Command

Purpose

Opens a line of communication to send messages to other users on the system in real time.

Syntax

To query all messages awaiting replies from users on a host and display them with their handles, type the following:

write -q [**-n** *Host*]

To Reply to a Message Sent by a Utility or a Shell Script, or Redisplay the Message Associated with a Given handle, type the following:

write -h*Handle,* { ok | cancel | query } [**-n** *Host*]

To send messages to a user, optionally on another host or a particular device, type the following:

write [**-r**] { [**-n** *Host*] *User* | *User***@***Host* } [*Line*]

Description

The **write** command enables message sending over the system in real time. It provides conversation-like communication with another logged-in user. Each user alternately sends and receives short messages from the other workstation. Long messages can be sent by putting the complete message in a file and then redirecting that file as input to the **write** command.

For another user (as specified by the *User* parameter) to receive a message, that user must be logged in and must not have refused message permission. When a message is sent to a user who is not logged in, the message `user not logged in` appears. If the message is sent to a user who has refused message permission by setting the **mesg** command to no, the message `write: permission denied` appears.

When the **write** command is issued, it immediately sends the following message, along with an attention-getting sound (the ASCII BEL character) to the message recipient or target:

```
Message from SenderID on SenderHostname (ttynn) [Date] ...
```

With a successful connection, the **write** command sends two ASCII BEL characters to both workstations. The beep alerts the sender that the message can begin and it alerts the receiving user that a message is coming.

Sending occurs one line at a time as the Enter key is pressed. The communication link from the sender to the receiver remains open and sending continues until the Ctrl-D key sequence ends the sending link. Then an end-of-text character (<EOT>) is sent to the target workstation and the **write** command mode is terminated.

The receiving or target user can respond by sending a **write** command to the originating user. This opens a line of communication from the receiver back to the sender, enabling message responses in return. For this type of exchange, the following convention is useful: When you first write to others, wait for a response before sending any text. End a message with a signal such as o (over) to alert the other person to reply. Use oo (over and out) when the conversation is finished.

If the character `!` (exclamation point) is found at the beginning of a line, the **write** command calls the shell to execute the rest of the line as a command.

When you write to a user who is logged in at more than one workstation or multi-using more than one process, the **write** command uses the first login instance found in the **/etc/utmp** file as the message delivery point (usually the login or console shell), and you get the message:

```
UserID is logged on more than one place.
You are connected to "Workstation".
Other locations are:
Workstation
```

When this message is received, if you wish to send the message to a location other than the initial login location, the target user can be contacted at a different location by specifying the *Line* of the location (tty00, for example).

Permission to write to another user is granted or denied by the individual user with the **mesg** command. Some commands deny message permission while they are running to prevent interference with their output. A user with root user authority can write to any workstation regardless of the workstation's message permission.

You can use the **write** command to converse with users on other hosts. You can identify a user on a remote host by using the **-n***HostName* flag or the *User***@***Host* parameter. In order to write to a user on a remote host, the **writesrv** daemon must be running on both the current host and the remote host.

The **write** command is also used by the **qdaemon** daemon to send messages to users on other hosts and to wait for replies. There are only three valid replies:

Item	Description
`ok`	The original write exits with a status of 0.
`cancel`	The original write exits with a status of 1.
`query`	The message associated with the given handle is displayed.

Parameters

Item	Description
User	Specifies the user ID of the person to receive the message text.
*User***@***Host*	Specifies the user ID and remote host of the person to receive the message text.
Line	Contacts the target user at another location (`tty00`, for example).

Flags

Item	Description
-h *Handle***,***Reply*	Replies to a message sent by a utility or shell script using write with the reply option. The value to be used for the *Handle* variable is generated internally and supplied to the user in the text of the original message. The reply can be `ok`, `cancel`, or `query`.
-n*Host*	Specifies a remote host. The *Host* variable may be a nickname or an internet address.
-q	Queries all messages awaiting replies from users on a host and displays them with their handles.
-r	Generates a message handle, places it in the message header, sends the message, and waits for a reply. This flag is used by the **qdaemon** daemon for operator messages and can be put in shell scripts. It is not used for interactive conversations. An exit status of 0 indicates that the reply was `ok`, a status of 1 indicates that the reply was `cancel`, and an exit status of 2 indicates that the user could not be contacted.

Requirements:

- The **writesrv** daemon must be running on the target host in order for any of the flags to work. If you are not using either the **-n** flag or **@***Host*, but using **-h**, **-q**, or **-r**, the **writesrv** daemon must be running on your host.
- If TCP/IP is not installed on your machine but the *HostName* is set, in order to converse with users on the local host using the **write** command with the **-h**, **-q**, or **-r** flag, you must append your host name to the end of the `loopback` entry in the **/etc/hosts** file. The original entry should read:

```
127.0.0.1 loopback LocalHostName
```

The new entry should read:

```
127.0.0.1 loopback LocalHostName HostName
```

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	The addressed user either is not logged on or denies permission.

Examples

1. To write a message to a user who is logged in, enter:

   ```
   write june
   ```

   Press the Enter key and type,

   ```
   I need to see you! Meet me in the computer room at 12:30.
   ```

   Then press the Ctrl-D key sequence to terminate the **write** command mode.

   If your user ID is `karen` and you are using workstation `tty3`, `june`'s workstation displays:

   ```
   Message from karen on trek tty3 Aug 17 11:55:24  ...
   I need to see you!  Meet me in the computer room at 12:30.
   <EOT>
   ```

2. To hold a conversation, enter:

   ```
   write june
   ```

   Press the Enter key and type,

   ```
   Meet me in the computer room at 12:30.
   o
   ```

   This starts the conversation. The o at the beginning of the next line means the message is over. It tells June that you are waiting for a response. Do not press Ctrl-D if you wish to continue.

   Now June replies by typing:

   ```
   write karen
   ```

   Presses the Enter key and types,

   ```
   I'm running tests at 12:30. Can we meet at 3?
   o
   ```

   And you might respond:

   ```
   OK--the computer room at 3.
   oo
   ```

   The oo means *over and out*, telling June that you have nothing more to say. If June is also finished oo, then you both press Ctrl-D to end the conversation.

3. To write someone a prepared message, enter:

   ```
   write june < message.text
   ```

This writes the contents of the **message.text** file to `june`'s workstation.

4. To write to the person using a certain workstation, enter:

```
write -n console
```

Press the Enter key and type,

```
The printer in building 998 has jammed.
Please send help.
```

Then press the Ctrl-D key sequence.

This writes the message to the person logged in at the workstation `/dev/console`.

5. To send a message to user `spuds` at host `partya`, enter:

```
write -n partya spuds
```

Press the Enter key and type,

```
Your new tape has just arrived,
come see me to pick it up.
Thanks!
```

Then press the Ctrl-D key sequence.

OR

```
write spuds@partya
```

Press the Enter key and type,

```
Your new tape has just arrived,
come see me to pick it up.
Thanks!
```

Then press the Ctrl-D key sequence.

6. Here is an example of a message sent by the **qdaemon** daemon:

```
Message from mary on trek (tty10) Aug 17 10:03:34 ...
Use "write -h 6398492,reply" to reply
Please insert tape number 5 into rmt0.
<EOT>
```

To reply in the affirmative, enter:

```
write -h 6398492,ok
```

Then press the Ctrl-D key sequence.

To reply in the negative, enter:

```
write -h 6398492,cancel
```

Then press the Ctrl-D key sequence.

With the **-h** flag, there is no need to supply the host name or user ID. This information is tracked with the handle.

Files

Item	Description
/etc/hosts	Contains TCP/IP host information.
/etc/utmp	Contains user and accounting information for the **who**, **write**, and **login** commands.

writesrv Daemon

Purpose

Allows users to send messages to and receive messages from a remote system.

Syntax

writesrv

Description

The **writesrv** daemon allows users to send messages to users on a remote system and receive responses from users on a remote system with the **write** command.

The **writesrv** utility receives incoming requests from a **write** command and creates a server process to handle the request. This server process communicates with the client process (**write**) and provides whatever services are requested.

To perform these services, the **writesrv** daemon creates a socket that is attached to the port defined in the **/etc/services** file. All requests for service are sent as messages to this socket.

> **Note:** If the **writesrv** daemon terminates abnormally (such as a system crash, power failure, or the **kill -9** command), the **/var/spool/writesrv** directory must be manually cleaned out to remove any files left behind by the **writesrv** daemon.

Examples

1. To start the **writesrv** daemon from the **/etc/rc** script, enter:

   ```
   /usr/sbin/writesrv
   ```

   The **writesrv** daemon is started from the **/etc/rc** script. This is the usual way the daemon is started.
2. To start the **writesrv** daemon using the System Resource Controller (SRC), enter:

   ```
   startsrc -s writesrv &
   ```

   The **writesrv** daemon is started using SRC.

Files

Item	Description
/etc/services	Contains the Network Services directory.

wtmpfix Command

Purpose

Manipulates connect-time accounting records by correcting date and time stamp inconsistencies.

Syntax

/usr/sbin/acct/wtmpfix [*File* ...]

Description

The **wtmpfix** command is called by the **runacct** procedure to examine standard input or *File*s that contain records in **wtmp** format, and correct problems that could make the **acctcon1** or **acctcon2** commands fail. The **wtmpfix** command corrects date and time stamp inconsistencies, and writes the corrected records to standard output. If the date and time stamps are not consistent when the **acctcon1** command runs, the **acctcon1** command generates an error and stops.

The **wtmpfix** command also checks the validity of the name field to ensure that it consists only of alphanumeric characters, a $ (dollar sign), or spaces. If the name is invalid, the **wtmpfix** command changes the login name to **INVALID** and writes a diagnostic message to standard error. In this way, the **wtmpfix** command reduces the chance that the **acctcon2** command will fail.

Each time the date is set (on system startup or with the **date** command), a pair of date change records is written to the **/var/adm/wtmp** file. The first record is the old date, denoted by the *old time* string. The *old time* string is placed in the line field and the **OLD_TIME** flag is placed in the type field. The second record is the new date, denoted by the string *new time.* The *new time* string is placed in the line field and the **NEW_TIME** flag is placed in the type field. The **wtmpfix** command uses these records to synchronize all date and time stamps in the file.

Flags

None.

Parameters

Item	Description
File	Specifies the file to examine that contains records in **wtmp** format.

Security

Access Control: These commands should grant execute (x) access only to members of the **adm** group.

Examples

1. To correct date and time stamp inconsistencies in **/var/adm/wtmp** and write the corrected records to **dummy.file**, enter:

```
/usr/sbin/acct/wtmpfix  /var/adm/wtmp > /tmp/dummy.file
```

Files

Item	Description
/usr/sbin/acct/wtmpfix	Contains the **wtmpfix** command.
/var/adm/wtmp	Contains records of date changes that include an old date and a new date.
/usr/include/utmp.h	Contains history records that include a reason, date, and time.

wump Command

Purpose

Starts the hunt the wumpus game.

Syntax

wump

Description

A wumpus is a creature living in a cave with many rooms interconnected by tunnels. You move among the rooms trying to shoot the wumpus with an arrow and trying to avoid being eaten by the wumpus or falling into bottomless pits. There are also super bats that may pick you up and drop you in some randomly selected room. For moving among the rooms and shooting arrows, the game prompts you with appropriate questions and follows your instructions. For example:

```
You are in room 14.
I feel a draft.
There are tunnels to 1 13 18.
Move or shoot? (m-s) m
Which room? 1
You are in room 1.
I feel a draft.
There are tunnels to 14 17 18.
Move or shoot? (m-s) m
Which room? 17
You are in room 17.
You fell into a pit!
Another game? (y-n)
```

In the above example, you start out in room 14. The computer displays `I feel a draft.` This is the hint that a pit is nearby. You choose to move to room 1. Again you are warned of the pit. You then choose to move to room 17 where you fall into a pit and die.

At the beginning of the game, you are prompted `Instructions? (y-n)`. Choosing y provides an explanation of the warnings, how to move, and how to shoot.

The game ends and you are prompted `Another game? (y-n)` if:

- You kill the wumpus.
- The wumpus eats you.
- You fall into a bottomless pit.
- You run out of arrows.

To quit the game at any time, press the interrupt (Ctrl-C) key sequence.

Files

Item	Description
/usr/games	Contains the location of the system's games.

X

The following AIX commands begin with the with the letter *x*.

X Command

Purpose

Starts the X Server.

Syntax

X [**-a** *Number*] [**-auth** *FileName*] [**-bc** | **+bc**] [**-bp** *Color*] [**-broadcast**] [**-bs** | **-nobs**] [**-c** *Volume*] [**-cc** *VisualType* [**:***Display*]] [**-class** *DisplayClass*] [**-co** *File*] [**-cookie** *XDMAuthenticationBit*] [**-D** *File*] [**-d** *Depth* [**:***Display*]] **-displayID** *DisplayID*] [**-damage** | **+damage**] [**-evie** | **+evie**] [**-f** *Number*] [**-fc** *Font*] [**-fixes** | **+fixes**] [**-fn** *Font*] [**-fp** *Font*] [**-help**] [**-I**] [**-indirect** *HostName*] [**-layer** *#* [**:***Display*]] [**-logo** | **nologo**] [**-n :***Number*] [**-once**] [**-P** *RowColumn Display* }] [**-pbuffer** *level* [**:display** *name* | **:display** *number*]] [**-p** *Number*] [**-port** *PortNumber*] [**-query** *HostName*] [**-r** | **r**] [**+render** | **-render**][**-s** *Number*] [**-secIP** [*PermissionCode*]] [**-secLocal** [*PermissionCode*]] [**-secSMT** [*PermissionCode*]] [**-stereo** [**:***Display*]] [**-su**] [**-T**] [**-t** *Number*] [**-to** *Number*] [**-v**] [**-vfb**][**-wm**] [**-wp** *Color*] [**-wrap** | [**-wrapx**] [**-wrapy**]] [**-x** *ExtensionName*] [**-xkbdir** *Directory*] [**-xkbmap** *FileName*] [[**+**|**-**]**accessx**] [**-ar1** *Milliseconds*] [**-ar2** *Milliseconds*] [**-sp** *FileName*] [**+/- xinerama** *FileName*]

Description

The **X** command starts the X Server, a display server that runs on bitmapped terminals. The X Server distributes input and output requests to or from programs located on either the host system or systems connected to it through a network.

End an Enhanced X-Windows session by using the Ctrl+Alt+Backspace key sequence.

You can specify one or more display devices. If none are specified, the default is all. The default configuration order is determined by the adapter slot order. The adapter in the first slot is initialized as the left most screen, the adapter in the second slot is the next screen to the right. To rearrange the layout of the screens, use the **-P** flag. The **-P** flag associates the row and column of the device with the device name. You can determine the device name by using the **lsdisp** command.

The two displays are arranged either vertically or horizontally. The following example shows **-P** flags specifying a horizontal arrangement:

```
-P11 ppr0 -P12 ppr1
```

The 2 in the right position of the second **-P** flag indicates that the second monitor view is along the x-axis. This produces the horizontal arrangement:

```
         Display                       Display
                  1                             2
```

To see two monitors in a vertical arrangement, the **-P** flags should read:

```
-P11 ppr0 -P21 ppr1
```

The 2 in the first position indicates that the monitors are in a vertical configuration along the y-axis:

```
         Display
                  1
```

```
          Display
                    2
```

In the horizontal configuration, when a mouse is traveling from left to right in Display 1 and reaches the border of Display 1 and 2, the cursor continues into Display 2 at the same y-axis position. When it reaches the edge of Display 2 and the **-wrapx** flag is set, it appears at the leftmost edge of Display 1 in the same y-axis position. If the **-wrapx** flag is not set, the mouse stops at the far edge of Display 2.

In a vertical configuration, when the mouse is traveling from top to bottom in Display 1 and reaches the border of Display 1 and Display 2, the cursor continues into Display 2 at the same x-axis position. When the cursor reaches the bottom of the display 2 and the **-wrapy** flag is set, the cursor appears at the top edge of Display 1 in the same x-axis position. If the **-wrapy** flag is not set, the mouse stops at the bottom of Display 2.

In addition, information and error messages (for example, a message indicating that an extension not able to load) are listed in the **/tmp/xlogfile** file. This file can provide useful information in cases when the X Server encounters a problem. This file is re-written every time the X Server is instantiated. This file provides additional error and non-error information but is not a complete error log for the X Server.

When X-Server is started, it comes up using the default color class. Depending on the driver, the X-Server may default to using the PseudoColor or TrueColor class.

The PseudoColor class uses a colormap to display the colors on the screen. Many graphic adapters only support one hardware colormap. In this case, if the default color class is PseudoColor and an application is using a colormap that is different from the default colormap, incorrect colors may be displayed on the screen. Only the window that has focus will display the correct colors. It is advisable to run the X-Server in the TrueColor mode to prevent wrong colors from being displayed on the screen.

The **-cc X-Server** command flag can be used to bring the X-Server up using the TrueColor class.
The **/usr/lpp/X11/defaults/xserverrc** file can be modified to allow this as shown in the following example.

As a root user, edit the **/usr/lpp/X11/defaults/xserverrc** file. Update the **EXTENSIONS=""** variable as shown in the following example:

```
#-----------------------------------------------------------------------
# Start the X server in True Color mode
#-----------------------------------------------------------------------
EXTENSIONS="$EXTENSIONS -cc 4"
```

Restart X Server by logging out of CDE and clicking reset.

Note: The **xdpyinfo** command can be used to verify the default color class.

Flags

Item	Description
-a *Number*	Specifies the acceleration multiplier for mouse movement. For example, a value of 5 causes the cursor to move five times as fast as the mouse. The default is 4 pixels; any value specified must be a positive value greater than 0.
-auth *FileName*	Specifies to X the file from which to read the MIT (Massachusetts Institute of Technology) magic cookie.
-bc	Turns off backward compatibility with Enhanced X-Windows version 1.1.
+bc	Turns on backward compatibility with Enhanced X-Windows version 1.1. This is the default.

Item	Description
-bp *Color*	Specifies a black pixel color for the display. The default is display dependent.
-bs	Enables backing store support on all screens. Backing store support is disabled by default.
-c *Volume*	Specifies key click volume.
-cc *VisualType* [**:***Display*]	Specifies the type of visual to use for the root window of the screen specified by the display name. Not all visual types are available on all adapters at all depths. The **:***Display* parameter is optional, but useful when using the multihead option. The **:***Display* parameter is the name of the display as shown in the **lsdisp** command. If no display number or name is supplied, the specified visual is selected for all screens. To specify the visual type and depth for the default visual, use the **-cc** and **-d** flags, respectively. Values for the *VisualType* parameter are specified as a string or a number as follows: **String Numeric equivalent** **StaticGray** 0 **GrayScale** 1 **StaticColor** 2 **PseudoColor** 3 **TrueColor** 4 **DirectColor** 5
-co *File*	Sets the name of the red, green, and blue (RGB) color database. This is the default flag for the color database.
-D *File*	Specifies the full path name of the color definition database file. The default is **/usr/lib/X11/rgb**.
-d *Depth*[**:***Display*]	Specifies the root depth for the screen specified by the display name. Not all visual types will be available on all adapters at all depths. The **:***Display* parameter is optional, but useful when using the multihead option and must correspond to the values passed with the **-P** flag. The **:***Display* parameter is the name of the display as shown in the **lsdisp** command. In the absence of the **:***Display* parameter, the specified depth is selected for all the selected displays in the multihead option, as specified in the **-P** flag.
-damage	Disables the X Damage extension.
+damage	Enables the X Damage extension.
-evie	Disables the X Event Interception extension.
+evie	Enables the X Event Interception extension.

Item	Description
-f *Number*	Specifies the beep volume. The default is -1 or medium. The supported values are as follows: **Value** **Setting** **0** Off **1-33** Low **-1** or **34-66** Medium **67-100** High
-fc *Font*	Specifies the cursor font for cursor glyphs and cursor masks. The default depends on the operating system and the display.
-fixes	Disables the X Fixes extension.
+fixes	Enables the X Fixes extension.
-fn *Font*	Specifies the default text font. The default depends on the operating system and the display.
-fp *Font*	Specifies the font path.
-I	Causes all remaining command line arguments to be ignored. (Uppercase i)
-help	Prints a usage message.
-layer *#*[**:***Display*]	Specifies that the default visual should be in the *#* layer. The **:***Display* parameter is the name of the display as shown in the **lsdisp** command. Specifying this flag for an adapter that does not have overlays, or has less than 8 bits of overlay, has no effect. Specifying this flag with a *#* higher than the number of supported layers results in the default visual residing in the default layer of the screen (as if no **-layer** flag had been used).
-logo	Turns on the X Window System logo display in the screen saver. There is currently no way to change this from a client.
-n :*Number*	Specifies the connection number. Valid values for the *Number* parameter are 0 to 255. The default is the next available number. The *Number* parameter is used by programs to communicate with a specific X Server. For example, the command: `X -n :18` specifies that communication to the activated X Server takes place by unix:18 or by *Hostname*:18.
-nobs	Disables backing store support on all screens. This is the default.
nologo	Turns off the X Window System logo display in the screen saver. There is currently no way to change this from a client.
-once	Instructs the server to exit after the first session ends. Normally, the server starts sessions automatically.

Item	Description
-P*RowColumn Display*	Specifies the physical positioning of the displays in a multihead configuration. The *Row* parameter indicates the row in which the display is located. The *Column* parameter indicates the column in which the display is located. The *Display* parameter is the device name of the display as shown in the first column of output from the **Isdisp** command. The first **-P***RowColumn Display* occurrence on the command line describes screen 0 to the X Server, the second describes screen 1, and so on. The **-P** flag is for use with multiple head support.
-pbuffer *level* [**:display** *name* | **:display** *number*]	Specifies the **pbuffer** memory allocation level for the screen specified by **:display**. This flag is only useful when used in conjunction with the GLX extension. The *level* parameter indicates the relative amount of frame buffer memory to be reserved for pbuffers. Specified values must be in the range of [0..2]. A value of 0 indicates that no memory should be reserved for pbuffers. A value of 1 indicates that a low amount of memory should be reserved. A value of 2 indicates that a high amount of memory should be reserved. Not all adapters support pbuffers. For those that do, not all screen configurations support pbuffers. The actual amount of frame buffer memory reserved for pbuffers is device-dependent, and may be influenced by other factors, such as screen resolution or default pixel depth. The **:display** parameter is optional, but useful when using the multihead option. The **:display** parameter is the name of the display as shown in the **lsdisp** command. If no display *number* or *name* is supplied, the specified **pbuffer** width is selected for all screens.
-p *Number*	Specifies the time interval, in minutes, between changes of the X Window System logo position. This flag is used with the **-s** (screen saver timeout) flag to control the blanking of the screen.
-r	Disables autorepeat. The default is autorepeat enabled.
r	Turns on autorepeat.
+render	Enables the X Render extension. By default, the X Render extension is disabled. **Note:** X Render Extension can be activated on the X Server only with GXT135P, GXT145, GXT4500P, and GXT6500P graphics adapters. To check the available adapter on the system, run the **lsdisp** command. Use the **-vfb** flag with the virtual frame buffer along with the **+render** flag.

Item	Description
-render	Disables the X Render extension.
-s *Number*	Specifies the number of minutes to wait before blanking the screen. The default is 10 minutes. If this value is set to 0, the screen-saver is disabled.
-secIP [*PermissionCode*]	Sets local access control on the internet socket. The *PermissionCode* is 3 octal digits which can set read, write, and execute bits. If no *PermissionCode* is specified after a security flag, then permission is defaulted to 0 for that socket.
-secLocal [*PermissionCode*]	Sets access control on the unix socket. The *PermissionCode* is 3 octal digits which can set read, write, and execute bits. If no *PermissionCode* is specified after a security flag, then permission is defaulted to 0 for that socket.
-secSMT [*PermissionCode*]	Sets access control on the shared memory transport socket. The *PermissionCode* is 3 octal digits which can set read, write, and execute bits. If no *PermissionCode* is specified after a security flag, then permission is defaulted to 0 for that socket.
-stereo [**:***Display*]	Configures the graphics adapter for optimum stereo support for the screen specified by *Display*. Supported screens will configure the adapter to provide the best available support for stereo. This may decrease other resources such as texture memory. The actual amount of memory affected is device-dependent, and may be influenced by other factors, such as screen resolution or default pixel depth. The *Display* parameter is optional, but useful when using the multihead option. The *Display* parameter is the name of the display as shown in the **lsdisp** command. If no display number or name is supplied, the **-stereo** flag pertains to all supported screens. Unsupported screens will ignore the **-stereo** flag.
-su	Disables save under support on all screens.
-T	Disables the Ctrl+Alt+Backspace key sequence that, by default, ends the AIXwindows session and all windows opened from it.
-t *Number*	Specifies the mouse threshold. The default is 2 pixels. Acceleration takes effect only if the mouse is moved beyond the mouse threshold in one time interval and only applies to the amount beyond the threshold.
-to *Number*	Specifies the number of minutes to elapse between connection checks. The default is 60 minutes. A specified value must be greater than 0.

Item	Description
-v	Specifies that the display be replaced with the current background color after the time specified by the **-s** flag expires. By default, if the **-v** flag is not used, the entire display is painted with the background tile after the time specified by the **-s** flag expires.
-vfb	Starts the X Server with Virtual Frame Buffer (VFB), without initializing any graphics adaptor.
-wm	Forces the default backing store of all windows to have the **WhenMapped** value. This is a convenient way of applying backing store to all windows.
-wp *Color*	Specifies a white pixel display color. The default depends on the display.
-wrap	Specifies the behavior of the mouse when its hotspot reaches the left or right border or the top or bottom of any root window. If this flag is set and the hotspot of the mouse reaches the left border of the leftmost root window, the mouse is automatically positioned at the right border of the rightmost root window at the same y position. Conversely, if this flag is set and the hotspot of the mouse reaches the right border of the rightmost root window, the mouse is automatically positioned at the left border of the leftmost root window at the same y position. If this flag is not set, the mouse stops at the left or right border of any root window. If this flag is set and the hotspot of the mouse reaches the top border of the topmost root window, the mouse is positioned at the bottom border of the bottommost root window at the same x position. Conversely, if this flag is set and the hotspot of the mouse reaches the bottom border of the bottommost root window, the mouse is positioned at the top border of the topmost root window at the same x position. The **-wrap** flag is for use with multiple head support.

Item	Description
-wrapx	Specifies the behavior of the mouse when its hotspot reaches the left or right border of any root window. If this flag is set and the hotspot of the mouse reaches the left border of the leftmost root window, the mouse is positioned at the right border of the rightmost root window at the same y position. Conversely, if this flag is set and the hotspot of the mouse reaches the right border of the rightmost root window, the mouse is positioned at the left border of the leftmost root window at the same y position. If this flag is not set, the mouse stops at the left or right border of any root window. The **-wrapx** flag is for use with multiple head support.
-wrapy	Specifies the behavior of the mouse when its hotspot reaches the top or bottom border of any root window. If this flag is set and the hotspot of the mouse reaches the top border of the topmost root window, the mouse is positioned at the bottom border of the bottommost root window at the same x position. Conversely, if this flag is set and the hotspot of the mouse reaches the bottom border of the bottommost root window, the mouse is positioned at the top border of the topmost root window at the same x position. If this flag is not set, the mouse stops at the top or bottom border of any root window. The **-wrapy** flag is for use with multiple head support.
-x *ExtensionName*	Specifies that the extension name should be loaded when the server is initialized. This is particularly useful for large extensions, such as the Display PostScript Level 2 (**dps**). This flag can be specified more than once with multiple extension names.
-query *HostName*	Enables Enhanced X-Windows Display Manager Control Protocol (**XDMCP**) and sends a **Query** packet to the specified host. The **-query** flag is for use with **XDMCP**.
-broadcast	Enables **XDMCP** and broadcasts **BroadcastQuery** packets to the network. The first responding display manager is chosen for the session. The **-broadcast** flag is for use with **XDMCP**.
-indirect *HostName*	Enables **XDMCP** and sends **IndirectQuery** packets to the specified host. The **-indirect** flag is for use with **XDMCP**.

Item	Description
-port *PortNumber*	Specifies an alternative port number for **XDMCP**. This flag must be specified before any **-query**, **-broadcast**, or **-indirect** flags. Normally, the server starts sessions one after another. This flag causes the server to exit after the first session ends. The **-port** flag is for use with **XDMCP**.
-class *DisplayClass*	Sets the value for an additional display qualifier used by **XDMCP** in resource lookup for display-specific options. The **-class** flag is for use with **XDMCP**.
-cookie *XDMAuthenticationBits*	Specifies a private key to be shared between the server and the manager when testing XDM-AUTHENTICATION-1. The **-cookie** flag is for use with **XDMCP**.
-displayID *DisplayID*	Allows the display manager to identify each display so that it can locate the shared key specified by the **-cookie** flag. The **-displayID** flag is for use with **XDMCP**.
+/- xinerama	Enable/Disable panoramic screen or Virtual Large Screen (VLS). Allows users to treat all heads in a multihead environment as a large screen.

Xkeyboard Flags

Item	Description
-xkbdir *Directory*	Specifies the base directory for the keyboard layout files.
-xkbmap *FileName*	Specifies the keyboard description to load on startup.
[**+**|**-**]**accessx**	Enables (**+**) or disables (**-**) AccessX key sequences.
-ar1 *Milliseconds*	Sets the length of time in milliseconds that a key must be pressed before autorepeat starts.
-ar2 *Milliseconds*	Sets the length of time in milliseconds that should elapse between autorepeat generated keystrokes.

Security Extension Flags

Item	Description
-sp *FileName*	Causes the server to attempt to read and interpret *FileName* as a security policy file with the format described below. The file is read at server startup and reread at each server reset. The syntax of the security policy file is as follows. Notation: "*" means zero or more occurrences of the preceding element, and "+" means one or more occurrences. To interpret *foo***/***bar*, ignore the text after the /; it is used to distinguish between instances of *foo* in the next section. *policy file* ::= *version line other line** *version line* ::= *string/v* '\n' *other line* ::= *comment* | *access rule* | *site policy* | *blank line* *comment* ::= *# not newline** '\n' *blank line* ::= *space* '\n' *site policy* ::= sitepolicy *string/sp* '\n' *access rule* ::= property *property/ar window perms* '\n' *property* ::= *string* *window* ::= any | root | *required property* *required property* ::= *property/rp* | *property with value* *property with value* ::= *property/rpv = string/rv* *perms* ::= [*operation* | *action* | *space*]* *operation* ::= r | w | d *action* ::= a | i | e *string* ::= *dbl quoted string* | *single quoted string* | *unquoted string* *dbl quoted string* ::= *space* " *not dqoute** " *space* *single quoted string* ::= *space* ' *not squote** ' *space* *unquoted string* ::= *space not space+ space* *space* ::= [' ' | '\t']* Character sets: *not newline* ::= any character except '\n' *not dqoute* ::= any character except " *not squote* ::= any character except ' *not space* ::= any character except those in *space*

Item	Description

The semantics associated with the previously described syntax are as follows.

version line
The first line in the file, specifies the file format version. If the server does not recognize the version *string***/v**, it ignores the rest of the file. The version string for the file format described here is `version-1`.

Once past the *version line*, lines that do not match the above syntax are ignored.

comment
Lines are ignored.

sitepolicy
Lines are currently ignored. They are intended to specify the site policies used by the XC-QUERY-SECURITY-1 authorization method.

access rule
Lines specify how the server should react to untrusted client requests that affect the X Window property named *property***/ar**. The rest of this section describes the interpretation of an *access rule*.

For an *access rule* to apply to a given instance of *property***/ar**, *property***/ar** must be on a window that is in the set of windows specified by *window*. If *window* is **any**, the rule applies to *property***/ar** on any window. If *window* is **root**, the rule applies to *property***/ar** only on root windows.

If *window* is *required property*, the following apply. If *required property* is a *property***/rp**, the rule applies when the window also has that *property***/rp**, regardless of its value. If *required property* is a *property with value*, *property***/rpv** must also have the value specified by *string***/rv**. In this case, the property must have type STRING and format 8, and should contain one or more null- terminated strings. If any of the strings match *string***/rv**, the rule applies.

The definition of string matching is simple case-sensitive string comparison with one elaboration: the occurrence of the character '*' in *string***/rv** is a wildcard meaning "any string." A *string***/rv** can contain multiple wildcards anywhere in the string. For example, x* matches strings that begin with **x**, *x matches strings that end with **x**, *x* matches strings containing **x**, and x*y* matches strings that start with **x** and subsequently contain **y**.

There may be multiple *access rule* lines for a given *property***/ar**. The rules are tested in the order that they appear in the file. The first rule that applies is used.

Item **Description**

perms
Specify operations that untrusted clients may attempt, and the actions that the server should take in response to those operations.

operation
Can be **r** (read), **w** (write), or **d** (delete). The following information shows how X Protocol property requests map to these operations in the X Consortium server implementation.

GetProperty
r, or **r** and **d** if delete = True

ChangeProperty
w

RotateProperties
r and **w**

DeleteProperty
d

ListProperties
none, untrusted clients can always list all properties

action
Can be **a** (allow), **i** (ignore), or **e** (error).

Allow
Executes the request as if it had been issued by a trusted client.

Ignore
Treats the request as a no-op. In the case of GetProperty, ignore means return an empty property value if the property exists, regardless of its actual value.

Error
Specifies not to execute the request and return a `BadAtom` error with the `atom` set to the property name. Error is the default action for all properties, including those not listed in the security policy file.

An *action* applies to all *operation*s that follow it, until the next *action*> is encountered. Thus, `irwad` means ignore read and write, allow delete.

GetProperty and RotateProperties might do multiple operations (**r** and **d**, or **r** and **w**). If different actions apply to the operations, the most severe action is applied to the whole request; there is no partial request execution. The severity ordering is: allow < ignore < error. Thus, if the *perms* for a property are `ire`d (ignore read, error delete), and an untrusted client attempts GetProperty on that property with `delete = True`, an error is returned, but the property value is not. Similarly, if any of the properties in a RotateProperties do not allow both read and write, an error is returned without changing any property values.

Item	Description

An example a security policy file follows:

```
version-1

# Allow reading of application resources, but not writing.
property RESOURCE_MANAGER     root      ar iw
property SCREEN_RESOURCES     root      ar iw

# Ignore attempts to use cut buffers.  Giving errors causes apps to crash,
# and allowing access may give away too much information.
property CUT_BUFFER0          root      irw
property CUT_BUFFER1          root      irw
property CUT_BUFFER2          root      irw
property CUT_BUFFER3          root      irw
property CUT_BUFFER4          root      irw
property CUT_BUFFER5          root      irw
property CUT_BUFFER6          root      irw
property CUT_BUFFER7          root      irw

# If you are using Motif, you probably want these.

property _MOTIF_DEFAULT_BINDINGS        rootar iw
property _MOTIF_DRAG_WINDOW   root      ar iw
property _MOTIF_DRAG_TARGETS  any       ar iw
property _MOTIF_DRAG_ATOMS    any       ar iw
property _MOTIF_DRAG_ATOM_PAIRS         any ar iw

# The next two rules let xwininfo -tree work when untrusted.
property WM_NAME              any       ar
```

```
# Allow read of WM_CLASS, but only for windows with WM_NAME.
# This might be more restrictive than necessary, but demonstrates
# the required property facility, and is also an attempt to
# say "top level windows only."
property WM_CLASS             WM_NAME   ar

# These next three let xlsclients work untrusted.  Think carefully
# before including these; giving away the client machine name and command
# may be exposing too much.
property WM_STATE             WM_NAME   ar
property WM_CLIENT_MACHINE    WM_NAME   ar
property WM_COMMAND           WM_NAME   ar

# To let untrusted clients use the standard colormaps created by
# xstdcmap, include these lines.
property RGB_DEFAULT_MAP      root      ar
property RGB_BEST_MAP         root      ar
property RGB_RED_MAP          root      ar
property RGB_GREEN_MAP        root      ar
property RGB_BLUE_MAP         root      ar
property RGB_GRAY_MAP         root      ar
```

```
# To let untrusted clients use the color management database created
# by xcmsdb, include these lines.
property XDCCC_LINEAR_RGB_CORRECTION    rootar
property XDCCC_LINEAR_RGB_MATRICES      rootar
property XDCCC_GRAY_SCREENWHITEPOINT    rootar
property XDCCC_GRAY_CORRECTION          rootar


# oddball property names and explicit specification of error conditions
property "property with spaces"         'property with "'aw er ed

# Allow deletion of Woo-Hoo if window also has property OhBoy with value
# ending in "son".  Reads and writes will cause an error.
property Woo-Hoo              OhBoy = "*son"ad
```

Example

To start the X Server with X Render extension, enter the following command:

```
$X -T -force :0 -vfb -d 32 +render
```

In this example, the X Server will use the Virtual Frame Buffer (VFB) for rendering instead of using the physical graphics adaptor.

x_add_fs_fpe Command

Purpose

Adds a network font server to a font path.

Syntax

x_add_fs_fpe *Host Port Position TypeName*

Description

The **x_add_fs_fpe** command adds a font path element to the font path of the selected network type name for a font server to access fonts.

Item	Description
Host	Specifies the name of the system where the font server resides.
Port	Specifies the number of the font server port. This number must be in the **/etc/services** file and specified in decimal.
Position	Specifies where to insert this element in the font path.
TypeName	Specifies the name of the network type. Each network type has a font path consisting of one or more font path elements. Specify the name of the network type to which the font path element will be added, or choose to have it added to all network type names by specifying `All`. If a font path element is added to `All` network types, will be placed at the end of each font path.

Security

Access Control: Only the root user should have execute (x) access to this command.

Example

To add the font server to the start of the font path for network type `x_st_mgr.ether`, enter:

```
x_add_fs_fpe winter 7500 1 x_st_mgr.ether
```

In this example, the font server on host `winter` has been added to the start of the font path for network type `x_st_mgr.ether`. The font server port is `7500`.

Files

Item	Description
/usr/lpp/x_st_mgr/bin/x_add_fs_fpe	Contains the **x_add_fs_fpe** command.
/etc/x_st_mgr/ether.cf	Contains the network type **x_st_mgr.ether** configuration file (sample).

x_add_nfs_fpe Command

Purpose

Adds a NFS/TFTP accessed font directory to a font path.

Syntax

x_add_nfs_fpe *Host Directory Method Position TypeName*

Description

The **x_add_nfs_fpe** command adds a font path element to the font path of the selected network type name. This font directory will be accessed using Network File System (NFS) or Trivial File Transfer Protocol (TFTP).

Item	Description
Host	Specifies the system name to access for the font directory.
Directory	Specifies the complete path to the directory that contains the fonts.
Method	Specifies either `nfs` or `tftp` to be used to access the fonts.
Position	Specifies where to insert this element in the font path.
TypeName	Specifies the name of the network type. Each network type has a font path consisting of one or more font path elements. Specify the name of the network type to which the font path element will be added, or choose to have it added to all network type names by specifying `All`. If a font path element is added to `All` network types, it will be placed at the end of each font path.

Security

Access Control: Only the root user should have execute (x) access to this command.

Example

To add the fonts in `/usr/lib/X11/fonts/100dpi` to the network type `x_st_mgr.ether`, enter:

```
x_add_nfs_fpe cedar /usr/lib/X11/fonts/100dpi nfs Last \ x_st_mgr.ether
```

In this the font path element `/usr/lib/X11/fonts/100dpi` is added to the end of the font path for network type `x_st_mgr.ether`. The font directory is on the host `cedar`, which is accessed using NFS.

Files

Item	Description
/usr/lpp/x_st_mgr/bin/x_add_nfs_fpe	Contains the **x_add_nfs_fpe** command.
/etc/x_st_mgr/ether.cf	Contains the network type **x_st_mgr.ether** configuration file (sample).

x_rm_fpe Command

Purpose

Removes a font path element from a font path.

Syntax

x_rm_fpe *TypeName Position Method Host Post Directory*

Description

The **x_rm_fpe** command removes a font path element from the font path of the selected network type name.

Item	Description
TypeName	Specifies from which network type name the element is to be removed.
Position	Specifies where the element is in the font path.
Method	Specifies the method used to access the font path element. The valid options are: **tcp** for Network Font Server; **default** for initial default font path element; **nfs** for NFS; and **tftp** for TFTP.
Host	Specifies the name of the system specified in the font path element. For elements using the default method, specify **None**.
Port	Specifies the number of the server port specified in the font path element. For elements using the `nfs` or `tftp` method, specify **None**.
Directory	Specifies the complete path to the directory that contains the fonts. For a Network Font Server element, specify **None**.

Security

Access Control: Only the root user should have execute (x) access to this command.

Examples

To remove the font element `/usr/lib/X11/fonts/100dpi` from the font path for network type `x_st_mgr.ether`, enter:

```
x_rm_fpe x_st_mgr.ether 3 nfs waco None /usr/lib/X11/fonts/100dpi
```

In this example, the font path element `/usr/lib/X11/fonts/100dpi` that is accessed on host `waco` using NFS has been removed from the third position of the font path for network type `x_st_mgr.ether`. Because a port number is not used for NFS, this parameter was set to None.

Files

Item	Description
/usr/lpp/x_st_mgr/bin/x_rm_fpe	Contains the **x_rm_fpe** command.
/etc/x_st_mgr/ether.cf	Contains the network type **x_st_mgr.ether** configuration file (sample).

xargs Command

Purpose

Constructs parameter lists and runs commands.

Syntax

xargs [**-p t x**] [**-e** [*EOFString*]] [**-E** *EOFString*] [**-i** [*ReplaceString*]] [**-I** *ReplaceString* | **-L** *Number* | **-n** *Number*] [**-l** [*Number*]] [**-s** *Size*] [*Command* [*Argument* ...]]

Note: Do not put a blank space between the lowercase flags and the parameter.

Description

The generated command line length is the sum of the size, in bytes, of the *Command* and each *Argument* treated as strings, including a null byte terminator for each of these strings. The **xargs** command limits the command line length. When the constructed command line runs, the combined *Argument* and environment lists can not exceed **ARG_MAX** bytes. Within this constraint, if you do not specify the **-n** or the **-s** flags, the default command line length is at least the value specified by **LINE_MAX**.

Flags

Item	Description
-e[*EOFString*]	Obsolete flag. Use the **-E** flag. Uses the *EOFString* parameter as the logical EOF string. If you do not specify the **-e** or the **-E** flags, underscore (_) is assumed for the logical EOF string. If you do not specify the *EOFString* parameter, the logical EOF string capability is disabled, and underscores are taken literally. The **xargs** command reads from standard input until either EOF or the specified string is reached.
-E *EOFString*	Specifies a logical EOF string to replace the default underscore (_). The **xargs** command reads standard input until either EOF or the specified string is reached.
-i [*ReplaceString*]	Obsolete flag. Use the **-I** (uppercase i) flag. If you do not specify the *ReplaceString* parameter, the string {} is used. **Note:** The **-I** (uppercase i), **i**, **-L** (uppercase l) , **l**, and **-n** flags are mutually exclusive. The last flag specified takes effect.
-I *ReplaceString*	(Uppercase i). Inserts each line of standard input as an argument for the *Command* parameter, inserting it in *Argument* for each occurrence of *ReplaceString*. *ReplaceStrings* can not be used in more than 5 arguments. Blank characters at the beginning of each standard input line are ignored. Each *Argument* can contain one or more *ReplaceStrings,* but may not be larger than 255 bytes. The **-I** flag also turns on the **-x** flag. The -I (uppercase i) flag means -L1. Therefore, only one standard input line can be substituted as an argument at a time. If the replaced string appears more than once in the command parameter, the same standard input line is substituted for each occurrence of the replaced string. **Note:** The **-I** (uppercase i), **i**, **-L** (uppercase l) , **l**, and **-n** flags are mutually exclusive. The last flag specified takes effect.
-l [*Number*]	(Lowercase L). Obsolete flag. Use the **-L** flag. If you do not specify the *Number* parameter, a value of 1 is used. The **-l** flag also turns on the **-x** flag. **Note:** The **-I** (uppercase i), **i**, **-L** (uppercase l), **-l**, and **-n** flags are mutually exclusive. The last flag specified takes effect.

Item	Description
-L *Number*	Runs the *Command* parameter with the specified number of nonempty parameter lines read from standard input. The last invocation of the *Command* parameter can have fewer parameter lines if fewer than the specified *Number* remain. A line ends with the first new-line character unless the last character of the line is a space or a tab. A trailing space indicates a continuation through the next nonempty line. Reads the *Number* lines from the standard input and places them at the end of the command line. **Note:** The **-I** (uppercase i), **i**, **-L** (uppercase l), **-l**, and **-n** flags are mutually exclusive. The last flag specified takes effect.
-n *Number*	Runs the *Command* parameter using as many standard input arguments as possible, up to the maximum specified by the *Number* parameter. Reads the maximum of *Number* arguments from the standard input and places them at the end of the command line. The **xargs** command uses fewer arguments if: • The accumulated command line length exceeds the bytes specified by the **-s** *Size* flag. • The last iteration has fewer value than the value specified by the *Number* argument, but not zero, arguments remaining. **Note:** The **-I** (uppercase i), **i**, **-L** (uppercase l), **-l**, and **-n** flags are mutually exclusive. The last flag specified takes effect.
-p	Asks whether to run the *Command* parameter. It displays the constructed command line, followed by a ?**. . .** (question mark, ellipsis) prompt. Enter an affirmative response specific to the locale to run the *Command* parameter. Any other response causes the **xargs** command to skip that particular invocation of the parameter. You are asked about each invocation. The **-p** flag also turns on the **-t** flag.
-s *Size*	Sets the maximum total size of the constructed *Command* line. The *Size* parameter must be a positive integer. Fewer arguments are used if: 1. The total number of arguments exceeds those specified by the **-n** flag. 2. The total number of lines exceeds those specified by the **-L** or **-l** (Lowercase L) flags. 3. EOF is reached before the number of bytes specified by the *Size* parameter are accumulated.
-t	Enables the trace mode and echoes the constructed *Command* line to standard error before running.
-x	Stops running the **xargs** command if any *Command* line is greater than the number of bytes specified by the **-s** *Size* flag. This **-x** flag is turned on if you specify either the **-I** (Uppercase i) or **-l** (Lowercase L) flag. If you do not specify the **-i**, **-I** (Uppercase i), **-l** (Lowercase L), **-L**, or**-n** flag, the total length of the *Command* line must be within the limit specified by the -s *Size* flag.

Exit Status

This command returns the following exit values:

Item	Description
0	All invocations of the *Command* parameter returned exit status 0.
1-125	A command line meeting the specified requirements could not be assembled, one or more of the invocations of the *Command* parameter returned a non-zero exit status, or some other error occurred.
126	*Command* was found but could not be invoked.
127	*Command* could not be found.

If a command line meeting the specified requirements cannot be assembled, the command cannot be invoked, an invocation of the command is terminated by a signal, or an invocation of the command exits with exit status 255. The **xargs** command will write a diagnostic message and exit without processing any remaining input.

Examples

1. To use a command on files whose names are listed in a file, type:

   ```
   xargs lint -a <cfiles
   ```

   If the `cfiles` file contains the following text:

   ```
   main.c readit.c
   gettoken.c
   putobj.c
   ```

   the **xargs** command constructs and runs the following command:

   ```
   lint -a main.c readit.c gettoken.c putobj.c
   ```

   If the `cfiles` file contains more file names than fit on a single shell command line (up to **LINE_MAX**), the **xargs** command runs the **lint** command with the file names that fit. It then constructs and runs another **lint** command using the remaining file names. Depending on the names listed in the `cfiles` file, the commands might look like the following:

   ```
   lint -a main.c readit.c gettoken.c . . .
   lint -a getisx.c getprp.c getpid.c . . .
   lint -a fltadd.c fltmult.c fltdiv.c . . .
   ```

   This command sequence is not quite the same as running the **lint** command once with all the file names. The **lint** command checks cross-references between files. However, in this example, it cannot check between the `main.c` and the `fltadd.c` files, or between any two files listed on separate command lines.

   For this reason you may want to run the command only if all the file names fit on one line. To specify this to the **xargs** command use the **-x** flag by typing:

   ```
   xargs -x lint -a <cfiles
   ```

   If all the file names in the `cfiles` file do not fit on one command line, the **xargs** command displays an error message.

2. To construct commands that contain a certain number of file names, type:

   ```
   xargs -t -n 2 diff <<EOF
   starting chap1 concepts chap2 writing
   chap3
   EOF
   ```

   This command sequence constructs and runs **diff** commands that contain two file names each (**-n 2**):

```
diff starting chap1
diff concepts chap2
diff writing chap3
```

The **-t** flag causes the **xargs** command to display each command before running it, so you can see what is happening. The <<E0F and E0F pattern-matching characters define a *here document*, which uses the text entered before the end line as standard input for the **xargs** command.

3. To insert file names into the middle of command lines, type:

   ```
   ls | xargs -t -I {} mv {} {}.old
   ```

   This command sequence renames all files in the current directory by adding `.old` to the end of each name. The **-I** flag tells the **xargs** command to insert each line of the **ls** directory listing where {} (braces) appear. If the current directory contains the files `chap1`, `chap2`, and `chap3`, this constructs the following commands:

   ```
   mv chap1 chap1.old
   mv chap2 chap2.old
   mv chap3 chap3.old
   ```

4. To run a command on files that you select individually, type:

   ```
   ls | xargs -p -n 1 ar r lib.a
   ```

   This command sequence allows you to select files to add to the `lib.a` library. The **-p** flag tells the **xargs** command to display each **ar** command it constructs and to ask if you want to run it. Type y to run the command. Press the any other key if you do not want to run the command.

   Something similar to the following displays:

   ```
   ar r lib.a chap1 ?...
   ar r lib.a chap2 ?...
   ar r lib.a chap3 ?...
   ```

5. To construct a command that contains a specific number of arguments and to insert those arguments into the middle of a command line, type:

   ```
   ls | xargs -n6 | xargs -I{} echo {} - some files in the directory
   ```

   If the current directory contains files chap1 through chap10, the output constructed will be the following:

   ```
   chap1 chap2 chap3 chap4 chap5 chap6 - some files in the directory
   chap7 chap8 chap9 chap10 - some files in the directory
   ```

File

Item	Description
/usr/bin/xargs	Contains the **xargs** command.

xauth Command

Purpose

Edits and displays the authorization information used in connecting to the X server.

Syntax

xauth [**-f** *AuthFile*] [**-v** | **-q**] [**-i**] [**-b**] [*CommandArgument ...*]

Description

The **xauth** command is usually used to edit and display the authorization information used in connecting to the X server. This program extracts authorization records from one machine and merge them into another (for example, when using remote logins or granting access to other users).

The following commands can be entered interactively, on the **xauth** command line, or in scripts. Note that this program does not contact the X server.

Item	Description
add *DisplayName ProtocolName Hexkey*	An authorization entry is added to the authorization file for the indicated display using the given protocol and key data. The data is specified as an even-length string of hexadecimal digits, each pair representing one octet. The first digit of each pair gives the most significant 4 bits of the octet, and the second digit of the pair gives the least significant 4 bits. For example, a 32-character hexkey would represent a 128-bit value. A protocol name consisting of just a single period is treated as an abbreviation for **MIT-MAGIC-COOKIE-1**.
extract *FileName DisplayName...*	Authorization entries for each of the specified displays are written to the indicated file. The extracted entries can be read back in using the **merge** and **nmerge** commands. If the file name consists of just a single dash, the entries are written to the binary output.

Item	Description
generate *DisplayName ProtocolName* [*trusted* | *untrusted*] [*timeout seconds*] [*group group-id*] [*data hexdata*]	This command is similar to **add**. The main difference is that instead of requiring the user to supply the key data, it connects to the server specified in *displayname* and uses the SECURITY extension in order to get the key data to store in the authorization file. If the server cannot be contacted or if it does not support the SECURITY extension, the command fails. Otherwise, an authorization entry for the indicated display using the given protocol is added to the authorization file. A protocol name consisting of just a single period is treated as an abbreviation for MIT-MAGIC-COOKIE-1. If the trusted option is used, clients that connect using this authorization will have full run of the display, as usual. If untrusted is used, clients that connect using this authorization will be considered untrusted and prevented from stealing or tampering with data belonging to trusted clients. See the SECURITY extension specification for full details on the restrictions imposed on untrusted clients. The default is untrusted. The timeout option specifies how long in seconds this authorization will be valid. If the authorization remains unused (no clients are connected with it) for longer than this time period, the server purges the authorization, and future attempts to connect using it will fail. Note that the purging done by the server does not delete the authorization entry from the authorization file. The default timeout is 60 seconds. The group option specifies the application group that clients connecting with this authorization should belong to. See the application group extension specification for more details. The default is to not belong to an application group. The data option specifies data that the server should use to generate the authorization. Note that this is not the same data that gets written to the authorization file. The interpretation of this data depends on the authorization protocol. The *hexdata* is in the same format as the *hexkey* described in the **add** command. The default is to send no data.
list [*DisplayName*...]	Authorization entries for each of the specified displays (or all displays if none are named) are printed on the standard output in a textual format. Key data is always displayed in the hexadecimal format given in the description of the **add** command.

Item	Description
merge [*FileName*...]	Authorization entries are read from the specified files and are merged into the authorization database, superseding any matching existing entries. If a file name consists of just a single dash, the binary input is read if it has not been read before.
[n]extract *Filename DisplayName*...	Authorization entries for each of the specified displays are written to the indicated file. The entries are written in a numeric format suitable for non-binary transmission (such as secure electronic mail). The extracted entries can be read back in using the **merge** and **nmerge** commands. If the file name consists of just a single dash, the entries are written to the standard output.
[n]list [*DisplayName*...]	Authorization entries for each of the specified displays (or all displays if none are named) are printed on the standard output in the numeric format used by the **nextract** command. Key data is always displayed in the hexadecimal format given in the description of the **add** command.
[n]merge [*FileName*...]	Authorization entries are read from the specified files and are merged into the authorization database, superseding any matching existing entries. The numeric format given in the description of the **extract** command is used. If a file name consists of just a single dash, the standard input is read if it has not been read before.
remove *DisplayName*...	Authorization entries matching the specified displays are removed from the authority file.
source *FileName*	The specified file is treated as a script containing **xauth** commands to execute. Blank lines and lines beginning with a # (pound sign) are ignored. A single dash can be used to indicate the standard input, if it has not already been read.
info	Information describing the authorization file, whether or not any changes have been made, and from where **xauth** commands are being read is printed on the standard output.
exit	If any modifications have been made, the authority file is written out (if allowed), and the program exits. An end of file is treated as an implicit exit command.
quit	The program exits, ignoring any modifications. This may also be accomplished by pressing the interrupt character.
help [*String*]	A description of all commands that begin with the given string (or all commands if no string is given) is printed on the standard output.
?	A short list of the valid commands is printed on the standard output.

Display names for the **add**, [**n**]**extract**, [**n**]**list**, [**n**]**merge**, and **remove** commands use the same format as the **DISPLAY** environment variable and the common *display* command-line argument. Display-specific information (such as the screen number) is unnecessary and is ignored. Same-machine connections (such as local-host sockets, shared memory, and the Internet Protocol *HostName LocalHost*) are referred to as `HostName/unix:DisplayNumber` so that local entries for different machines can be stored in one authority file.

> **Note:** Users that have unsecure networks should take care to use encrypted file transfer mechanisms to copy authorization entries between machines. Similarly, the MIT-MAGIC-COOKIE-1 protocol is not very useful in unsecure environments. Sites that are interested in additional security may need to use encrypted authorization mechanisms such as Kerberos. Spaces are currently not allowed in the protocol name. Quoting could be added.

Flags

The following options are used with the **xauth** command. They can be given individually (for example, **-q -i**) or combined (for example, **-qi**).

Item	Description
-f *AuthFile*	Specifies the name of the authority file to use. By default, **xauth** uses the file specified by the **XAUTHORITY** environment variable or .xauthority in the user's home directory.
-v	Indicates that **xauth** should operate verbosely and print status messages indicating the results of various operations (for example, how many records have been read in or written out). This is the default if **xauth** is reading commands from its standard input and its standard output is directed to a terminal.
-q	Indicates that **xauth** should operate quietly and not print unsolicited status messages. This is the default if an **xauth** command is given on the command line or if the standard output is not directed to a terminal.
-i	Indicates that **xauth** should ignore any authority file locks. Normally, **xauth** refuses to read or edit any authority files that have been locked by other programs (usually **xdm** or another **xauth**).
-b	Indicates that **xauth** should attempt to break any authority file locks before proceeding. Use this option only to clean up stale locks.

Example

The most common use for the **xauth** command is to extract the entry for the current display, copy it to another machine, and merge it into the user's authority file on the remote machine:

```
% xauth extract \- $DISPLAY | rsh otherhost xauth merge \-
```

Files

Item	Description
$HOME/.Xauthority	Contains the default authority file if the **XAUTHORITY** environment variable is not defined.

xclock Command

Purpose

Continuously displays the current time of day.

Syntax

xclock [*-Xtoolkitoption ...*] [**-analog** | **-digital**] [**-chime**] [**-hd** *Color*] [**-help**] [**-hl** *Color*] [**-padding** *Number*] [**-update** *Seconds*]

Description

The **xclock** command gets the time from the system clock, then displays and updates it in the form of a digital or analog clock. Select the **-analog** or **-digital** flag to display the clock in analog or digital formats. You can also select flags to specify the presentation of the clock, including chime and update frequency, colors, and border width.

This command uses the Athena clock widget, which understands core resource names and classes. To specify these resources, you need to know the hierarchy of the widgets that comprise the **xclock** command. In the following example, the indented items indicate the hierarchical structure. The widget class name is given first, followed by the widget instance name:

```
XClock xclock
Clock clock
```

The following examples demonstrate the possible ways to specify resources for this client:

```
xclock.clock.background
```

```
XClock*background
```

```
xclock*background
```

> **Note:** Specifying resources as xclock.background which worked with the previous version of xclock will not work with this version.

Flags

Item	Description
-Xtoolkitoption	The **xclock** command accepts all of the standard X Toolkit command-line option flags in addition to the specific flags listed. You can view standard X Toolkit command-line option flag the in the **custom** command.
-analog	Sets the analog display mode, which is the default mode. Draws a conventional 12-hour clock face with ticks for each minute and stroke marks on each hour.
-chime	Specifies the sounding of a chime once on the half hour and twice on the hour.
-digital	Sets the 24-hour digital display mode. Displays the date and time in digital form.
-hd *Color*	Specifies the color of the hands in analog mode on color displays. The default is black.
-help	Prints a brief summary of the allowed options.
-hl *Color*	(lowercase HL) Specifies the highlight color of the edges of the hands of the analog clock. The default is black.
-padding *Number*	Specifies the width in pixels of the padding between the window border and the clock text or picture. The default is 8.

Item	Description
-update *Seconds*	Specifies the frequency in seconds that the **xclock** command updates its display. If the **xclock** window is obscured and then exposed, the **xclock** command redisplays immediately. The specification of an update frequency less than 30 seconds enables the second hand in the analog mode. The default update frequency is 60 seconds.

.Xdefaults Keywords

Use the following keywords to set the defaults for the **xclock** command.

Item	Description
analog (class Boolean)	Specifies an analog clock instead of a digital clock. The default is true.
chime (class Boolean)	Specifies whether a bell sounds on the hour and half hour.
fontSet (class FontSet)	Specifies the fontset for the digital clock. Variable-width fonts do not always display correctly.
foreground (class Foreground)	Specifies the color of tick marks on color displays. If **reverseVideo** is specified, the default is white, otherwise the default is black.
hands (class Foreground)	Specifies the color on the inside of the hands in the analog clock on color displays. If **reverseVideo** is specified, the default is white, otherwise the default is black.
highlight (class Foreground)	Specifies the color used to highlight the clock's hands. If **reverseVideo** is specified, the default is white, otherwise the default is black.
height (class Height)	Specifies the height of the clock. The default for the analog clock is 164 pixels. The default for the digital clock is whatever is required to hold the clock when displayed in the chosen font.
padding (class Margin)	Specifies the amount of internal padding in pixels. The default is 8.
update (class Interval)	Specifies the frequency in seconds in which the **xclock** command updates its display.
width (class Width)	Specifies the width of the clock. The default for the analog clock is 164 pixels. The default for the digital clock is whatever is needed to hold the clock when displayed in the chosen font.

Environment Variables

Item	Description
DISPLAY	Gets the default host and display number.
XENVIRONMENT	Gets the name of a resource file that overrides the global resources stored in the RESOURCE_MANAGER property.

Examples

1. To specify a digital clock display, enter:

```
xclock -digital
```

2. To specify red hands on an analog clock, enter:

```
xclock -hd red
```

File

Item	Description
/usr/lib/X11/app-defaults/XClock	Specifies the required resources.

xcmsdb Command

Purpose

Loads, queries, or removes Screen Color Characterization Data stored in properties on the root window of the screen.

> **Note:** The **xcmsdb** command is only supported in X11R5 (AIXwindows Version 1.2.3).

Syntax

xcmsdb [**-display** *Display*] [[**-query**] [**-remove**] [**-color**]] | [**-format 32** | **16** | **8**] [*FileName*]

Description

The **xcmsdb** command is used to load, query, or remove Screen Color Characterization Data stored in properties on the root window of the screen. Screen Color Characterization Data is an integral part of **Xlib**, which is necessary for proper conversion between device-independent and device-dependent color specifications. **Xlib** uses the **XDCCC_LINEAR_RGB_MATRICES** and **XDCCC_LINEAR_RGB_CORRECTION** properties to store color characterization data for color monitors. It uses **XDCCC_GRAY_SCREENWWHITEPOINT** and **XDCCC_GRAY_CORRECTION** properties for gray scale monitors. Because **Xlib** allows the addition of Screen Color Characterization Function Sets, added function sets may place their Screen Color Characterization Data on other properties. This utility is unaware of these other properties; therefore, you will need to use a similar utility provided with the function set, or use the example **xprop** utility.

The ASCII readable contents of the *FileName* parameter (or the standard input if no input file is given) are appropriately transformed for storage in properties, provided the **-query** or **-remove** flag options are not specified.

> **Note:** The Xcms API in **libX11.a** is supported; however, the client side color name data base, **/usr/lib/X11/Xcms.txt**, and a device color characterization file, **/usr/lib/X11/XcmsIBM5081.dcc**, are provided as unsupported samples.

Flags

Item	Description
-display*Display*	Specifies the server to which you are converting.
- query	Reads or attempts to read the XDCCC properties off the screen's root window. If successful, it transforms the data into a more readable format, and then sends the data to standard output.

Item	Description
-remove	Removes or attempts to remove the XDCCC properties on the screen's root window.
-color	Sets the **-query** and **-remove** options to only check for the **XDCCC_LINEAR_RGB_MATRICES** and **XDCCC_LINEAR_RGB_CORRECTION** properties. If the **-color** option is not set, the **-query** and **-remove** options check for all the properties.
-format 32 | **16** | **8**	Specifies the property format (32, 16, or 8 bits per entry) for the **XDCCC_LINEAR_RGB_CORRECTION** property. Precision of encoded floating-point values increases with the increase in bits per entry. The default is 32 bits per entry.

Parameter

Item	Description
FileName	Specifies the ASCII readable contents of a Screen Color Characterization Data file.

Examples

1. Use the following example to put Screen Color Characterization Data on the root window by telling the **xcmsdb** command to read it from a file:

```
xcmsdb /usr/lib/X11/XcmsIBM5081.dcc
```

2. Use the following example after you have already put Screen Color Characterization Data on the root window to tell the **xcmsdb** command to read the data back if it exists:

```
xcmsdb -query
```

xdm Command

Purpose

Manages a collection of X Displays with support for XDMCP.

Syntax

xdm [**-config** *ConfigurationFile*] [**-debug** *DebugLevel*] [**-nodaemon**] [**-error** *ErrorLogFile*] [**-resources** *ResourceFile*] [**-server** *ServerEntry*] [**-udpPort** *PortNumber*] [**-session** *SessionProgram*] [**-xrm** *ResourceSpecification*]

Description

The **xdm** (X Display Manager) command manages a collection of X displays, which may be on the local host or remote servers. The design of the **xdm** command was guided by the needs of X terminals as well as the X Consortium standard XDMCP, the *X Display Manager Control Protocol*. The **xdm** command provides services similar to those provided by the **init**, **getty**, and **login** commands on character terminals: prompting for login name and password, authenticating the user, and running a session.

A *session* is defined by the lifetime of a particular process; in the traditional character-based terminal world, it is the user's login shell. In the **xdm** context, it is an arbitrary session manager. This is because in a windowing environment, a user's login shell process does not necessarily have any terminal-like interface with which to connect. When a real session manager is not available, a window manager or terminal emulator is typically used as the *session manager*, meaning that ending this process ends the user's session.

When the session is ended, **xdm** resets the X server and (optionally) restarts the whole process.

When the **xdm** command receives an **Indirect** query by way of XDMCP, it can run a **chooser** process to perform an XDMCP **BroadcastQuery** (or an XDMCP Query to specified hosts) on behalf of the display and offer a menu of possible hosts that offer XDMCP display management. This feature is useful with X terminals that do not offer a host menu themselves.

Because the **xdm** command provides the first interface that users see, it is designed to be simple to use and easy to customize to the needs of a particular site.

Typical Usage

The **xdm** command is designed to operate in a wide variety of environments.

First, set up the **xdm** configuration file. Make a directory (usually **/usr/lib/X11/xdm**) to contain all of the relevant files. The following is a reasonable configuration file, which could be named **xdm-config**:

```
DisplayManager.servers:       /usr/lib/X11/xdm/Xservers
DisplayManager.errorLogFile:  /usr/lib/X11/xdm/xdm-errors
DisplayManager*resources:     /usr/lib/X11/xdm/Xresources
DisplayManager*startup:       /usr/lib/X11/xdm/Xstartup
DisplayManager*session:       /usr/lib/X11/xdm/Xsession
DisplayManager.pidFile:       /usr/lib/X11/xdm/xdm-pid
DisplayManager._0.authorize:  true
DisplayManager*authorize:     false
```

This file contains references to other files. Some of the resources are specified with an * (asterisk) separating the components. These resources can be made unique for each display by replacing the * (asterisk) with the display name, but typically this is not useful. See the Resources section on the next page for a complete discussion.

The first file, **/usr/lib/X11/xdm/Xservers**, contains the list of displays to manage that are not using **XDMCP**. Most workstations have only one display, numbered 0 (zero), so the file looks something like this:

```
:0 Local local /usr/bin/X11/X -force
```

This keeps **/usr/bin/X11/X** running on this display and manages a continuous cycle of sessions.

The **/usr/lib/X11/xdm/xdm-errors** file contains error messages from **xdm** and anything output to standard error by **Xsetup**, **Xstartup**, **Xsession** or **Xreset** scripts. If you have trouble starting the **xdm** command, check the **/usr/lib/X11/xdm/xdm-errors** file to see if the **xdm** command has any clues to the trouble.

The next configuration entry, **/usr/lib/X11/xdm/Xresources**, is loaded onto the display as a resource database using the **xrdb** command. As the authentication widget reads this database before starting, it usually contains parameters for that widget.

Flags

All of these options (except **-config**) specify values that can also be specified in the configuration file as resources.

Item	Description
-config *ConfigurationFile*	Names the configuration file, which specifies resources to control the behavior of the **xdm** command. The **/usr/lib/X11/xdm/xdm-config** file is the default.
-debug *DebugLevel*	Specifies the numeric value for the **DisplayManager.debugLevel** resource. A nonzero value causes **xdm** to print debugging statements to the terminal and disables the **DisplayManager.daemonMode** resource, forcing **xdm** to run synchronously. These error messages may be unclear. To interpret them, check the X11R4 source code for the **xdm** command.

Item	Description
-nodaemon	Specifies False as the value for the **DisplayManager.daemonMode** resource. This suppresses the usual daemon behavior, in which the **xdm** command closes all file descriptors, disassociates itself from the controlling terminal, and puts itself in the background when it first starts up.
-error *ErrorLogFile*	Specifies the value for the **DisplayManager.errorLogFile** resource. This file contains errors from **xdm** as well as anything written to standard error by the various scripts and programs run during the progress of the session.
-resources *ResourceFile*	Specifies the value for the **DisplayManager*resources** resource. This file is loaded using the **xrdb** command to specify configuration parameters for the authentication widget.
-server *ServerEntry*	Specifies the value for the **DisplayManager.servers** resource. See the section Server Specification for a description of this resource.
-udpPort *PortNumber*	Specifies the value for the **DisplayManager.requestPort** resource. This sets the port number that the **xdm** command monitors for **XDMCP** requests. **XDMCP** uses the registered well-known UDP port 177. Do not change this resource except when debugging.
-session *SessionProgram*	Specifies the value for the **DisplayManager*session** resource. This indicates the program to run as the session after the user has logged in.
-xrm *ResourceSpecification*	Allows an arbitrary resource to be specified, as in most X Toolkit applications.

Resources

At many stages, the actions of **xdm** can be controlled through the use of its configuration file, which is in the X resource format. Some resources modify the behavior of **xdm** on all displays, while others modify its behavior on a single display. When actions relate to a specific display, the display name is inserted into the resource name between "DisplayManager" and the final resource name segment. For example, **DisplayManager.expo_0.startup** is the name of the resource that defines the startup shell file on the "expo:0" display. Because the resource manager uses colons to separate the name of the resource from its value and dots to separate resource name parts, **xdm** substitutes underscores for both dots and colons when generating the resource name.

Item	Description
DisplayManager.servers	Specifies either a file name full of server entries, one per line (if the value starts with a slash), or a single server entry. See the section Server Specification for details.
DisplayManager.requestPort	Indicates the UDP port number that the **xdm** command uses to listen for incoming **XDMCP** requests. Unless you need to debug the system, leave this with its default value of 177.
DisplayManager.errorLogFile	Redirects error messages to go to the named file rather than to the console. This file also contains any output directed to standard error by the **Xsetup**, **Xstartup**, **Xsession**, and **Xreset** files, so it will contain descriptions of problems in those scripts as well.

Item	Description
DisplayManager.debugLevel	If the integer value of this resource is greater than 0 (zero), the **xdm** command outputs a large amount of debugging information. It also disables daemon mode, which would discard the information and allow nonroot users to run the **xdm** command that would typically not be useful.
DisplayManager.daemonMode	The **xdm** command attempts to make itself into a daemon process unassociated with any terminal. This is accomplished by forking and leaving the parent process to exit, and then closing file descriptors and releasing the controlling terminal. In some environments this is not desired (in particular, when debugging). Setting this resource to False disables this feature.
DisplayManager.pidFile	The file name specified is created to contain an ASCII representation of the process ID of the main **xdm** process. The **xdm** command also uses file locking on this file to attempt to eliminate multiple daemons running on the same machine, which would have unpredictable results.
DisplayManager.lockPidFile	Controls whether the **xdm** command uses file locking to keep multiple display managers from running simultaneously.
DisplayManager.authDir	Names a directory in which the **xdm** command stores authorization files while initializing the session. The default value is **/usr/lib/X11/xdm**.
DisplayManager.autoRescan	A Boolean value that controls whether the **xdm** command rescans the configuration, servers, access control, and authentication keys files after a session ends and the files have changed. By default the value is True. You can force the **xdm** daemon to reread these files by sending a **SIGHUP** signal to the main process.
DisplayManager.removeDomainname	When computing the display name for **XDMCP** clients, the name resolver typically creates a fully qualified host name for the terminal. As this is sometimes confusing, the **xdm** command removes the domain name portion of the host name if it is the same as the domain name of the local host when this variable is set. The default value is True.
DisplayManager.keyFile	XDM-AUTHENTICATION-1 style **XDMCP** authentication requires that a private key be shared between the **xdm** daemon and the terminal. This resource specifies the file containing those values. Each entry in the file consists of a display name and the shared key. By default, the **xdm** command does not include support for XDM-AUTHENTICATION-1 because it requires the data encryption method (DES), which is not generally distributable because of United States export restrictions.

Item	Description
DisplayManager.accessFile	To prevent unauthorized **XDMCP** service and to allow forwarding of **XDMCP IndirectQuery** requests, this file contains a database of host names that are allowed direct access to this machine or have a list of hosts to which queries should be forwarded. The format of this file is described in the XDMCP Access Control section.
DisplayManager.exportList	A whitespace-separated list of additional environment variables to pass on to the **Xsetup**, **Xstartup**, **Xsession**, and **Xreset** programs.
DisplayManager.randomFile	A file to checksum to generate the seed of authorization keys. This should be a file that changes frequently. The default is **/dev/mem**.
DisplayManager.choiceTimeout	Number of seconds to wait for the display to respond after a user has selected a host from the chooser. If the display sends an `XDMCP IndirectQuery` within this time, the request is forwarded to the chosen host. Otherwise, it is assumed to be from a new session and the chooser is offered again. The default is 15.
DisplayManager.DISPLAY.resources	Specifies the name of the file to be loaded by the **xrdb** command as the resource database onto the root window of screen 0 of the display. The Login widget, **Xsetup**, and **chooser** programs use the resources set in this file. This resource data base is loaded just before the authentication procedure is started, so it can control the appearance of the login window. See the section Authentication Client , which describes the various resources that are appropriate to place in this file. There is no default value for this resource, but **/usr/lib/X11/xdm/Xresources** is the conventional name.
DisplayManager.DISPLAY.chooser	Specifies the program run to offer a host menu for indirect queries redirected to the special host name **CHOOSER**. **/usr/lib/X11/xdm/chooser** is the default. See the sections **XDMCP Access Control** and **Chooser**.
DisplayManager.DISPLAY.xrdb	Specifies the program used to load the resources. By default, the **xdm** command uses **/usr/bin/X11/xrdb**.
DisplayManager.DISPLAY.cpp	Specifies the name of the C preprocessor that is used by the **xrdb** command.
DisplayManager.DISPLAY.setup	Specifies a program that is run (as root) before offering the login window. This resource may be used to change the appearance of the screen around the login window or to put up other windows (for example, you may want to run **xconsole** here). By default, no program is run. The conventional name for a file used here is **Xsetup**. See the section Setup Program.
DisplayManager.DISPLAY.startup	Specifies a program that is run (as root) after the authentication process succeeds. By default, no program is run. The conventional name for a file used here is **Xstartup**. See the section Startup Program.

Item	Description
DisplayManager.DISPLAY.session	Specifies the session to be run (when not running as root). By default, **/usr/bin/X11/xterm** is run. The conventional name is the **Xsession** script. See the section Session Program.
DisplayManager.DISPLAY.reset	Specifies a program that is run (as root) after the session ends. By default, no program is run. The conventional name is the **Xreset** script. See the section Reset Program.
DisplayManager.DISPLAY.openDelay	Controls the behavior of the **xdm** command when attempting to open intransigent servers by specifying the length of the pause (in seconds) between successive attempts.
DisplayManager.DISPLAY.openRepeat	Controls the behavior of the **xdm** command when attempting to open intransigent servers by specifying the number of attempts to make.
DisplayManager.DISPLAY.openTimeout	Controls the behavior of the **xdm** command when attempting to open intransigent servers by specifying the number of seconds to wait while actually attempting the open (that is, the maximum time spent in the **connect(2)** system call).

Item	Description
DisplayManager.DISPLAY.startAttempts	Controls the behavior of the **xdm** command when attempting to open intransigent servers by specifying the number of times that the entire process is completed before giving up on the server. After the number of attempts specified by the Display Manager **openRepeat** resource have been made, or if the number of seconds specified by the Display Manager **openTimeout** resource elapse in any particular attempt, the **xdm** command ends and restarts the server, attempting to connect again. This process is repeated *startAttempts* times, at which point the display is declared inactive and disabled. Although this behavior may seem arbitrary, it has been empirically developed and works well on most systems. The default is a value of **5** for *openDelay*, a value of **5** for *openRepeat*, a value of **30** for *openTimeout*, and a value of **4** for *startAttempts*.
DisplayManager.DISPLAY.pingInterval	To discover when remote displays disappear, the **xdm** command occasionally pings them, using an X connection and **XSync** calls. This resource specifies the time (in minutes) between ping attempts. By default, it is set to 5 minutes. If you frequently use X terminals, which can become isolated from the managing host, you may want to increase this value. **Note:** AIXwindows sessions may continue to exist after the terminal has been accidentally disabled. The **xdm** command does not ping local displays. A workstation session can be ended if the server hangs for NFS service and does not respond to the ping.

Item	Description
DisplayManager.DISPLAY.pingTimeout	To discover when remote displays disappear, the **xdm** command occasionally pings them, using an X connection and **XSync** calls. This resource specifies the maximum amount of time (in minutes) to wait for the terminal to respond to the request. If the terminal does not respond, the session is declared inactive and ended. By default, it is set to 5 minutes. If you frequently use X terminals, which can become isolated from the managing host, you may want to increase this value. **Note:** AIXwindows sessions may continue to exist after the terminal has been accidentally disabled. The **xdm** command does not ping local displays. A workstation session could be ended if the server hangs for NFS service and does not respond to the ping.
DisplayManager.DISPLAY.terminateServer	Specifies whether the X server should be canceled when a session ends (instead of resetting it). This option can be used when the server tends to grow without bound over time, to limit the amount of time the server is run. The default value is False.
DisplayManager.DISPLAY.userPath	The **xdm** command sets the **PATH** environment variable for the session to this value. It should be a list of directories separated by colons; see the **sh** command in *Commands Reference* for a full description. **:/bin:/usr/bin:/usr/bin/X11:/usr/ucb** is a common setting. The default value can be specified at build time in the AIXwindows system configuration file with the **DefaultUserPath** resource.
DisplayManager.DISPLAY.systemPath	The **xdm** command sets the **PATH** environment variable for the startup and reset scripts to the value of this resource. The default for this resource is specified at build time by the **DefaultSystemPath** resource entry in the system configuration file; **/etc:/bin:/usr/bin:/usr/bin/X11:/usr/ucb** is a common choice. Note the absence of . (period) (the current directory) from this entry. This is a good practice to follow for root; it avoids many common "Trojan Horse" system penetration schemes.
DisplayManager.DISPLAY.systemShell	The **xdm** command sets the **SHELL** environment variable for the startup and reset scripts to the value of this resource. It is **/bin/sh** by default.
DisplayManager.DISPLAY.failsafeClient	If the default session fails to run, the **xdm** command returns to this program. This program is run with no arguments, using the same environment variables as the session would have had (see the section Session Program). By default, **/usr/bin/X11/xterm** is used.

Item	Description
DisplayManager.DISPLAY.grabServer **DisplayManager.DISPLAY.grabTimeout**	To improve security, the **xdm** command grabs the server and keyboard while reading the login name and password. The **grabServer** resource specifies if the server should be held for the duration of the name/password reading. When set to False, the server is ungrabbed after the keyboard grab succeeds, otherwise the server is grabbed until just before the session begins. The default value is False. The **grabTimeout** resource specifies the maximum time that the **xdm** command waits for the grab to succeed. The grab may fail if some other client has the server grabbed, or possibly if the network latencies are high. This resource has a default value of 3 seconds; be cautious when raising it, as a user may be confused by a look-alike window on the display. If the grab fails, the **xdm** command becomes inactive and restarts the server (if possible) and the session.
DisplayManager.DISPLAY.authorize **DisplayManager.DISPLAY.authName**	The **authorize** is a Boolean resource that controls whether the **xdm** command generates and uses authorization for the local server connections. If authorization is used, the **xdm** command uses the authorization mechanisms indicated as a whitespace-separated list as the value of the **authName** resource. **XDMCP** connections dynamically specify which authorization mechanisms are supported, so the **authName** resource is ignored in this case. When the **authorize** resource is set for a display and authorization is not available, the user is informed by a different message displayed in the Login widget. By default, the **authorize** resource is True; **authName** is **MIT-MAGIC-COOKIE-1**.
DisplayManager.DISPLAY.authFile	Indicates the file is used to communicate the authorization data from the **xdm** command to the server, using the **-auth** server command-line option. It should be kept in a directory with restricted write permissions as it could easily be removed, disabling the authorization mechanism in the server.
DisplayManager.DISPLAY.authComplain	If set to a value of False, this disables the use of the **unsecureGreeting** in the login window. See the section Authentication Client . The default is a value of True.
DisplayManager.DISPLAY.resetSignal	The number of the signal that the **xdm** command sends to reset the server. See the section Controlling the Server . The default is **1(SIGHUP)**.
DisplayManager.DISPLAY.termSignal	The number of the signal that the **xdm** command sends to end the server. See the section Controlling the Server . The default is **15(SIGTERM)**.
DisplayManager.DISPLAY.resetForAuth	Causes the **xdm** command to send SIGHUP to the server after setting up the authorization file, causing an additional server reset to occur, during which time the new authorization information is read. The default is a value of False, which works for all AIXwindows servers.

Item	Description
DisplayManager.DISPLAY.userAuthDir	When the **xdm** command is unable to write to the usual user authorization file (**$HOME/.Xauthority**), it creates a unique file name in this directory and sets the **XAUTHORITY** environment variable to the name of the created file. It uses **/tmp** by default.

XDMCP Access Control

The database file specified by the **DisplayManager.accessFile** resource provides information that the **xdm** command uses to control access from displays requesting **XDMCP** service. This file contains three types of entries:

- Entries that control the response to **Direct** and **Broadcast** queries.
- Entries that control the response to **Indirect** queries.
- Macro definitions.

Direct query entries contain either a host name or a pattern, which is distinguished from a host name by the inclusion of one or more pattern-matching characters. An * (asterisk) matches any sequence of 0 (zero) or more characters, and a ? (question mark) matches any single character. These are compared against the host name of the display device. If the entry is a host name, all comparisons are done using network addresses, so that any name that converts to the correct network address may be used. For patterns, only actual host names are used in the comparison, so ensure that you do not attempt to match aliases. Preceding either a host name or a pattern with an ! (exclamation point) causes hosts that match that entry to be excluded.

An Indirect entry also contains a host name or pattern, but follows it with a list of host names or macros to which **indirect** queries should be sent.

A macro definition contains a macro name and a list of host names and other macros that the macro expands to. To distinguish macros from host names, macro names start with a % (percent) character. Macros may be nested.

Indirect entries may also specify to have the **xdm** command run the **chooser** command to offer a menu of hosts to which to connect. For more information, see Chooser.

When checking access for a particular display host, each entry is scanned in turn and the first matching entry determines the response. For example, a **Direct** query entry is ignored when scanning for an **Indirect** entry. A **Broadcast** query entry is ignored when scanning for a **Direct** entry.

Blank lines are ignored. The **#** character is treated as a comment delimiter causing the rest of that line to be ignored, and a \ (backslash) at the end of the line causes the new line to be ignored, allowing indirect host lists to span multiple lines.

The following is an example **Xaccess** file:

```
#
# Xaccess - XDMCP access control file
#
```

```
#
# Direct/Broadcast query entries
#
```

```
!xtra.lcs.mit.edu       # disallow direct/broadcast service for xtra
bambi.ogi.edu     # allow access from this particular display
*.lcs.mit.edu     # allow access from any display in LCS
```

```
#
# Indirect query entries
#
```

```
%HOSTS          expo.lcs.mit.edu xenon.lcs.mit.edu \\
               excess.lcs.mit.edu kanga.lcs.mit.edu
```

```
extract.lcs.mit.edu    xenon.lcs.mit.edu    #force extract to contact xenon
!xtra.lcs.mit.edu         dummy     #disallow indirect access
*.lcs.mit.edu     %HOSTS           #all others get to choose
```

Chooser

For X terminals that do not offer a host menu for use with **Broadcast** or **Indirect** queries, the **chooser** program can do this for them. In the **Xaccess** file, specify **CHOOSER** as the first entry in the Indirect host list. The **chooser** program sends a **Query** request to each of the remaining host names in the list and offers a menu of all the hosts that respond.

The list may consist of the word **BROADCAST**, in which case **chooser** sends a **Broadcast** query instead, again offering a menu of all hosts that respond.

The following is an example **Xaccess** file using **chooser**:

```
extract.lcs.mit.edu       CHOOSER   %HOSTS         #offer a menu of these hosts
xtra.lcs.mit.edu          CHOOSER   BROADCAST      #offer a menu of all hosts
```

The program to use for **chooser** is specified by the **DisplayManager.DISPLAY.chooser** resource. Resources for this program can be put into the file named by the **DisplayManager.DISPLAY.resources** resource.

The **chooser** has been implemented using a Motif **SelectionBoxWidget**. Refer to the **XmSelectionBoxWidget Class** documentation for a description of resources and widget or gadget names.

Server Specification

The resource **DisplayManager.servers** gives a server specification or, if the values starts with a / (slash), the name of a file containing server specifications, one per line.

Each specification indicates a display that should constantly be managed and that is not using **XDMCP**. Each consists of at least three parts:

- Display name
- Display class
- Display type
- For local servers, a command line to start the server.

A typical entry for local display number 0 would be:

```
:0 IBM-GT local /usr/bin/X11/X :0
```

The display types are:

Item	Description
local	local display: **\fIxdm\fP** must run the server
foreign	remote display: **\fIxdm\fP** opens an X connection to a running server

The display name must be something that can be passed in the **-display** option to an X program. This string is used to generate the display-specific resource names, so be careful to match the names (for example, use "`:0 local /usr/bin/X11/X :0`" instead of "`` `localhost:0 local /usr/bin/X11/X :0 ``" if your other resources are specified as "`DisplayManager._0.session`"). The display class portion is also used in the display-specific resources as the class of the resource. This is useful if you have a large collection of similar displays (like a corral of X terminals) and would like to set resources for groups of them. When using XDMCP, the display is required to specify the display class, so the manual for your particular X terminal should

document the display class string for your device. If it does not, you can run the **xdm** command in debug mode and look at the resource strings that it generates for that device, which will include the class string.

Setup Program

The **Xsetup** file is run after the server is reset, but before the login window is offered. The file is typically a shell script. It is run as root, so be careful about security. This is the place to change the root background or bring up other windows that should be displayed on the screen along with the Login widget. Because **xdm** grabs the keyboard, other windows will not be able to receive keyboard input. They will be able to interact with the mouse, however; beware of potential security holes here. If **DisplayManager.DISPLAY.grabServer** is set, **Xsetup** will not be able to connect to the display at all. Resources for this program can be put into the file named by **DisplayManager.DISPLAY.resources**.

In addition to any specified by **DisplayManager.exportList**, the following environment variables are passed:

Item	Description
DISPLAY	Specifies the associated display name.
PATH	Specifies the value of **DisplayManager.DISPLAY.systemPath**.
SHELL	Specifies the value of **DisplayManager.DISPLAY.systemShell**.
XAUTHORITY	Specifies that it may be set to an authority file.

Authentication Client

The MIT authentication widget has been replaced by an authentication client composed of standard Motif widgets. The following is a list of the widget names (and their widget class):

```
outframe(xmFrameWidget)
  inframe(xmFrameWidget)
    main(XmFormWidget)
      tframe(xmFrameWidget)
        greeting(xmLabelGadget)
      logoline(xmFormWidget)
        dpyname(xmLabelWidget)
      userline(xmRowColumnWidget)
        userlabel(xmLabelWidget)
        username(xmTextWidget)
        passlabel(xmLabelWidget)
        password(xmTextWidget)
      failsafeline(xmFormWidget)
        failsafe(xmToggleButtonWidget)
      cancelline(xmFormWidget)
        cancel(xmPushButtonWidget)
      message(xmLabelWidget)
```

The authentication client reads a name/password pair from the keyboard. Put resources for this client into the file named by **DisplayManager.DISPLAY.resources**. All of these have reasonable default values, so it is unnecessary to specify any of them. See **/usr/lib/X11/xdm/Xresources** for more information on default values for authentication client resources as well as the appropriate widget class documentation. The following resources are also supported by the authentication client:

Item	Description
Xlogin*foreground	Specifies the color used for the foreground.
Xlogin*background	Specifies the color used for the background.
Xlogin*greeting	Specifies a string that identifies this window. The default is AIXwindows environment.
Xlogin*greetFont	Specifies the font used to display the greeting.
Xlogin*frameColor	Specifies the background color used to display the greeting.
Xlogin*titleFont	Specifies the font used to display the title.
Xlogin*namePrompt	Specifies the string displayed to prompt for a user name. The Xrdb program strips trailing white space from resource values. Add spaces escaped with backslashes at the end of the prompt. The default is "login:".

Item	Description
Xlogin*promptFont	Specifies the font used to display both prompts.
Xlogin*failFont	Specifies the font used for the failsafe button.
Xlogin*cancelFont	Specifies the font used for the cancel button.
Xlogin*messageFontlist	Specifies the font used to display the failure message.
Xlogin*failColor	Specifies the color used to display the failure message.
Xlogin*failTimeout	Specifies the number of seconds that the failure message is displayed. The default is thirty seconds.
Xlogin*sessionArgument	Specifies the argument to be passed to the session program.
Xlogin*XmText.translations	This specifies the translations use for the authentification client. Refer to the X Toolkit documentation for a complete discussion on translations. The default translation table is:

```
Ctrl<Key>b: backward-character()\n\
Ctrl<Key>a: beginning-of-line()\n\
Ctrl<Key>e: end-of-line()\n\
Ctrl<Key>f: forward-character()\n\
Ctrl<Key>d: kill-next-character()\n\
Ctrl<Key>k: kill-to-end-of-line()\n\
Ctrl<Key>u: kill-to-start-of-line()\n
```

You may setup XDM to use the standard XDM translations by replacing the XmText translations as defined in Xresources:

Note: Use <Key>osfHelp instead of <Key>F1 due to the Motif default virtual bindings.)

```
Xlogin*XmText.translations: #override\n\

<Key>osfHelp:    set-session-argument(failsafe) finish-field()\n\
Ctrl<Key>Return: set-session-argument(failsafe) finish-field()\n\
Ctrl<Key>H:      delete-previous-character() \n\
Ctrl<Key>D:      delete-character() \n\
Ctrl<Key>B:      move-backward-character() \n\
Ctrl<Key>F:      move-forward-character() \n\
Ctrl<Key>A:      move-to-beginning() \n\
Ctrl<Key>E:      move-to-end() \n\
Ctrl<Key>K:      erase-to-end-of-line() \n\
Ctrl<Key>U:      erase-line() \n\
Ctrl<Key>X:      erase-line() \n\
<Key>Return:     finish-field() \n
<Key>BackSpace:  delete-previous-character() \n\
<Key>Delete:     delete-previous-character() \n
```

In addition to the typical XmText actions, the following actions are also supported by the client to be compatible with the standard XDM translations:

delete-previous-character
Erases the character before the cursor.

delete-character
Erases the character after the cursor.

move-backward-character
Moves the cursor backward.

move-forward-character
Moves the cursor forward.

move-to-beginning
Moves the cursor to the beginning of the editable text.

move-to-end
Moves the cursor to the end of the editable text.

erase-to-end-of-line
Erases all text after the cursor.

erase-line
Erases the entire text.

finish-field
If the cursor is in the name field, proceeds to the password field; if the cursor is in the password field, checks the current name/password pair. If the name/password pair is valid, **xdm** starts the session. Otherwise the failure message is displayed and the user is prompted again.

insert-char
Inserts the character typed.

set-session-argument
Specifies a single word argument that is passed to the session at startup. See the sections Session Program and Typical Usage.

Startup Program

The **Xstartup** file is typically a shell script. Because it is run as the root user, be careful about security when it runs. It usually contains commands that add entries to **/etc/utmp**, mount users' home directories from file servers, display the message of the day, or cancel the session if logins are not allowed.

In addition to the environment variables specified by **DisplayManager.exportList**, the following variables are passed:

Item	Description
DISPLAY	Specifies the associated display name.
HOME	Specifies the initial working directory of the user.
USER	Specifies the user name.
PATH	Specifies the value of **DisplayManager.DISPLAY.systemPath**.
SHELL	Specifies the value of **DisplayManager.DISPLAY.systemShell**.
XAUTHORITY	May be set to an authority file.

No arguments are passed to the script. The **xdm** command waits until this script exits before starting the user session. If the exit value of this script is nonzero, the **xdm** command discontinues the session and starts another authentication cycle.

Session Program

The **Xsession** program establishes the style of the user's session. It is run with the permissions of the authorized user.

In addition to any specified by **DisplayManager.exportList**, the following environment variables are passed:

Item	Description
DISPLAY	Specifies the associated display name.
HOME	Specifies the initial working directory of the user.
USER	Specifies the user name.
PATH	Specifies the value of **DisplayManager.DISPLAY.userPath**.
SHELL	Specifies the user's default shell (from **getpwnam**).
XAUTHORITY	May be set to a nonstandard authority file.

At most installations, the **Xsession** program should look in the user's home directory (**$HOME**) for a file **.xsession**, which contains the commands that the user would like to use as a session. The **Xsession** program should also implement a system default session if no user-specified session exists. See the section Typical Usage.

An argument may be passed to this program from the authentication widget using the `set-session-argument' action. This can be used to select different styles of session. Usually, this feature is used to allow the user to escape from the ordinary session when it fails. This allows users to repair their own **.xsession** if it fails, without requiring administrative intervention. The section Typical Usage demonstrates this feature.

Reset Program

The **Xreset** script is run after the user session has ended. Run as root, it should contain commands that undo the effects of commands in **Xstartup** by removing entries from **/etc/utmp** or unmounting directories from file servers. The environment variables that are passed to **Xstartup** are also passed to **Xreset**. This program is symmetrical with the **Xstartup** program.

Controlling the Server

The **xdm** command controls local servers using POSIX signals. The **SIGHUP** signal is expected to reset the server, closing all client connections and performing other cleanup duties. The **SIGTERM** signal is expected to cancel the server. If these signals do not perform the expected actions, the resources **DisplayManager.DISPLAY.resetSignal** and **DisplayManager.DISPLAY.termSignal** can specify alternate signals.

To control remote terminals that are not using **XDMCP**, the **xdm** command searches the window hierarchy on the display and uses the protocol request **KillClient** in an attempt to clean up the terminal for the next session. This may not actually cause all of the clients to become inactive, because only those that have created windows will be noticed. **XDMCP** provides a more sure mechanism; when the **xdm** command closes its initial connection, the session is over and the terminal is required to close all other connections.

Controlling XDM

The **xdm** command responds to two signals: **SIGHUP** and **SIGTERM**. When sent a **SIGHUP**, **xdm** rereads the configuration file, the access control file, and the servers file. For the servers file, it notices if entries have been added or removed. If a new entry has been added, the **xdm** command starts a session on the associated display. Entries that have been removed are disabled immediately, meaning that any session in progress is ended without notice and no new session is started.

When sent a **SIGTERM**, the **xdm** command stops all sessions in progress and exits. This can be used when shutting down the system.

The **xdm** command attempts to mark its various subprocesses for use by the **ps** command by editing the command-line argument list in place. Because the **xdm** command cannot allocate additional space for this task, it is useful to start the **xdm** command with a reasonably long command line (using the full path name should be enough). Each process that is servicing a display is marked **-display**.

Other Possibilities

You can use the **xdm** command to run a single session at a time, using the **xinit** command options or other suitable daemons by specifying the server on the command line:

```
xdm -server ":0 local /usr/bin/X11/X :0 -force"
```

It might also run a file server and a collection of X terminals. The configuration for this is identical to the previous sample, except the **Xservers** file would look like the following:

```
        extol:0 VISUAL-19 foreign
        exalt:0 NCD-19 foreign
        explode:0 NCR-TOWERVIEW3000 foreign
```

This directs the **xdm** command to manage sessions on all three of these terminals. See the section Controlling XDM for a description of using signals to enable and disable these terminals.

Note: The **xdm** command does not coexist well with other window systems. To use multiple window systems on the same hardware, use the **xinit** command.

Examples

1. The sample **xstartup** script that follows prevents login while the file **/etc/nologin** exists. As there is no provision for displaying any messages here (there is no core X client that displays files), the setup in this example is not recommended because the login would fail without explanation. Thus, this is not a complete example, but a demonstration of the available functionality.

   ```
   #!/bin/sh
   #
   # Xstartup
   #
   # This program is run as root after the user is verified
   #
   ```

```
if [ \-f /etc/nologin ]; then
        exit 1
fi
exit 0
```

2. This **Xsession** script recognizes the special **failsafe** mode, specified in the translations in the preceding **Xresources** file, to provide an escape from the ordinary session:

```
#!/bin/sh
exec > $HOME/.xsession-errors 2>&1
case $# in
1)
        case $1 in failsafe)
               exec aixterm -geometry 80x24-0-0
               ;;
        esac
esac
startup=$HOME/.xsession
resources=$HOME/.Xresources
if [ -f /usr/bin/X11/startx ]; then
       exec /usr/bin/X11/startx -t -wait
elif [ -f $startup]; then
       exec $startup
else
       if [ -f $resources ]; then
               xrdb -load $resources
       fi
       mwm &
       exec aixterm -geometry 80x24+10+10 -ls
fi
```

3. To have **xdm** come up from system startup, as root type the following:

```
/usr/lib/X11/xdm/xdmconf
```

4. To disable **xdm** on reboot, as root type the following:

```
/usr/lib/X11/xdm/xdmconf -d
```

5. When using **xdm** to manage your display, an authentication procedure ensures that only clients that are allowed can connect to your display. Clients that are built using X11 R4 and X11 R5 libraries understand this protocol. Clients that are built with X11 R3 or earlier libraries do not support this authentication protocol and are not allowed to connect to the Xserver unless **xhost** permission is granted. You can connect local clients by typing the following:

```
xhost =localhost
```

or

```
xhost =machine
```

where *machine* is the hostname of the local client.

Files

Item	Description
/usr/lib/X11/xdm/xdm-config	The default configuration file.
/usr/lib/X11/xdm/Xaccess	The default access file, listing authorized displays.
/usr/lib/X11/xdm/Xservers	The default server file, listing non-XDMCP servers to manage.
$(HOME)/.Xauthority	User authorization file where **xdm** stores keys for clients to read.
/usr/lib/X11/xdm/chooser	The default chooser.
/usr/bin/X11/xrdb	The default resource database loader.

Item	Description
/usr/bin/X11/X	The default server.
/usr/bin/X11/xterm	The default session program and failsafe client.
/usr/lib/X11/xdm/A<host>\-<suffix>	The default place for authorization files.

xfindproxy Command

Purpose

Locates proxy services.

Syntax

xfindproxy **-manager** *ManagerAddress* **-name** *ServiceName* **-server** *ServerAddress* [**-auth**] [**-host** *HostAddress*] [**-options** *Options*]

Description

xfindproxy is a program used to locate available proxy services. It utilizes the Proxy Management Protocol to communicate with a proxy manager. The proxy manager keeps track of all available proxy services, starts new proxies when necessary, and makes sure that proxies are shared whenever possible.

If **xfindproxy** is successful in obtaining a proxy address, it will print it to stdout. The format of the proxy address is specific to the proxy service being used. For example, for a proxy service of LBX, the proxy address would be the X display address of the proxy (e.g, `blah.x.org:63`).

If **xfindproxy** is unsuccessful in obtaining a proxy address, it will print an error to **stderr**.

Flags

Item	Description
-manager	This argument is required, and it specifies the network address of the proxy manager. The format of the address is a standard ICE network id (for example, `tcp/blah.x.org:6500`).
-name	This argument is required, and it specifies the name of the desired proxy service (for example, LBX). The name is case insensitive.
-server	This argument is also required, and it specifies the address of the target server. The format of the address is specific to the proxy service specified with the **-name** argument. For example, for a proxy service of LBX, the address would be an X display address (e.g, `blah.x.org:0`).
-auth	This argument is optional. If specified, **xfindproxy** will read 2 lines from standard input. The first line is an authorization/authentication name. The second line is the authorization/authentication data in hex format (the same format used by xauth). **xfindproxy** will pass this auth data to the proxy, and in most cases, will be used by the proxy to authorize/authenticate itself to the target server.

Item	Description
-host	This argument is optional. If **xfindproxy** starts a new proxy service, it will pass the host specified. The proxy may choose to restrict all connections to this host. In the event that **xfindproxy** locates an already existing proxy, the host will be passed, but the semantics of how the proxy uses this host are undefined.
-options	This argument is optional. If **xfindproxy** starts a new proxy service, it will pass any options specified. The semantics of the options are specific to each proxy server and are not defined here. In the event that **xfindproxy** locates an already existing proxy, the options will be passed, but the semantics of how the proxy uses these options are undefined.

xfs Command

Purpose

Supplies fonts to X Window System display servers.

Syntax

xfs [**-config** *ConfigurationFile*] [**-ls** *ListenSocket*] [**-port** *Number*]

Description

xfs is the AIXwindows font server. It supplies fonts to AIXwindows display servers.

The **xfs** server responds to the following signals:

Item	Description
SIGTERM	Causes the font server to exit cleanly.
SIGUSR1	Causes the server to re-read its configuration file.
SIGUSR2	Causes the server to flush any cached data it may have.
SIGHUP	Causes the server to reset, closing all active connections and re-reading the configuration file.

The server is usually run by a system administrator, and started by way of boot files such as **/etc/rc.tcpip**. Users may also wish to start private font servers for specific sets of fonts.

The configuration language is a list of keyword and value pairs. Each keyword is followed by an = (equal sign) and the desired value.

The following list shows recognized keywords and the types and descriptions of valid values:

Item	Description
#	A comment character when located in the first column.

Item	Description
catalogue (List of string)	Ordered list of font path element names. The current implementation only supports a single catalogue ("all"), containing all of the specified fonts.
alternate-servers (List of string)	List of alternate servers for this font server.
client-limit (Cardinal)	Number of clients that this font server will support before refusing service. This is useful for tuning the load on each individual font server.
clone-self (Boolean)	Whether this font server should attempt to clone itself when it reaches the client-limit.
default-point-size (Cardinal)	The default point size (in decipoints) for fonts that do not specify.
default-resolutions (List of resolutions)	Resolutions the server supports by default. This information may be used as a hint for pre-rendering and substituted for scaled fonts which do not specify a resolution. A resolution is a comma-separated pair of **x** and **y** resolutions in pixels per inch. Multiple resolutions are separated by commas.
error-file (String)	Filename of the error file. All warnings and errors are logged here.
port (Cardinal)	TCP port on which the server will listen for connections. The default is 7100.
use-syslog (Boolean)	Whether the **syslog** function (on supported systems) is to be used for errors.
deferglyphs (String)	Set the mode for delayed fetching and caching of glyphs. Value is `none`, meaning deferred glyphs is disabled. `all`, meaning deferred glyphs is enabled for all fonts, and `16` , meaning deferred glyphs is enabled only for 16-bit fonts.

One of the following forms can be used to name a font server that accepts TCP connections:

```
tcp/hostname:port
tcp/hostname:port/cataloguelist
```

The hostname specifies the name (or decimal numeric address) of the machine on which the font server is running. The port is the decimal TCP port on which the font server is listening for connections. The cataloguelist specifies a list of catalogue names, with '+' as a separator. The following are some examples:

```
tcp/expo.lcs.mit.edu:7100, tcp/18.30.0.212:7101/all
```

One of the following forms can be used to name a font server that accepts DECnet connections:

```
decnet/nodename::font$objname
decnet/nodename::font$objname/cataloguelist
```

The nodename specifies the name (or decimal numeric address) of the machine on which the font server is running. The objname is a normal, case-insensitive DECnet object name. The cataloguelist specifies a list of catalogue names, with '+' as a separator.

Flags

Item	Description
-config *ConfigurationFile*	Specifies the configuration file the font server will use.
-ls *ListenSocket*	Specifies a file descriptor that is already set up to be used as the listen socket. This option is only intended to be used by the font server itself when automatically spawning another copy of itself to handle additional connections.
-port *Number*	Specifies the TCP port number on which the server will listen for connections.

Examples

```
#
# sample font server configuration file
#
```

```
# allow a max of 10 clients to connect to this font server
client-limit = 10
```

```
# when a font server reaches its limit, start up a new one
clone-self = on
```

```
# alternate font servers for clients to use
alternate-servers = hansen:7101,hansen:7102
```

```
# where to look for fonts
# the first is a set of Speedo outlines, the second is a set of
# misc bitmaps and the last is a set of 100dpi bitmaps
#
catalogue = /usr/lib/fonts/type1,
   /usr/lib/X11/ncd/fonts/misc,
   /usr/lib/X11/ncd/fonts/100dpi/
```

```
# in 12 points, decipoints
default-point-size = 120
```

```
# 100 x 100 and 75 x 75
default-resolutions = 100,100,75,75
```

Files

Item	Description
/usr/lib/X11/fs/config	The default configuration file.

xget Command

Purpose

Receives secret mail in a secure communication channel.

Syntax

xget

Description

The **xget** command is used to receive secret mail in a secure communication channel. The messages can be read only by the intended recipient. The **xget** command asks for your password and enables you to read your secret mail.

The **xget** command is used with the **enroll** command and the **xsend** command to send and receive secret mail. The **enroll** command sets up the password used to receive secret mail. The **xsend** command sends mail that can be read only by the intended recipient.

When you issue the **xget** command, you are prompted for your encryption key. Enter the password you previously set up using the **enroll** command.

The prompt for the **xget** command is a ? (question mark). The following subcommands control message disposition:

Item	Description
q (quit)	Writes any mail not yet deleted to the user's mailbox and exits. Pressing End Of File (Ctrl-D) has the same effect.
n (delete) or **d** (delete) or **Enter**	Deletes the current message and displays the next message.
!*Command*	Runs the specified workstation command.
s[*Filename*]	Saves the message in the named *File* parameter instead of in the default mail file, **mbox**.
w[*Filename*]	Saves the message, without its header, in the specified *File* parameter instead of in the default mail file **mbox**.
? (help)	Displays a subcommand summary.

Examples

1. To receive secret mail, enter:

   ```
   xget
   ```

   You are prompted for the password, established with the **enroll** command. After entering your password, the **xget** command prompt (?) and a listing of any secret mail is displayed.
2. To display your secret mail, at the **xget** prompt (?), press the Enter key.

   After the most recent message is displayed, a ? (question mark) indicates the **xget** command is waiting for one of the **xget** subcommands. Enter `help` or a ? (question mark) to list the subcommands available.
3. To save a message or a file to the default mail file, enter:

   ```
   xget
   ```

   Press the Enter key after the ? (question mark) prompt until the desired file is displayed. When the appropriate file is displayed, enter:

   ```
   s
   ```

   In this example, the file is saved in the default mail file, **mbox**.
4. To save a message or a file to a specific file, enter:

   ```
   xget
   ```

Press the Enter key after the ? (question mark) prompt until the desired file is displayed. When the appropriate file is displayed, enter:

```
s mycopy
```

In this example, the file is saved in a file named `mycopy`, instead of the default mail file.

5. To delete a message, enter:

```
xget
```

Press the Enter key after the ? (question mark) prompt until the desired file is displayed. When the appropriate file is displayed, enter:

```
d
```

In this example, the current file is deleted.

Files

Item	Description
/var/spool/secretmail/*User***.key**	Contains the encrypted key for *User*.
/var/spool/secretmail/*User***.[0-9]**	Contains the encrypted mail messages for *User*.
/usr/bin/xget	Contains executable files.

xhost Command

Purpose

Controls who accesses Enhanced X-Windows on the current host machine.

Syntax

xhost [**+** | **-**] [*Name*]

Description

The **xhost** command adds or deletes host names on the list of machines from which the X Server accepts connections.

This command must be run from the machine with the display connection. You can remove a name from the access list by using the **-***Host* parameter. Do not remove the current name from the access list. If you do, log off the system before making any corrections.

Entering the **xhost** command with no variables shows the current host names with access your X Server and a message indicating whether or not access is enabled.

For security, options that affect access control may only be run from the *controlling host*. For workstations, this is the same machine as the server. For X terminals, it is the login host.

To enable a remote name by default, the name can be defined in the **/etc/X***?***.hosts** file, where *?* is the display number to which you enable access.

For example, the display `jeanne:0` can be accessed by systems defined in the **/etc/X0.hosts** file on a system that uses the default host name of `jeanne`. In both the display name and the file name, 0 indicates the display number that the defined remote systems are allowed to access through Enhanced X-Windows.

Flags

Item	Description
+*Name*	Defines the host name (the plus sign is optional) to be added to the X Server access list.
- *Name*	Defines the host name to be removed from the X Server access list. Existing connections are not broken, but new connection attempts will be denied. Note that you can remove the current machine; however, further connections (including attempts to add it back) are not permitted. The only way to allow local connections again is to reset the server (thereby breaking all connections).
+	Specifies that access is unlimited. Access control is turned off.
-	Turns access control on.

The complete *Name* has a the following *family*:*name* syntax:

inet
Internet host

local
Contains only one name, the empty string

Note: The family is case sensitive. The format of the name varies with the family.

xinit Command

Purpose

Initializes the X Window System.

Syntax

xinit [[*Client*] *Options*] [**-** - [*Server*] [*Display*] *Options*]

Description

The **xinit** command starts the AIXwindows server and a first client program on systems that cannot start X directly from **/etc/init** or in environments that use multiple window systems. When this first client exits, the **xinit** command stops the X server and then ends.

If no specific client program is given on the command line, the **xinit** command looks for a file to run to start up client programs. The **xinit** command looks for the **$XINITRC** environment variable. If the file is not there, it then looks for the **$HOME/.xinitrc** file. If it still does not find the file, it follows these steps:

1. The **xinit** command looks next to **/usr/lib/X11/$LANG/xinitrc .**
2. Next, it looks to **/usr/lpp/X11/defaults/$LANG/xinitrc**.
3. And finally, it looks to **/usr/lpp/X11/defaults/xinitrc**.

If no such file exists, **xinit** uses the following as a default:

```
aixterm \-geometry +1+1 \-n login \-display :0
```

If no specific server program is given on the command line, the **xinit** command follows these steps:

1. The **xinit** command looks for a file to run as a shell script to start up the server. The **xinit** command looks for files first in the **$XSERVERRC** environment variable.
2. If the file is not there, it looks for the **$HOME/.xserverrc** file.
3. If it still does not find the **$HOME/.xserverrc** file, it looks next to **/usr/lpp/X11/defaults/xserverrc** file.

4. And finally, if it does not find any of the previous files, the **xinit** command runs the **X** command to start the X server and uses the following as a default:

```
X :0
```

Note that this assumes that there is a program named X in the current search path. However, servers are usually named X*displaytype* where *displaytype* is the type of graphics display which is driven by this server. The site administrator should, therefore, make a link to the appropriate type of server on the machine, or create a shell script that runs the **xinit** command with the appropriate server.

Note: If you attempt to start AIXwindows without an available pointer device, such as a mouse or a tablet, AIXwindows will not open. Some devices can be plugged in but not defined and thus not available to the system, as well as the reverse.

An important point is that programs which are run by **.xinitrc** should be run in the background if they do not exit right away, so that they do not prevent other programs from starting up. However, the last long-lived program started (usually a window manager or terminal emulator) should be left in the foreground so that the script does not exit (which indicates that the user is done and that xinit should exit).

An alternate client and/or server may be specified on the command line. The desired client program and its arguments should be given as the first command line arguments to **xinit**. To specify a particular server command line, add a — (double dash) to the **xinit** command line (after any client and arguments) followed by the desired server command.

Both the client program name and the server program name must begin with a / (slash) or a . (period). Otherwise, they are treated as an arguments to be added to their respective startup lines. This makes it possible to add arguments (for example, foreground and background colors) without having to retype the whole command line.

If a clear server name is not given and the first argument following the — (double dash) is a : (colon) followed by a number, **xinit** uses that number as the display number instead of zero. All remaining arguments are added to the server command line.

The following environment variables are used with the **xinit** command:

Item	Description
DISPLAY	This variable gets set to the name of the display to which clients should connect.
XINITRC	This variable specifies an init file containing shell commands to start up the initial windows. By default, **.xinitrc** in the home directory is used.
Options	List any option you wish that is available to the client you specified.
Client	Specify the client with which you are working. For example, xterm or aixterm. The client you specify must begin with a . (dot) or a / (slash).
Server	Use any valid xserver. The server you specify must begin with a . (dot) or a / (slash).

Examples

1. To start up a server named X and run the user's **xinitrc** program, if it exists, or else start an **aixterm** command enter:

```
xinit
```

2. To start a specific type of server on an alternate display, enter:

```
xinit -- /usr/bin/X11/X qdss:1
```

3. To start up a server named X, and add the given arguments to the default **xinitrc** or **aixterm** command, enter:

```
xinit -geometry =80x65+10+10 -fn 8x13 -j -fg white -bg navy
```

4. To use the command **/Xsun -l -c** to start the server and add the arguments **-e widgets** to the default **xinitrc** or **aixterm** command, enter:

```
xinit -e widgets -- ./Xsun -l -c
```

5. To start a server named X on display 1 with the arguments **-a 2 -t 5**, then start a remote shell on the machine **fasthost** in which it runs the command **cpupig**, telling it to display back on the local workstation, enter:

```
xinit /usr/ucb/rsh fasthost cpupig -display ws:1 -- :1 -a 2 -t 5
```

6. The following sample of the **.xinitrc** script starts a clock, several terminals, and leaves the window manager running as the last application. Assuming that the window manager has been configured properly, the user then chooses the **Exit** menu item to end the AIXwindows session.

```
xrdb -load $HOME/.Xresources
xsetroot -solid gray &
xclock -g 50x50-0+0 -bw 0 &
xload -g 50x50-50+0 -bw 0 &
xterm -g 80x24+0+0 &
xterm -g 80x24+0-0 &
mwm
```

7. Sites that want to create a common startup environment could simply create a default **.xinitrc** script that references a site-wide startup file:

```
#!/bin/sh . /usr/local/lib/site.xinitrc
```

8. Another approach is to write a script that starts the **xinit** command with a specific shell script. Such scripts are usually named `x11`, `xstart`, or `startx` and are a convenient way to provide a simple interface for novice users:

```
#!/bin/sh xinit /usr/local/lib/site.xinitrc -- /usr/bin/X11/X bc
```

Files

Item	Description
.xinitrc	Contains the default client script files.
aixterm	Contains the command the client runs if **.xinitrc** does not exist.
.xserverrc	Contains the default server script.
X	Contains the command the server runs if **.xserverrc** does not exist.

xkbcomp Command

Purpose

Compiles XKB keyboard description.

Syntax

xkbcomp [**-a**] [**-C**] [**-dflts**] [**-I** *Directory*] [**-l**] [**-m** *Name*] [**-merge**] [**-o** *OutputFile*] [**-opt** *Parts*] [**-R** *Directory*] [**-synch**] [**-w** *Level*] [**-xkb**] [**-xkm**] *Source* [*Destination*]

Description

The **xkbcomp** command is a keymap compiler that converts a description of an XKB keymap into one of several output formats. The most common use for **xkbcomp** is to create a compiled keymap file (**.xkm** extension) which can be read directly by XKB-capable X servers or utilities. The keymap compiler can also produce C header files or XKB source files. The C header files produced by **xkbcomp** can be included by X servers or utilities that need a built-in default keymap. The XKB source files produced by **xkbcomp** are fully resolved and can be used to verify that the files which typically make up an XKB keymap are merged correctly or to create a single file which contains a complete description of the keymap.

The *Source* may specify an X display, or an **.xkb** or **.xkm** file; unless explicitly specified, the format of *destination* depends on the format of the source. Compiling a **.xkb** (keymap source) file generates a **.xkm** (compiled keymap file) by default. If the source is a **.xkm** file or an X display, **xkbcomp** generates a keymap source file by default.

If the *Destination* is an X display, the keymap for the display is updated with the compiled keymap.

The name of the *destination* is usually computed from the name of the source, with the extension replaced as appropriate. When compiling a single map from a file which contains several maps, **xkbcomp** constructs the destination file name by appending an appropriate extension to the name of the map to be used.

Flags

Item	Description
-a	Shows all keyboard information, reporting implicit or derived information as a comment. Only affects **.xkb** format output.
-C	Produces a C header file as output (**.h** extension).
-dflts	Computes the defaults for any missing components, such as key names.
-I *Directory*	Specifies the top-level directories to be searched for files included by the keymap description.
-l	List maps that specify the *map* pattern in any files listed on the command line.
-m *Name*	Specifies a map to be compiled from an file with multiple entries.
-merge	Merges the compiled information with the map from the server.
-o *Name*	Specifies a name for the generated output file. The default is the name of the source file with an appropriate extension for the output format.
-opt *Parts*	Specifies a list of optional parts. Compilation errors in any optional parts are not fatal. *Parts* may consist of any combination of the letters **c**, **g**, **k**, **s**, **t** which specify the **c**ompatibility map, **g**eometry, **k**eycodes, **s**ymbols, and **t**ypes, respectively.
-R *Directory*	Specifies the root directory for relative path names.
-synch	Forces synchronization for X requests.
-w *Level*	Controls the reporting of warnings during compilation. A warning level of 0 disables all warnings; a warning level of 10 enables them all.

Item	Description
-xkb	Generates a source description of the keyboard as output (**.xkb** extension).
-xkm	Generates a compiled keymap file as output (**.xkm** extension).

xkbevd Daemon

Purpose

XKB event daemon.

Syntax

xkbevd [**-help**] [**-cfg** *File*] [**-sc** *Command*] [**-sd** *Directory*] [**-display** *Display*] [**-bg**] [**-synch**] [**-v**]

Description

The **xkbevd** event daemon listens for specified XKB events and executes requested commands if they occur. The configuration file consists of a list of event specification/action pairs and/or variable definitions.

An event specification consists of a short XKB event name followed by a string or identifier which serves as a qualifier in parentheses; empty parenthesis indicate no qualification and serve to specify the default command which is applied to events which do not match any of the other specifications. The interpretation of the qualifier depends on the type of the event:

- Bell events match using the name of the bell.
- Message events match on the contents of the message string.
- Slow key events accept any of **press**, **release**, **accept**, or **reject**.

No other events are recognized.

An action consists of an optional keyword followed by an optional string argument. **xkbevd** recognizes the following actions:

- **none**
- **ignore**
- **echo**
- **printEvent**
- **sound**
- **shell**

If the action is not specified, the string is taken as the name of a sound file to be played unless it begins with an exclamation point, in which case it is taken as a shell command.

Variable definitions in the argument string are expanded with fields from the event in question before the argument string is passed to the action processor. The general syntax for a variable is either:

```
$c
```

or

```
$(str)
```

where *c* is a single character and *str* is a string of arbitrary length. All parameters have both single-character and long names. The list of recognized parameters varies from event to event.

The **ignore**, **echo**, **printEvent**, **sound**, and **shell** actions do what you would expect commands named **ignore**, **echo**, **printEvent**, **sound**, and **shell** to do, except that the **sound** command has only been implemented and tested for SGI machines.

The only currently recognized variables are *soundDirectory* and *soundCommand*.

Flags

Item	Description
-bg	Tells **xkbevd** to fork itself and run in the background.
-cfg *File*	Specifies the configuration file to read. If no configuration file is specified, **xkbevd** looks for **~/.xkb/xkbevd.cf** and **$(LIBDIR)/xkb/xkbevd.cf** in that order.
-display *Display*	Specifies the display to use. If not present, **xkbevd** uses **$DISPLAY**.
-help	Prints a usage message.
-sc *Command*	Specifies the command used to play sounds.
-sd *Directory*	Specifies a top-level directory for sound files.
-synch	Forces synchronization of all X requests. Slow.
-v	Prints more information, including debugging messages. Multiple specifications of **-v** causes more output.

xkbprint Command

Purpose

Prints an XKB keyboard description.

Syntax

xkbprint [**-?** | **-help**] [**-color**] [**-dflts**] [**-diffs**] [**-eps**] [**-fit**] [**-full**] [**-grid** *Resolution*] [**-if** *FontName*] [**-label** *Type*] [**-lc** *Locale*] [**-level1**] [**-level2**] [**-lg** *Group*] [**-ll** *Level*] [**-mono**] [**-n** *Number*] [**-nkg** *Number*] [**-npk** *Number*] [**-o** *File*] [**-R** *Directory* [**-pict** *Which*]] *Source* [*OutputFile*]

Description

The **xkbprint** command generates a printable or encapsulated PostScript description of the XKB keyboard description specifed by *Source*. The *Source* can be any compiled keymap, **.xkm** file, that includes a geometry description or an X display specification. If an *OutputFile* is specified, **xkbprint** writes to it. Otherwise, **xkbprint** creates the output file, replacing the extension of the source file with **.ps** or **.eps** depending on the requested format. If the source is a non-local X display, for example `:0`, **xkbprint** appends the appropriate prefix to the display specification, replacing the colon with a **-** (dash). For a local display, **xkprint** uses `server-`n where *n* is the number of the display.

Flags

Item	Description
-? | **-help**	Prints a usage message.

Item	Description
-color	Prints using the colors specified in the geometry file; by default, **xkbprint** prints a black-and-white image of the keyboard.
-dflts	Attempts to compute default names for any missing components, such as keys.
-diffs	Shows symbols only where they are explicitly bound.
-eps	Generates an encapsulated PostScript file.
-fit	Fits the keyboard image on the page, this is the default.
-full	Prints the keyboard at full size.
-grid *Resolution*	Prints a grid with *Resolution***mm** resolution over the keyboard.
-if *FontName*	Specifies an internal PostScript type 1 font to dump to the specified output file or to *fontName***.pfa**, if no output file is specified. No keyboard description is printed if an internal font is dumped.
-label *Type*	Specifies the labels to be printed on keys. Valid types are: • **none** • **name** • **code** • **symbols**
-lc *Locale*	Specifies a locale in which KeySyms should be resolved.
-level1	Generates a level 1 PostScript.
-level2	Generates a level 2 PostScript.
-lg *Group*	Prints symbols in keyboard groups starting from *Group*.
-ll *Level*	Prints symbols starting from shift level *Level*.
-mono	Generates a black-and-white image of keyboard, this is the default.
-n *Number*	Prints *Number* of copies.
-nkg *Number*	Prints the symbols in *Number* keyboard groups.
-npk *Number*	Specifies the *Number* of keyboard images to print on each page. For EPS files, this specifies the total number of keyboard images to print.
-o *File*	Writes the output to *File*.
-R *Directory*	Use *Directory* as the root directory; all path names are interpreted relative to *Directory*.

Item	Description
-pict *Which*	Controls the use of pictographs instead of keysym names where available. Valid values for *Which* are: • **all** • **none** • **common** (default).
-synch	Forces synchronization for X requests.
-w *Level*	Sets warning level. • 0 for no warning • 10 for all warnings

xlock Command

Purpose

Locks the local X display until a password is entered.

Syntax

xlock [**-batchcount** *Number*] [**-bg** *Color*] [**-delay** *Users*] [**-display** *Display*] [**-fg** *Color*] [**-font** *FontName*] [**-info** *TextString*] [**-invalid** *TextString*] [**-mode** *ModeName*] [**+mono** | **-mono**] [**-username** *TextString*] [**-nice** *Level*] [**+nolock** | **-nolock**] [**-password** *TextString*] [**+remote** | **-remote**] [**+allowaccess** | **-allowaccess**] [**+allowroot** | **-allowroot**] [**+echokeys** | **-echokeys**] [**+enablesaver** | **-enablesaver**] [**-help**] [**-saturation** *Value*] [**-timeout** *Seconds*] [**+usefirst** | **-usefirst**] [**+v** | **-v**] [**-validate** *TextString*]

Description

The **xlock** command locks the X server until the user enters a password at the keyboard. While the **xlock** command is running, all new server connections are refused. The screen saver is disabled, the mouse cursor is turned off, the screen is blanked, and a changing pattern is displayed. If a key or a mouse button is pressed, a prompt asks for the password of the user who started the **xlock** command**.**

If the correct password is typed, the screen is unlocked and the X server is restored. When typing the password, Ctrl-U and Ctrl-H are active as kill and erase, respectively. To return to the locked screen, click in the small icon version of the changing pattern.

To function properly, **xlock** needs to run with root permission since the operating system restricts access to the password and access control files. To give **xlock** root permission, perform the following steps:

1. Log in as root.
2. Go to the directory that contains the **xlock** program file.
3. Run these two commands:
   a. **chown root xlock**
   b. **chmod u+s xlock**

Authentication

The **xlock** command is a Pluggable Authentication Module (PAM) enabled X server command that locks the X server until the user enters a password. It supports both local UNIX authentication and PAM authentication for unlocking the X server.

You can set the system-wide configuration to use PAM for authentication by providing root user access and by modifying the value of the *auth_type* attribute to *PAM_AUTH* in the **usw** stanza of the `/etc/security/login.cfg` file.

The authentication mechanisms that are used when PAM is enabled are dependent on the configuration of the login service in the `/etc/pam.conf` file. The **xlock** command requires the `/etc/pam.conf` file entry for the **auth**, **account**, **password**, and **session** module types. The following configuration is recommended for the `/etc/pam.conf` file entry in the **xlock** command:

```
xlock               auth                required          pam_aix
xlock               account             required          pam_aix
xlock               password            required          pam_aix
xlock               session             required          pam_aix
```

Flags

Item	Description
-batchcount *Number*	Sets the number of things to do per batch. *Number* refers to different things depending on the mode: **qix** Refers to the number of lines rendered in the same color. **hop** Refers to the number of pixels rendered in the same color. **image** Refers to the number of sunlogos on screen at once. **swarm** Refers to the number of bees **life and blank** Does not apply.
-bg *Color*	Sets the color of the background on the password screen.
-delay *Number*	Sets the speed at which a mode operates to the number of microseconds to delay between batches of **hopalong** pixels, **qix** lines, **life** generations, **image** bits, and **swarm** motions. In the **blank** mode, it is important to set this to a small number because the keyboard and mouse are only checked after each delay. A delay of zero would needlessly consume the processing unit while checking for mouse and keyboard input in a tight loop since the **blank** mode has no work to do.
-display *Display*	Sets the X11 display to lock. The **xlock** command locks all available screens on the server and restricts you to locking only a local server, such as **unix:0**, **localhost:0**, or **:0** (unless you set the **-remote** flag).
-fg *Color*	Sets the color of the text on the password screen.
-font *FontName*	Sets the font to be used on the prompt screen.
-help	Prints a brief description of available options.
-info *TextString*	Defines an informational message. The default is `Enter password to unlock; select icon to lock`.
-invalid *TextString*	Specifies an password message. The default is `Invalid login`.

Item	Description
-mode *ModeName*	Specifies one the following six display modes: **blank** Displays a black screen. **hop** Displays the real plane fractals from the September, 1986 issue of *Scientific American*. **image** Displays several randomly appearing sun logos. **life** Displays Conway's game of life. **qix** Displays spinning lines. **swarm** Displays a swarm of bees following a wasp.
-nice *NiceLevel*	Sets system nicelevel of the **xlock** process.
-password *TextString*	Specifies the password prompt string. The default is `Password:`.
-saturation *Value*	Sets saturation of the color ramp. A value of 0 (zero) is grayscale and a value of 1 is very rich color. A value of 0.4 is a medium pastel.
-timeout *Seconds*	Sets the number of seconds before the password screen times out.
-username *TextString*	Specifies the message shown in front of the user name. The default is `Name:`.
-validate *TextString*	Specifies the message that is shown while validating the password. The default is `Validating login...`.
-/+allowaccess	Allows the disabling of the access control list, but still causes the local server to prompt for a password. If **xlock** is killed using the **-KILL** command, the access control list is not lost. This flag is also needed when running the **xlock** command remotely since access to the control list is restricted.
-/+allowroot	Allows the root password to unlock the server as well as the user who started the **xlock** command.
-/+echokeys	Causes the **xlock** command to echo to screen a '?' (question mark) character for each key typed into the password prompt. The default is no echo.
+/-enablesaver	Enables the default screensaver. It is possible to set delay parameters long enough to cause phosphor burn on some displays. This flag can be used as an added precaution.
+/-mono	Causes the **xlock** command to display monochrome (black and white) pixels rather than the default colored ones on color displays.
+/-nolock	Causes the **xlock** command to only draw the patterns and not to lock the display. A keypress or a mouse click terminates the screen saver.
+/-remote	Allows remote locking of X11 servers. This flag should be used with care. It is intended mainly to lock X11 terminals that cannot run the **xlock** command locally. If you lock a workstation other than your own, that person will need your password to unlock it.The **-remote** option does not disable your ability to toggle to another shell.

Item	Description
+/-usefirst	Allows using the keystroke which obtained the password screen as the first input character in the password. The default ignores the first keystroke.
+/-v	Minus prefix enables the verbose mode to tell which options the **xlock** command is going to use. The plus prefix is the default.

xlsfonts Command

Purpose

Displays the font list for X-Windows.

Syntax

xlsfonts [**-display** *Host*:*Display*] [**-l** [**l** [**l**]]] [**-m**] [**-C**] [**-1**] [**-w** *Width*] [**-n** *Columns*] [**-u**] [**-o**] [**-fn** *Pattern*]

Description

The **xlsfonts** command lists the fonts that match a specified *Pattern* parameter. Use the wildcard character "*" (asterisk) to match any sequence of characters (including none), and the "?" (question mark) to match any single character. If no pattern is given, "*" is assumed.

> **Note:** The "*" and "?" characters must be placed within quotation marks to prevent them from being expanded by the shell.

You can use flags to specify servers, number and width of columns to print, size of font listings, whether the output should be sorted, and whether to use **OpenFont** instead of **ListFonts**.

Flags

> **Note:** Using the **-l** (lowercase L) flag of the **xlsfonts** command can tie up your server for a long time. This is typical of single-threaded non-preemptable servers, and not a program error.

Item	Description
-1	Indicates that listings should use a single column. This flag is the same as the **-n** *1* flag.
-C	Indicates that listings should use multiple columns. This flag is the same as the **-n** *0* flag.
-display *Host*:*Display*	Identifies the X Server to contact by specifying the host name and display number.
-fn *Pattern*	Specifies the fontname *Pattern* that **xlsfonts** will list.
-l [**l** [**l**]]	(lowercase L) Indicates that medium, long, and very long listings, respectively, should be generated for each font.
-m	Indicates that long listings should also print the minimum and maximum bounds of each font.
-n *Columns*	Specifies the number of columns to use to display the output. By default, the **xlsfonts** command tries to fit as many columns of font names into the number of characters specified by the **-w** *Width* flag.

Item	Description
-o	Instructs the **xlsfonts** command to perform **OpenFont** (and **QueryFont**, if appropriate) instead of **ListFonts**. The **-o** flag is useful if the **ListFonts** or **ListFontsWithInfo** fails to list a known font, as is the case with some scaled font systems.
-u	Indicates that the output should remain unsorted.
-w *Width*	Specifies the width in characters that should be used to determine how many columns to print. The default is 79.

Environment Variable

Item	Description
DISPLAY	Gets the default host and display to use.

Examples

1. To specify a medium-sized list of each font, use a lowercase L and enter:

   ```
   xlsfonts -l
   ```

2. To specify a three-column list of each font, enter:

   ```
   xlsfonts -n 3
   ```

3. To display all fonts with the string iso8859 within their names, enter:

   ```
   xlsfonts -ll "*"iso8859"*"
   ```

4. To list all fonts with rom1 plus one following character in their names, enter:

   ```
   xlsfonts rom1"?"
   ```

   This obtains a listing similar to:

   ```
   rom10  rom11  rom14  rom16  rom17
   ```

xmbind Command

Purpose

Configures virtual key bindings.

Syntax

xmbind [**-display** *Host:Display:ScreenID*] [*FileName*]

Description

The **xmbind** command is an X Windows System client that configures the virtual key bindings for AIXwindows applications. This action is performed by the **mwm** command at its startup, so the **xmbind** client is only needed when **mwm** is not in use or when you want to change bindings without restarting **mwm**. If a file is specified, its contents are used as the virtual key bindings. If a file is not specified, the **.motifbind** file in the user's home directory is used. If this file is not found, the **xmbind** command loads the default virtual key bindings.

Flags

Item	Description
-display *Host*:*Display*:*ScreenID*	Specifies the display to use. The **-display** option has the following parameters: ***Host*** Specifies the host name of a valid system on the network. Depending on the situation, this could be the host name of the user or the host name of a remote system. ***Display*** Specifies the number (usually 0) of the display on the system on which the output is to be displayed. ***ScreenID*** Specifies the number of the screen where the output is to be displayed. This number is 0 for single-screen systems.

Parameters

Item	Description
FileName	Specifies the file containing bindings for virtual mouse and key events.

Exit Status

This command returns the following exit values:

Item	Description
0	Indicates successful completion.
>0	Indicates an error occurred.

xmkmf Command

Purpose

Creates a **Makefile** from an **Imakefile**.

Syntax

xmkmf [**-a**] [*TopDir* [*CurDir*]]

Description

The **xmkmf** command creates a **Makefile** from an **Imakefile** shipped with third-party software. When invoked with no arguments or variables in a directory containing an **Imakefile** file, the **imake** command runs with arguments appropriate for your system (configured into **xmkmf** when X was built) and generates a **Makefile**.

Flag

Item	Description
-a	First builds the **Makefile** in the current directory, then automatically executes **make Makefiles**, **make includes**, and **make depend**. This is how to configure software that is outside of the MIT X build tree.

Variables

Specify *TopDir* and *CurDir* if you are working inside the MIT X build tree (highly unlikely unless you are an X developer).

Item	Description
TopDir	Specify as the relative path name from the current directory to the top of the build tree.
CurDir	Specify as a relative path name from the top of the build tree to the current directory. The *CurDir* variable is required if the current directory has subdirectories; otherwise, the **Makefile** will not be able to build the subdirectories. If a *TopDir* variable is given in its place, **xmkmf** assumes nothing is installed on your system and searches for files in the build tree instead of using the installed versions.

xmwlm Command

Purpose

Provides recording of system performance or WLM metrics.

Syntax

`xmwlm` [`-d` *recording_dir*] [`-n` *recording_name*] [`-t` *trace_level*] [`-L`]

Description

The `xmwlm` agent provides recording capability for a limited set of local system performance metrics. These include common CPU, memory, network, disk, and partition metrics typically displayed by the `topas` command. Daily recordings are stored in the `/etc/perf/daily` directory. The `topasout` command is used to output these recordings in raw ASCII or spreadsheet format. The `xmwlm` agent can also be used to provide recording data from Workload Management (WLM). This is the default format used when `xmwlm` is run without any flags. Daily recordings are stored in the `/etc/perf/wlm` directory. The `wlmmon` command can be used to process WLM-related recordings. The `xmwlm` agent can be started from the command line, from a user script, or can be placed near the end of the `/etc/inittab` file. All recordings cover 24-hour periods and are only retained for seven days by default. You can configure the **retain** value in the **/usr/lpp/perfagent/daily.cf** file to change the default recording duration.

Flags

Item	Description
`-d` *recording_dir*	Specifies the output directory for the recording files. The default location is `/etc/perf/wlm` when `xmwlm` is run without any flags and `/etc/perf/daily` when `xmwlm` is run with the `-L` flag.
`-L`	Specifies the collection of topas-like metrics. The metric set is not user configurable.

Item	Description
`-n` *recording_name*	Specifies the name for the recording file. By default, `xmwlm` creates recording files named in an **xmwlm.***YYMMDD* format. For example, if **-n myrecording** is specified, the recording files will be named **myrecording.***YYMMDD*.
`-t` *trace_level*	Specifies a trace level. `xmwlm` prints various information to a log file in the appropriate `/etc/perf` subdirectory. The trace level can be set from 1 to 9. More trace data is generated at higher trace levels. This trace data is useful to determine `xmwlm` recording status and for debugging purposes. The log file name is `xmwlm.log1` or `xmwlm.log2`. `xmwlm` cycles between these two files after a file reaches the maximum allowable size.

Session Recovery by xmwlm

If the `xmwlm` agent is terminated and restarted, `xmwlm` examines the recording files in the appropriate `/etc/perf` subdirectory or in the directory specified by the `-d` flag. If a recording file exists with the current date, `xmwlm` appends data to this file and continues to write to the recording file. Otherwise, a new recording file is created.

Location

/usr/bin/xmwlm

Files

Item	Description
`/usr/bin/xmwlm`	Contains the `xmwlm` agent. The agent is part of the `perfagent.tools` fileset.

xmodem Command

Purpose

Transfers files with the **xmodem** protocol, detecting data transmission errors during asynchronous transmission.

Syntax

xmodem { **-s** | **-r** } *FileName*

Description

The **xmodem** shell command is used with the Asynchronous Terminal Emulation (ATE) program to transfer a file, designated by the *FileName* parameter, using the **xmodem** protocol.

The **xmodem** protocol is an 8-bit transfer protocol to detect data transmission errors and retransmit the data. The workstation sending data waits until the remote system sends a signal indicating it is ready to receive data.

After the receiving system get data, it returns an acknowledgment to the sending system. In the ATE program the receiving system times out if data is not received within 90 seconds after the file transfer is initiated.

Sending and receiving with the **xmodem** command are complementary operations. One system must be set to send while the other is set to receive. Use the **xmodem** command on the remote system in combination with the **send** subcommand or the **receive** subcommand from the ATE Connected Main Menu on the local system.

To interrupt an **xmodem** file transfer, press the Ctrl-X key sequence.

Note:

1. The DOS operating system terminates each line in an ASCII file with a newline character and a carriage return (Ctrl-M) character. UNIX terminates each line in an ASCII file only with a newline character. The carriage return characters are preserved when a DOS file is transferred to AIX. The `vi` text editor can be used to remove spurious Ctrl-M characters using the subcommand

   ```
   :%s/<Ctrl-V><Ctrl-M>//
   ```

   where <Ctrl-V> and <Ctrl-M> each represent a single control character that is typed. However, since Ctrl-V is the default ATE MAINMENU_KEY, the ATE defaults must be altered in order to issue the `vi` subcommand while logged in via ATE.
2. The `xmodem` file transfer process adds Ctrl-Z characters to the last packet transferred to make the packet 128 bytes long. Most files transferred will, therefore, have Ctrl-Z characters appended to the end. The DOS operating system terminates an ASCII file with a Ctrl-Z character. Every file transferred from DOS to AIX will, therefore, end with at least one Ctrl-Z character. These extra Ctrl-Z characters can be removed with the `vi` text editor.

Flags

Item	Description
-r	Receives data from the local workstation.
-s	Sends data to the local workstation.

Examples

Sending a File with the xmodem Protocol

To send the file `myfile` with the **xmodem** protocol, use the **ate** command and the **connect** or **directory** subcommand to establish a connection to the remote system.

1. After logging in to the remote system and before pressing the MAINMENU_KEY (usually the Ctrl-Vkey sequence) to return to ATE on the local system, enter:

   ```
   xmodem -r myfile
   ```

   at the shell command line. The **xmodem** protocol starts receive mode on the remote system.
2. Press the MAINMENU_KEY to return to ATE on the local system.

   The ATE Connected Main Menu displays.
3. Enter the **send** subcommand at the prompt on the ATE Connected Main Menu:

   ```
   s myfile
   ```

   The **send** subcommand instructs the local system to send `myfile` to the remote system. After transferring the file, the ATE Connected Main Menu displays.

Receiving a File with the xmodem Protocol

Receive the file `infile` from a remote system using **xmodem** protocol with the **ate** command and the **connect** or **directory** subcommand establishing a connection to the remote system.

1. After logging in to the remote system and before pressing the MAINMENU_KEY (usually the Ctrl-V key sequence) to return to ATE on the local system, enter:

   ```
   xmodem -s infile
   ```

   at the shell command line. The **xmodem** protocol starts, in send mode, on the remote system.
2. Press the MAINMENU_KEY to return to ATE on the local system.

   The ATE Connected Main Menu displays.
3. Enter the **receive** subcommand at the prompt on the ATE Connected Main Menu:

   ```
   r infile
   ```

   The **receive** subcommand instructs the local system to receive `infile` from the remote system. After transferring the file, the ATE Connected Main Menu displays.

File

Item	Description
ate.def	Contains ATE default values.

xmodmap Command

Purpose

Modifies keymaps in the X Server.

Syntax

xmodmap [**-display** *Display*] [**-e** *Expression*] [**-grammar** | **-help**] [**-n**] [**-pk**] [**-pke**] [**-pm**] [**-pp**] [**-quiet** | **-verbose**] [*FileName*]

Description

The **xmodmap** command edits and displays the keyboard modifier map and keymap table that client applications use to convert event keycodes into key symbols. It is usually run from the session startup script to configure the keyboard according to the personal tastes of the user.

Every time a keycode expression is evaluated, the server generates a **MappingNotify** event on every client. All of the changes should be batched together and done at one time. Clients that receive keyboard input and ignore **MappingNotify** events will not notice any changes made to keyboard mappings.

The *FileName* parameter specifies a file containing the **xmodmap** command expressions to be run. This file is usually kept in the home directory of the user with a name like **.xmodmaprc**. If no file is specified, input is taken from **stdin**.

The **xmodmap** command program reads a list of expressions and parses them all before attempting to run any of them. This makes it possible to refer to key symbols that are being naturally redefined without having to worry as much about name conflicts.

Item	Description
add	The key symbol names are evaluated as the line is read. This permits you to remove keys from a modifier without worrying about whether they were reassigned.

Item	Description
add *ModifierName* **=** *KeySymbolName...*	Adds the given key symbols to the indicated modifier map. The key symbol names are evaluated after all input expressions are read to make it easy to write expressions to swap keys.
clear *ModifierName*	Removes all entries in the modifier map for the given modifier, where the valid names are **Shift**, **Lock**, **Control**, **Mod1**, **Mod2**, **Mod3**, **Mod4**, and **Mod5** (case does not matter in modifier names, although it does matter for all other names). For example, **clear Lock** removes all keys bound to the **shift lock** modifier.
keycode *Number* **=** *KeySymbolName...*	Assigns the list of key symbols to the indicated keycode (which can be specified in decimal, hex, or octal and be determined by running the **xev** program in the **/usr/lpp/X11/Xamples/demos** directory). Usually only one key symbol is assigned to a given code.
keysym *KeySymbolName* **=** *KeySymbolName...*	The *KeySymbolName* on the left hand side is translated into matching keycodes used to perform the corresponding set of **keycode** expressions. The list of keysym names can be found in the keysym database **/usr/lib/X11/XKeysymDB** or the header file **X11/keysymdef.h** (without the *XK_* prefix). Note that if the same keysym is bound to multiple keys, the expression is run for each matching keycode.
pointer = default	Sets the pointer map back to its default settings (such as, button 1 generates a code of 1, button 2 generates a 2, and so forth).
pointer = *Button1 Button2 Button3...*	Sets the pointer map to contain the indicated button codes. The list always starts with the first physical button.
remove *ModifierName* **=** *KeySymbolName...*	Removes all keys containing the given keysyms from the indicated modifier map. Unlike **add**, the keysym names are evaluated as the line is read in. This allows for the removal of keys from a modifier without having to worry about whether or not they have been reassigned.

Lines that begin with an ! (exclamation point) are taken as comments.

If you want to change the binding of a modifier key, you must also remove it from the appropriate modifier map.

Flags

Item	Description
-display *Display*	Specifies the host and display to use.
-e *Expression*	Specifies an expression to be run. You can specify any number of expressions from the command line.
-grammar	Prints a help message describing the expression grammar used in files and with the **-e** *Expressions* flag prints to standard error.
-help	Prints a brief description of the command line arguments to standard error. This is done whenever an unhandled argument is given to the **xmodmap** command.

Item	Description
-n	Indicates that the **xmodmap** command should not change the mappings, but should display what it would do when given this flag.
-pk	Indicates that the current keymap table should print on the standard output.
-pke	Indicates that the current keymap table should be printed on the standard output in the form of expressions that can be fed back to **xmodmap**. This flag is specific to X11R5.
-pm	Indicates that the current modifier map should print on the standard output.
-pp	Indicates that the current pointer map should print on the standard output.
-quiet	Turns off the verbose logging. This is the default.
-verbose	Indicates that the **xmodmap** command should print logging information as it parses its input.

Examples

1. The following command reverses the button codes that get generated so that the primary button is pressed using the index finger of the left hand on a 3 button pointer:

```
xmodmap -e "pointer = 1 2 3 4 5"
```

2. The following command attaches meta to the multi-language key (sometimes labeled Compose Character). It also takes advantage of the fact that applications that need a Meta key simply need to get the keycode and do not require the key symbol to be in the first column of the keymap table. This means that applications that are looking for a Multi_key (including the default modifier map) will not notice any change.

```
keysym Multi_key = Multi_key Meta_L
```

3. To automatically generate less than and greater than characters when the comma and period keys are shifted, reset the bindings for the comma and period with the following scripts:

```
!
! make shift-, be < and shift-. be >
!
keysym comma = comma less
keysym period = period greater
```

4. To swap the location of the Control and Shift Lock keys, use the following script:

```
!
! Swap Caps_Lock and Control_L
!
remove Lock = Caps_Lock
remove Control = Control_L
keysym Control_L = Caps_Lock
keysym Caps_Lock = Control_L
add Lock = Caps_Lock
add Control = Control_L
```

xmpeek Command

Purpose

The **xmpeek** command allows you to query any host about the status of its **xmtopas** daemon.

Syntax

xmpeek [**-a** | **-l**] [*hostname*]

Description

The **xmpeek** command is used to list down the data consumers that currently have instruments (stat sets) defined with the **xmtopas** daemon, and list down all known data consumers by the **xmtopas** daemon . The **xmpeek** command is also used to print down all the available SPMI statistics for any given host.

Flags

Item	Description
-a	If this flag is specified, one line is listed for each data consumer known by the daemon. If this flag is not used, only data consumers that currently have instruments (stat sets) defined with the daemon are listed. This flag is optional.
-l	Lists all the available statistics of the remote host. This flag is optional.
hostname	If the hostname is specified, the daemon on the named host is asked. If no host name is specified, the daemon on the local host is asked.

Examples

The following is an example of the output from the **xmpeek** program:

```
Statistics for xmtopas daemon on *** birte ***
Instruments currently defined:    1
Instruments currently active:     1
Remote monitors currently known:  2
--Instruments--- Values  Packets
                                 Internet Protocol
Defined Active  Active  Sent        Address      Port Hostname
------- ------- ------- ------- ---------------- ---- ----------
1       1       16      3,344   129.49.115.208   3885 xtra
```

Output from **xmpeek** can take two forms.

The first form is a line that informs you that the **xmtopas** daemon is not feeding any data-consumer programs. This form is used if no statsets are defined with the daemon and no command flags are supplied.

The second form includes at least as much as is shown in the preceding example, except that the single detail line for the data consumer on host **xtra** is shown only if either the **-a** flag is used or if the data consumer has at least one instrument (statset) defined with the daemon. Note that **xmpeek** itself appears as a data consumer because it uses the Remote Statistics Interface (RSI) API to contact the daemon. Therefore, the output always shows at least one known monitor.

In the fixed output, first the name of the host where the daemon is running is shown. Then follows three lines giving the totals for current status of the daemon. In the above example, you can see that only one instrument is defined and that it's active. You can also see that two data consumers are known by the daemon, but that only one of them has an instrument defined with the daemon in **birte**. Obviously, this output was produced without the **-a** flag.

An example of more activity is shown in the following sample output from **xmpeek**. The output is produced with the command:

```
xmpeek -a birte
```

Notice that some detail lines show zero instruments defined. Such lines indicate that an **are_you_there** message was received from the data consumer but that no states were ever defined or that any previously defined states were erased.

```
Statistics for smeared daemon on *** birte ***
   Instruments currently defined:    16
   Instruments currently active:     14
   Remote monitors currently known:   6
--Instruments--- Values Packets   Internet Protocol
 Defined Active  Active  Sent         Address      Port  Hostname
   8       8       35    10,232    129.49.115.203  4184   birte
   6       4       28     8,322    129.49.246.14   3211   umbra
   0       0        0         0    129.49.115.208  3861   xtra
   1       1       16     3,332    129.49.246.14   3219   umbra
   0       0        0         0    129.49.115.203  4209   birte
   1       1       16       422    129.49.115.208  3874   xtra
------- ------- ------- ----------
  16      14       95    22,308
```

Notice that the same host name may appear more than once. This is because every running copy of **xmperf** and every other active data-consumer program is counted and treated as a separate data consumer, each identified by the port number used for UDP packets as shown in the **xmpeek** output.

The second detail shows that one particular monitor on host **umbra** has six instruments defined but only four active. This would happen if a remote **xmperf** console has been opened but is now closed. When you close an **xmperf** console, it stays in the Monitor menu of the **xmperf** main window and the definition of the instruments of that console remains in the tables of the data-supplier daemon but the instruments are not active.

xmscheck Command

Purpose

The **xmscheck** command is available to pre-parse a recording configuration file and to determine how the running **xmtopas** daemon is configured for recording.

Syntax

xmscheck [*file_name*]

Description

When the **xmtopas** command is started with the command line argument **-v**, its recording configuration file parser writes the result of the parsing to the log file. The output includes a copy of all lines in the recording configuration file, any error messages, and a map of the time scale with indication of when recording starts and stops.

Although this is useful to document what is read from the recording configuration file, it is not a useful tool for debugging of a new or modified recording configuration file. Therefore, the program **xmscheck** command is available to preparse a recording configuration file before you move it to the **/etc/perf** directory, where the **xmtopas** command looks for the recording configuration file.

When **xmscheck** command is started without any command line argument, it parses the file **/etc/perf/xmservd.cf**. This way, you can determine how the running daemon is configured for recording. If a file name is specified on the command line, that file is parsed.

Output from the **xmscheck** command goes to stdout. The parsing is done by the exact same module that does the parsing in the **xmtopas** command. That module is linked in as part of both programs. The parsing checks that all statistics specified are valid and prints the time scale for starting and stopping recording in the form of a “time table.”

In the time table, each minute has a numeric code. The meaning of codes is as follows:

Item	Description
0	Recording is inactive. Neither a start nor a stop request was given for the minute.
1	Recording is active. Neither a start nor a stop request was given for the minute.
2	Recording is inactive. A stop request was given for the minute.
3	Recording is active. A start request was given for the minute.

Examples

The following example shows how the **xmscheck** command formats the time table. The table only displays part of Tuesday.

```
Day 2, Hour 00: 000000000000000000000000000000000000000000000000000000000000000000000000
Day 2, Hour 01: 000000000000000000000000000000000000000000000000000000000000000000000000
Day 2, Hour 02: 000000000000000000000000000000000000000000000000000000000000000000000000
Day 2, Hour 03: 000000000000000000000000000000000000000000000000000000000000000000000000
Day 2, Hour 04: 000000000000000000000000000000000000000000000000000000000000000000000000
Day 2, Hour 05: 000000000000000000000000000000000000000000000000000000000000000000000000
Day 2, Hour 06: 000000000000000000000000000000000000000000000000000000000000000000000000
Day 2, Hour 07: 000000000000000000000000000000000000000000000000000000000000000000000000
Day 2, Hour 08: 000000000000000000000000000000311111111111111111111111111111111111111111
Day 2, Hour 09: 111111111111111111111111111111111111111111111111111111111111111111111111
Day 2, Hour 10: 111111111111111111111111111111111111111111111111111111111111111111111111
Day 2, Hour 11: 111111111111111111111111111111111111111111111111111111111111111111111111
Day 2, Hour 12: 200000000000000000000000000000000000000000000000000000000000000000000000
Day 2, Hour 13: 311111111111111111111111111111111111111111111111111111111111111111111111
Day 2, Hour 14: 111111111111111111111111111111111111111111111111111111111111111111111111
Day 2, Hour 15: 111111111111111111111111111111111111111111111111111111111111111111111111
Day 2, Hour 16: 111111111111111111111111111111111111111111111111111111111111111111111111
Day 2, Hour 17: 200000000000000000000000000000000000000000000000000000000000000000000000
Day 2, Hour 18: 000000000000000000000000000000000000000000000000000000000000000000000000
Day 2, Hour 19: 000000000000000000000000000000000000000000000000000000000000000000000000
Day 2, Hour 20: 000000000000000000000000000000000000000000000000000000000000000000000000
Day 2, Hour 21: 000000000000000000000000000000000000000000000000000000000000000000000000
Day 2, Hour 22: 000000000000000000000000000000000000000000000000000000000000000000000000
Day 2, Hour 23: 000000000000000000000000000000000000000000000000000000000000000000000000
```

xmtopas Command

Purpose

The **xmtopas** daemon acts as a recording facility and is controlled through the xmtopas.cf configuration file. This daemon simultaneously provides near real-time network-based data monitoring and local recording on a given node.

Syntax

xmtopas [**-v**] [**-b** *UDP_buffer_size*] [**-i** *min_remote_interval*] [**-l** *remove_consumer_timeout*] [**-m** *supplier_timeout*] [**-p** *trace_level*] [**-s** *max_logfile_size*] [**-t** *keep_alive_limit*] [**-x** *xmtopas_execution_priority*]

Description

The **xmtopas** command is always started from **inetd** daemon. Therefore, command line options must be specified on the line defining the **xmtopas** command to the **inetd** daemon in the /etc/inetd.conf file.

Note:

1. If you modify the **xmtopas** entry in the **inetd.conf** file on the Virtual IO Server (VIOS), the new entry is added after migration. You must replace the **xmtopas** entry to a default type, such as **xmtopas -p3**, before migration.

2. For any dynamic configuration changes to the logical partition, the **xmtopas** must be restarted to reflect the changes.

Flags

Item	Description
-v	Causes parsing information for the **xmtopas** recording configuration file to be written to the **xmtopas** log file.
-b *UDP_buffer_size*	Defines the size of the buffer used by the daemon to send and receive UDP packets. The buffer size must be specified in bytes and can be from 4,096 to 16,384 bytes. The buffer size determines the maximum number of data values that can be sent in one **data_feed** packet. The default buffer size is 4096 bytes, which allows for up to 124 data values in one packet.
-i *min_remote_interval*	Defines the minimum interval in milliseconds with which data feeds can be sent. Default is 500 milliseconds. A value between 100 and 5,000 milliseconds can be specified. Any value specified is rounded to a multiple of 100 milliseconds. Whichever minimum remote interval is specified causes all requests for data feeds to be rounded to a multiple of this value.
-l *remove_consumer_timeout*	Sets the **time_to_live** after feeding of statistics data has ceased as described in section Life and Death of **xmtopas**. Must be followed by a number of minutes. A value of 0 (zero) minutes causes the daemon to stay alive forever. The default **time_to_live** is 15 minutes.
-m *supplier_timeout*	When a dynamic data-supplier is active, this value sets the number of seconds of inactivity from the DDS before the SPMI assumes the DDS is dead. When the timeout value is exceeded, the **SiShGoAway** flag is set in the shared memory area and the SPMI disconnects from the area. If this flag is not given, the timeout period is set to 90 seconds. The size of the timeout period is kept in the SPMI common shared memory area. The value stored is the maximum value requested by any data consumer program, including the **xmtopas** command.
-p *trace_level*	Sets the trace level, which determines the types of events written to the **/var/perf/xmtopas.log1** log file or the **/var/perf/xmtopas.log2** log file. This flag must be followed by a digit from 0 to 9, with 9 being the most detailed trace level. Default trace level is 0 (zero), which disables tracing and logging of events but logs error messages.
-s *max_logfile_size*	Specifies the approximate maximum size of the log files. At least every **time_to_live** minutes, it is checked if the currently active log file is bigger than the *max_logfile_size* value. If so, the current log file is closed and logging continues to the alternate log file, which is first reset to zero length. The two log files are **/var/perf/xmtopas.log1** and **/var/perf/xmtopas.log2**. Default maximum file size is 100,000 bytes. You cannot make the *max_logfile_size* value smaller than 5,000 or larger than 10,000,000 bytes.

Item	Description
-t *keep_alive_limit*	Sets the *keep_alive_limit* value must be followed by a number of seconds from 60 to 900 (1 to 15 minutes). The default is 300 seconds (5 minutes).
-x *xmtopas_execution_priority*	Sets the execution priority of the **xmtopas** command. Use this option if the default execution priority of the **xmtopas** command is unsuitable in your environment. Generally, the daemon should be given as high execution priority as possible (a smaller number gives a higher execution priority). On systems other than IBM RS/6000 systems, the *-x* flag is used to set the nice priority of the **xmtopas** command. The nice priority is a value from -20 to 19. The default is -20.

Files used by the xmtopas command

You can specify the following entries in the **xmtopas.res** file:

```
docec:<arguments>
docluster:cluster=<cluster configuration file>
```

Example

```
docec: availmem=5 unavailmem=2
docluster: cluster=/etc/perf/xmtopasagg.cf
```

The following new fields are added to the **docec** entry in the **xmtopas.res** file to get the Hardware Management Console (HMC) details:

```
managedsys=[Managed system name under which this partition is configured]
hmc=[HMC name under which this partition is configured]
```

If the HMC platform cannot be configured for automatic queries, the global data fields that are not available to the local partition can be set by using the following options:

```
availmem = [Total amount of memory allocated to all partitions, in GB]
unavailmem = [Total amount of memory unallocated from the HMC, in GB]
availprocessor = [Total number of physical processors allocated for all partitions]
unavailprocessor = [Total number of physical processors unallocated from the HMC]
poolsize = [Defined Pool Size required if HMC Processor Utilization Authority restricts access]
partitions = [Number of partitions defined on the HMC]
reconfig = [Number of seconds between checking for HMC configuration changes.
Allowed values are 30, 60, 90, 120, 180, 240, 300 seconds.
The default is 60 seconds.]
```

Example

```
docec: hmc=hmcmac.in.ibm.com managedsys=cec1
```

xntpd Daemon

Purpose

Starts the Network Time Protocol (NTP) daemon.

Syntax

xntpd [**-a**] [**-b**] [**-d**] [**-D** *Level*] [**-m**] [-x][**-c** *ConfigFile*] [**-e** *AuthenticationDelay*] [**-f** *DriftFile*] [**-k** *KeyFile*] [**-l** *LogFile*] [**- o** *TraceFile*] [**-p** *pidFile*] [**-r** *BroadcastDelay*] [**-s** *StatsDirectory*] [**-t** *TrustedKey*] [**-v** *SystemVariable*] [**-V** *SystemVariable*]

Description

The **xntpd** daemon sets and maintains a UNIX system time-of-day in compliance with Internet standard time servers. The **xntpd** daemon is a complete implementation of the Network Time Protocol (NTP) version 3 standard, as defined by RFC 1305, and also retains compatibility with version 1 and 2 servers as defined by RFC 1059 and RFC 1119, respectively. The **xntpd** daemon does all computations in fixed point arithmetic and does not require floating point code.

The **xntpd** daemon reads from a configuration file (**/etc/ntp.conf** is the default) at startup time. You can override the configuration file name from the command line. You can also specify a working, although limited, configuration entirely on the command line, eliminating the need for a configuration file. Use this method when configuring the **xntpd** daemon as a broadcast or multicast client, that determines all peers by listening to broadcasts at runtime. You can display the **xntpd** daemon internal variables with the **ntpq** command (Network Time Protocol (NTP) query program). You can alter configuration options with the **xntpdc** command.

The **xntpd** daemon operates in several modes, including symmetric active/passive, client/server and broadcast/multicast. A broadcast/multicast client can automatically discover remote servers, compute one-way delay correction factors and configure itself automatically. This mode makes it possible to deploy a group of workstations without specifying a configuration file or configuration details specific to its environment.

> **Note:** When operating in a client mode, the **xntpd** daemon will exit with an error if no configured servers are within 1000 seconds of local system time. Use the **date** or **ntpdate** command to set the time of a bad skewed system before starting **xntpd**.

Flags

Item	Description
-a	Runs in authenticate mode
-b	Listens for broadcast NTP and synchronizes to them if available.
-c *ConfigFile*	Specifies the name of an alternate configuration file.
-d	Specifies debugging mode. This flag may occur multiple times (maximum of 10), with each occurrence indicating greater detail of display.
-D *Level*	Specifies debugging level directly (value from 1 to 10).
-e *AuthenticationDelay*	Specifies the time, in seconds, it takes to compute the NTP encryption field on this computer.
-f *DriftFile*	Specifies the location of the drift file.
-k *KeyFile*	Specifies the location of the file which contains the NTP authentication keys.
-l *LogFile*	(lowercase L) Specifies the use of a log file instead of logging to syslog.
-m	Listens for multicast messages and synchronizes to them if available. Assumes multicast address 224.0.1.1.
-o *TraceFile*	Specifies trace file name (default is **stderr**).

Item	Description
-p *pidFile*	Specifies the name of the file to record the daemon's process id. There is no default.
-r *BroadcastDelay*	Specifies the default delay (in seconds) if the calibration procedure fails. Normally, the **xntpd** daemon automatically compensates for the network delay between the broadcast/multicast server and the client.
-s *StatsDirectory*	Specifies the directory to use for creating statistics files.
-t *TrustedKey*	Adds the specified key number to the trusted key list.
-v *SystemVariable*	Adds the specified system variable
-V *SystemVariable*	Adds the specified system variable listed by default.
-x	Makes small time adjustments. (SLEWING)

Reference Clock Support

For the purposes of configuration, the **xntpd** daemon treats reference clocks in a manner analogous to normal NTP peers as much as possible. It refers to reference clocks by address, same as a normal peer is, though it uses an invalid IP address to distinguish them from normal peers. AIX supports one type of reference clock, based on the system clock (type 1).

Reference clock addresses are of the form 127.127.*Type*.*Unit* where *Type* is an integer denoting the clock type and *Unit* indicates the type-specific unit number. You configure reference clocks by using a server statement in the configuration file where the *HostAddress* is the clock address. The key, version and ttl options are not used for reference clock support.

Reference clock support provides the **fudge** command, which configures reference clocks in special ways. This command has the following format:

fudge 127.127.*Type***.***Unit* [**time1** *Seconds*] [**time2** *Seconds*] [**stratum** *Integer*] [**refid** *Integer*] [**flag1 0** | **1**] [**flag2 0** | **1**] [**flag3 0** | **1**] [**flag4 0** | **1**]

The **time1** and **time2** options are in fixed point seconds and used in some clock drivers as calibration constants.

The **stratum** option is a number in the range zero to 15 and used to assign a nonstandard operating stratum to the clock. Since the **xntpd** daemon adds one to the stratum of each peer, a primary server ordinarily displays stratum one. In order to provide engineered backups, use the **stratum** option to specify the reference clock stratum as greater than zero. Except where noted, this option applies to all clock drivers.

The **refid** option is an ASCII string in the range one to four characters and used to assign a nonstandard reference identifier to the clock.

The binary flags: **flag1**, **flag2**, **flag3** and **flag4** are for customizing the clock driver. The interpretation of these values, and whether they are used at all, is a function of the needs of the particular clock driver.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Auditing Events: N/A

Examples

1. To start the **xntpd** daemon, enter:

```
startsrc -s xntpd
```

2. To stop the **xntpd** daemon, enter:

```
stopsrc -s xntpd
```

3. To use the authentication key file `/etc/ntp.new.keys` when running the **xntpd** daemon, enter:

```
/usr/sbin/xntpd -k /etc/ntp.new.keys
```

Files

Item	Description
/usr/sbin/xntpd	Contains the **xntpd** daemon.
/etc/ntp.conf	Contains the default configuration file.
/etc/ntp.drift	Contains the default drift file.
/etc/ntp.keys	Contains the default key file.

xntpdc Command

Purpose

Starts the query/control program for the Network Time Protocol daemon, **xntpd**.

Syntax

xntpdc [**-i**] [**-l**] [**-n**] [**-p**] [**-s**] [**-c** *SubCommand*] [*Host ...*]

Description

The **xntpdc** command queries the **xntpd** daemon about its current state and requests changes to that state. It runs either in interactive mode or by using command-line arguments. The **xntpdc** command interface displays extensive state and statistics information. Nearly all the configuration options that can be specified at start-up using the **xntpd** daemon's configuration file, can also be specified at run-time using the **xntpdc** command.

If you enter the **xntpdc** command with one or more request flags, the NTP servers running on each of the hosts specified (or defaults to local host) receive each request. If you do not enter any request flags, the **xntpdc** command tries to read commands from standard input and run them on the NTP server running on the first host specified or on the local host by default. It prompts for subcommands if standard input is the terminal.

The **xntpdc** command uses NTP mode 7 packets to communicate with the NTP server and can query any compatible server on the network that permits it.

The **xntpdc** command makes no attempt to retransmit requests, and will time-out requests if the remote host does not respond within a suitable time.

Specifying a flag other than **-i** or **-n** sends the queries to the specified hosts immediately. Otherwise, the **xntpdc** command attempts to read interactive format commands from standard input.

Flags

Item	Description
-c *SubCommand*	Specifies an interactive format command. This flag adds *SubCommand* to the list of commands to run on the specified hosts. You can enter multiple **-c** flags.
-i	Specifies interactive mode. Standard output displays prompt and standard input reads commands.
-l	(lowercase L) Displays a list of the peers known to the servers. This is the same as the **listpeers** subcommand.
-n	Displays all host addresses in dotted decimal format (0.0.0.0) rather than the canonical host names.
-p	Displays a list of the peers known to the server and a summary of their state. This is the same as the **peers** subcommand.
-s	Displays a list of the peers known to the server and a summary of their state but in a format different from the **-p** flag. This is the same as the **dmpeers** subcommand.

Parameters

Item	Description
Host ...	Specifies the hosts.

xntpdc Internal Subcommands

You can run a number of interactive format subcommands entirely within the **xntpdc** command that do not send NTP mode 7 requests to a server. The following subcommands can only be used while running the **xntpdc** query program.

Interactive Format Subcommands

Interactive format subcommands consist of a keyword followed by zero to four arguments. You only need to type enough characters of the full keyword to uniquely identify the subcommand. The output of a subcommand goes to standard output, but you can redirect the output of individual subcommands to a file by appending a greater-than sign (**>**), followed by a file name, to the command line.

Item	Description
? [*SubCommand*]	Displays command usage information. When used without *SubCommand*, displays a list of all the **xntpdc** command keywords. When used with *SubCommand*, displays function and usage information about the command.
help [*SubCommand*]	Same as the **?** [*Subcommand*] subcommand.
delay *Milliseconds*	Specifies the time interval to add to timestamps included in requests that require authentication. This subcommand enables unreliable server reconfiguration over long delay network paths or between machines whose clocks are unsynchronized. If you enter this subcommand without an argument, it prints the current setting for this subcommand.
host *HostName*	Specifies the host to send queries to. *HostName* may be either a host name or a numeric address. If you enter this subcommand without an argument, it prints the current setting for this subcommand.

Item	Description
hostnames yes | **no**	Specifies whether to display the host name (**yes**) or the numeric address (**no**). The default is **yes** unless the **-n** flag is used. If you enter this subcommand without an argument, it prints the current setting for this subcommand.
keyid *Number*	Specifies the server key number to use to authenticate configuration requests. If you enter this subcommand without an argument, it prints the current setting for this subcommand.
passwd	Prompts you to type in the NTP server authentication password to use to authenticate configuration requests.
quit	Exits the **xntpdc** query program.
timeout *Milliseconds*	Specifies the time-out period for responses to server queries. The default is 8000 milliseconds. If you enter this subcommand without an argument, it prints the current setting for this subcommand.

Query Subcommands

The **xntpdc** query subcommands result in sending NTP mode 7 packets containing requests to the server. These subcommands are read-only (they do not modify the server configuration state).

Item	Description
clkbug *ClockPeerAddress* [*Addr2*] [*Addr3*] [*Addr4*]	Displays debugging information for a reference clock driver. Some clock drivers provide this information that is mostly undecodable without a copy of the driver source in hand.
clockbug *ClockPeerAddress* [*Addr2*] [*Addr3*] [*Addr4*]	Displays information concerning a peer clock. The values obtained provide information on the setting of fudge factors and other clock performance information.
dmpeers	Displays a list of peers for which the server is maintaining state, along with a summary of that state. Identical to the output of the **peers** subcommand except for the character in the leftmost column. Characters only are displayed beside peers that were included in the final stage of the clock selection algorithm. The possible character in the leftmost column are: **.** Indicates that this peer was cast off in the falseticker detection. **+** Indicates that the peer made it through. ***** Denotes the peer the server is currently synchronizing with.
iostats	Displays statistics counters maintained in the input-output module.

Item	Description
kerninfo	Displays kernel phase-lock loop operating parameters. This information is available only if the kernel of the hosts being generated has been specially modified for a precision timekeeping function.
listpeers	Displays a brief list of the peers for which the server is maintaining state. These include all configured peer associations as well as those peers whose stratum is such that the server considers them to be possible future synchronization candidates.
loopinfo [**oneline** | **multiline**]	Displays the values of selected loop filter variables. The loop filter is the part of NTP that adjusts the local system clock. The `offset` is the last offset given to the loop filter by the packet processing code. The `frequency` is the frequency error of the local clock in parts-per-million (ppm). The `poll adjust` controls the stiffness (resistance to change) of the phase-lock loop and the speed at which it can adapt to oscillator drift. The `watchdog timer` is the number of elapsed seconds since the last sample offset given to the loop filter. The **oneline** and **multiline** options specify the format to display this information. The **multiline** option is the default.
memstats	Displays statistics counters related to memory allocation code.
monlist	Displays traffic counts collected and maintained by the monitor facility.

Item	Description
peers	Displays a list of peers for which the server is maintaining state, along with a summary of that state. Summary information includes: • address of the remote peer, • reference ID (0.0.0.0 for an unknown reference ID), • the stratum of the remote peer (a stratum of 16 indicates the remote peer is unsynchronized), • the polling interval (seconds), • the reachability register (octal), and • the current estimated delay, offset and dispersion of the peer (seconds). The character in the left margin indicates the mode this peer entry is in: **+** symmetric active. **-** symmetric passive. **=** remote server polled in client mode. **^** server is broadcasting to this address. **~** remote peer is sending broadcasts. ***** marks the peer the server is currently synchronizing to. The contents of the host field may be a host name, an IP address, a reference clock implementation name with its parameter or REFCLK (*ImplementationNumber*, *Parameter*). Only IP addresses display when using **hostnames no**.
pstats *PeerAddress* [*Addr2*] [*Addr3*] [*Addr4*]	Displays per-peer statistic counters associated with the specified peers.
reslist	Displays the server's restriction list which may help to understand how the restrictions are applied.

Item	Description
sysinfo	Displays a variety of system state variables related to the local server. All except the last four lines are described in the NTP Version 3 specification, RFC 1305. The system flags show various system flags, some of which can be set and cleared by the **enable** and **disable** configuration statements. The `stability` is the residual frequency error remaining after applying the system frequency correction. You use it for maintenance and debugging. In most architectures, this value will initially decrease from as high as 500 ppm to a nominal value in the range .01 to 0.1 ppm. If it remains high for some time after starting the daemon, something may be wrong with the local clock, or the value of the kernel variable *Tick* may be incorrect. The `broadcastdelay` shows the default broadcast delay, as set by the **broadcastdelay** configuration statement, while the `authdelay` shows the default authentication delay, as set by the **authdelay** configuration statement.
sysstats	Displays statistics counters maintained in the protocol module.
timerstats	Displays statistics counters maintained in the timer/event queue support code.

Runtime Configuration Requests Subcommands

The server authenticates all requests that cause state changes in the server by using a configured NTP key. The server can also disable this facility by not configuring a key. You must make the key number and the corresponding key known to the **xtnpdc** command. You can do this by using the **keyid** and **passwd** subcommands, which prompts at the terminal for a password to use as the encryption key. The **xtnpdc** command will also prompt you automatically for both the key number and password the first time you give a subcommand that would result in an authenticated request to the server. Authentication not only verifies that the requester has permission to make such changes, but also protects against transmission errors.

Authenticated requests always include a timestamp in the packet data, as does the computation of the authentication code. The server compares this timestamp to the time at which it receives the packet.

The server rejects the request if they differ by more than 10 seconds. This makes simple replay attacks on the server, by someone able to overhear traffic on your LAN, much more difficult. It also makes it more difficult to request configuration changes to your server from topologically remote hosts. While the reconfiguration facility works well with a server on the local host, and may work adequately between time-synchronized hosts on the same LAN, it works very poorly for more distant hosts. So, if you choose reasonable passwords, take care in the distribution and protection of keys and apply appropriate source address restrictions, the run-time reconfiguration facility should provide an adequate level of security.

The following subcommands all make authenticated requests.

Item	Description
addpeer *PeerAddress* [*Keyid*] [*Version*] [**prefer**]	Adds a configured peer association operating in symmetric active mode at the specified address. You may delete an existing association with the same peer or simply convert an existing association to conform to the new configuration when using this subcommand. If the *Keyid* is a nonzero integer, all outgoing packets to the remote server will have an authentication field attached encrypted with this key. To specify no authentication , enter *Keyid* as 0 or leave blank. The values for *Version* can be 1, 2 or 3, with 3 as the default. The **prefer** option indicates a preferred peer used primarily for clock synchronization if possible. The preferred peer also determines the validity of the PPS signal. If the preferred peer is suitable for synchronization, so is the PPS signal.
addserver *PeerAddress* [*Keyid*] [*Version*] [**prefer**]	Same as the **addpeer** subcommand, except that the operating mode is client.
addtrap *Address* [*Port*] [*Interface*]	Sets a trap for asynchronous messages at the specified address and port number for sending messages with the specified local interface address. If you do not specify the port number, the value defaults to 18447. If you do not specify the interface address, the value defaults to the source address of the local interface.
authinfo	Displays information concerning the authentication module, including known keys and counts of encryptions and decryptions performed.
broadcast *PeerAddress* [*Keyid*] [*Version*]	Same as the **addpeer** subcommand, except that the operating mode is broadcast. The *PeerAddress* can be the broadcast address of the local network or a multicast group address assigned to NTP (224.0.1.1).
clrtrap *Address* [*Port*] [*Interface*]	Clears a trap for asynchronous messages at the specified address and port number for sending messages with the specified local interface address. If you do not specify the port number, the value defaults to 18447. If you do not specify the interface address, the value defaults to the source address of the local interface.
delrestrict *Address Mask* [**ntpport**]	Deletes the matching entry from the restrict list.
disable *Option* ...	Disables various server options. Does not affect options not mentioned. The **enable** subcommand describes the options.

Item	Description
enable *Option* ...	Enables various server options. Does not affect options not mentioned. You can specify one or more of the following values for *Option*: **auth** Causes the server to synchronize with unconfigured peers only if the peer has been correctly authenticated using a trusted key and key identifier. The default for this option is disable (off). **bclient** Causes the server to listen for a message from a broadcast or multicast server, following which an association is automatically instantiated for that server. The default for this argument is disable (off). **monitor** Enables the monitoring facility, with default enable (on). **pll** Enables the server to adjust its local clock, with default enable (on). If not set, the local clock free-runs at its intrinsic time and frequency offset. This option is useful when the local clock is controlled by some other device or protocol and NTP is used only to provide synchronization to other clients. **stats** Enables statistics facility filegen, with default enable (on).
fudge *PeerAddress* [*Time1*] [*Time2*] [*Stratum*] [*Refid*]	Provides a way to set certain data for a reference clock. *Time1* and *Time2* are in fixed point seconds and used in some clock drivers as calibration constants. *Stratum* is a number in the range zero to 15 and used to assign a nonstandard operating stratum to the clock. *Refid* is an ASCII string in the range one to four characters and used to assign a nonstandard reference identifier to the clock.
monitor yes | **no**	Enables or disables the monitoring facility. A **monitor no** subcommand followed by a **monitor yes** subcommand is a good way of resetting the packet counts.
readkeys	Purges the current set of authentication keys and obtains a new set by rereading the keys file specified in the **xntpd** configuration file. This allows you to change encryption keys without restarting the server.
reset *Module*	Clears the statistics counters in various modules of the server. You can specify one or more of the following values for *Module*: **io**, **sys**, **mem**, **timer**, **auth**, **allpeers**.

Item	Description
restrict *Address Mask Option ...*	Adds the values of *Option* to an existing restrict list entry, or adds a new entry to the list with the specified *Option*. The **mask** option defaults to 255.255.255.255, meaning that *Address* is treated as the address of an individual host. You can specify one or more of the following values for *Option*: **ignore** Ignore all packets from hosts that match this entry. Does not respond to queries nor time server polls. **limited** Specifies that these hosts are subject to client limitation from the same net. Net in this context refers to the IP notion of net (class A, class B, class C, etc.). Only accepts the first **client_limit** hosts that have shown up at the server and that have been active during the last **client_limit_period** seconds. Rejects requests from other clients from the same net. Only takes into account time request packets. Private, control, and broadcast packets are not subject to client limitation and therefore do not contribute to client count. The monitoring capability of the **xntpd** daemon keeps a history of clients. When you use this option, monitoring remains active. The default value for **client_limit** is 3. The default value for **client_limit_period** is 3600 seconds. **lowpriotrap** Declare traps set by matching hosts to low priority status. The server can maintain a limited number of traps (the current limit is 3), assigned on a first come, first served basis, and denies service to later trap requestors. This parameter modifies the assignment algorithm by allowing later requests for normal priority traps to override low priority traps. **nomodify** Ignore all NTP mode 6 and 7 packets that attempt to modify the state of the server (run time reconfiguration). Permits queries that return information. **nopeer** Provide stateless time service to polling hosts, but not to allocate peer memory resources to these hosts. **noquery** SIgnore all NTP mode 6 and 7 packets (information queries and configuration requests) from the source. Does not affect time service. **noserve** Ignore NTP packets whose mode is not 6 or 7. This denies time service, but permits queries. **notrap** Decline to provide mode 6 control message trap service to matching hosts. The trap service is a subsystem of the mode 6 control message protocol intended for use by remote event logging programs. **notrust** STreat these hosts normally in other respects, but never use them as synchronization sources. **ntpport** Match the restriction entry only if the source port in the packet is the standard NTP UDP port (123).
setprecision *Precision*	Sets the precision that the server advertises. *Precision* should be a negative integer in the range -4 through -20.
traps	Displays the traps set in the server.

Item	Description
trustkey *Keyid* ...	Adds one or more keys to the trusted key list. When you enable authentication, authenticates peers with trusted time using a trusted key.
unconfig *PeerAddress* [*Addr2*] [*Addr3*] [*Addr4*]	Removes the configured bit from the specified peers. In many cases deletes the peer association. When appropriate, however, the association may persist in an unconfigured mode if the remote peer is willing to continue on in this fashion.
unrestrict *Address Mask Option* ...	Removes the specified options from the restrict list entry indicated by *Address* and *Mask*. The **restrict** subcommand describes the values for *Option*.
untrustkey *Keyid* ...	Removes one or more keys from the trusted key list.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Access Control: You must be part of the system group to run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Auditing Events: N/A

Displays per-peer statistic counters associated with the specified peers.

Examples

1. To start the query/control program for the Network Time Protocol daemon, enter:

```
xntpdc
```

2. To display the statistic counters of the peer at address 127.127.1.0 on host 9.3.149.107, enter:

```
xntpdc -c "pstats 127.127.1.0" 9.3.149.107
```

Output similar to the following is displayed:

```
remote host:   LOCAL(0)
local interface:   127.0.0.1
time last received:   49s
time until next send:   15s
reachability change:   818s
packets sent:   13
packets received:   13
bad authentication:   0
bogus origin:   0
duplicate:   0
bad dispersion:   4
bad reference time:   0
candidate order:   1
```

Files

Item	Description
/usr/sbin/xntpdc	Contains the **xntpdc** command.

xpr Command

Purpose

Formats a window dump file for output to a printer.

Syntax

xpr [**-append** *FileName* [**-noff**] | **-output** *FileName*] [**-landscape** | **-portrait**] [**-compact**] [**-cutoff** *Level*] [**-density** *Dpi*] [**-gray** { **2** | **3** | **4** }] [**-header** *String*] [**-height** *Inches*] [**-left** *Inches*] [**-noposition**] [**-plane** *PlaneNumber*] [**-psfig**] [**-report**] [**-rv**] [**-scale** *Scale*] [**-split** *Number*] [**-top** *Inches*] [**-trailer** *String*] [**-width** *Inches*] [**-device** *Device*] [*ImageFile*]

Description

The **xpr** command uses a window dump file produced by the **xwd** utility as input and formats the dump file for output on all printers supported by the hardware. If you do not specify a file argument, the **xpr** command uses standard input. By default, the **xpr** command prints the largest possible representation of the window on the output page.

The **xpr** command options allow you to add headers and trailers, specify margins, adjust the scale and orientation, and append multiple window dumps to a single output file. Output is to standard output unless the **-output** flag is specified.

Flags

Item	Description
-append *FileName*	Specifies a file name previously produced by the **xpr** command to which the window is to append. (This flag is not supported on PostScript printers.)
-compact	Uses simple run-length encoding for compact representation of windows with many white pixels. This flag compresses white space but not black space, so it is not useful for reverse-video windows. (This flag supports PostScript, LIPS II+, and LIPSIII output only.)
-cutoff *Level*	Changes the intensity level where colors are mapped to black or white for monochrome output on a LaserJet printer. The *Level* variable is expressed as a percentage of full brightness. Fractions are acceptable.

Item	Description
-device *Device*	Specifies the device on which the file prints. The **xpr** command supports the following printers: **3812 or pp** IBM PP3812 **4207** Proprinter **5201** IBM Quietwriter 1 model 2 **5202** IBM Quietwriter 2 **jprinter** IBM Japanese Printer (Japanese data stream) **ljet** HP LaserJet and IBM Laser Printer **ps** PostScript printers (this is the default) **lips2** Canon LaserShot LIPS II+ mode **lips3** Canon LaserShot LIPS III mode
-density *Dpi*	Indicates the dots-per-inch (dpi) density that the HP printer uses. 300 dpi is the default. Allowable densities are 300, 150, 100, and 75 dpi.
-gray *Number*	Specifies gray-scale conversion to a color image, rather than mapping to a black-and-white image. The *Number* variable must be one of the following: **2** 2 x 2 conversion **3** 3 x 3 conversion **4** 4 x 4 conversion This conversion doubles, triples, or quadruples, respectively, the effective width and height of the image. **Note:** This option is valid only for PostScript printers.
-header *String*	Specifies a header string to print above the window.
-height *Inches*	Specifies the maximum height of the page.
ImageFile	Contains the captured bitmap of the image. If you do not specify the *ImageFile* parameter, the **xpr** command uses standard input.
-landscape	Forces the window to print in landscape mode. (The display is laid out with the windows being wider than they are high.) By default, a window prints so that its longest side follows the long side of the paper.
-left *Inches*	Specifies the left margin in inches. Fractions are acceptable. By default, this flag prints the window on the center of the page.
-noff	When specified in conjunction with the **-append** flag, the window is displayed on the same page as the previous window. (This flag is not supported on PostScript printers.)

Item	Description
-noposition	Causes the header, trailer, and image positioning command generation to be bypassed for the LaserJet printer.
-output *FileName*	Specifies an output file name. If you do not specify this option, the **xpr** command uses standard output.
-plane*PlaneNumber*	Specifies which bit plane to use in an image. The default uses the entire image and maps values into black and white based on color intensities. This option is not supported for the LaserJet printer.
-portrait	Forces the window to print in portrait mode. (The display is laid out with the windows being higher than they are wide.) By default, a window prints so that its longest side follows the long side of the paper.
-psfig	Suppresses translation of the PostScript picture to the center of the page.
-report	Prints out statistics to standard error about the window *ImageFile* parameter.
-rv	Forces the window to print in reverse video.
-scale *Scale*	Affects the size of the window on the page. PostScript printers are able to translate each bit in a window pixel map into a grid of a specified size. For example, each bit might translate into a 3 x 3 grid. To specify a 3 x 3 grid, enter **-scale 3**. By default, a window prints with the largest scale that fits on the page for the specified orientation. If you do not specify a device, the aspect ratio can vary.
-split *Number*	Splits a window into several pages. This might be necessary for very large windows that would otherwise cause the printer to overload and print the page in an obscure manner. (This flag is not supported on PostScript or HP Laserjet printers.)
-top *Inches*	Specifies the top margin for the window in inches. Fractions are acceptable. By default, this flag prints the window on the center of the page.
-trailer *String*	Specifies a trailer string to print below the window.
-width *Inches*	Specifies the maximum width of the page.

Note: The 4207, 5201, and 5202 printers' images must be recorded by the **xwd** utility in XYPixmap or XYBitmap format. XYPixmap images are converted into bitmaps using a thresholding algorithm. For the HP Laserjet printer, multiplane images must be recorded in ZPixmap format. Single plane images may be either XYPixmap, XYBitmap, or ZPixmap formats.

xpreview Command

Purpose

Displays troff files on an X display.

Syntax

xpreview [**-BackingStore** *BackingStoreType*] [**-page** *Number*] [*ToolKitFlag ...*] { *File* | **-** }

Description

The **xpreview** command is an AIXwindows 2.1- and Motif2.1-based application that displays output from the **troff** command on an AIXwindows display. The **troff** command output file must be prepared for the devX100 device.

The user interface contains the standard AIXwindows interface controls for calling the root menu, iconifying the window, and setting the window to full screen size. The interface also includes a main window with a scrollable display area for text. Use the pushbuttons for Next, Previous, Goto Page, Print Page, Print File, and Newfile to manipulate the viewing document.

Mouse button three actuates a popup menu for configuring print capabilities. The menu includes an option to set the command line and another to select a printer queue. The command line dialog box expects command line input through the **troff** command. For example,

```
pic -Tibm3816 troff-input-file |tbl|troff -mm -Tibm3816
```

is an acceptable command line. The printer queue option displays a list of configured printer queues. If this option is not selected, the **xpreview** command uses the system-defined default queue.

When you are previewing an input file, the Print Page and Print File buttons require command line input. Note that once a printer queue is selected, it remains selected for the duration of the viewing session, or until an alternate printer queue is selected.

Fonts supported for the devX100 device in European locales are:

- Times New Roman in normal, italic, and bold
- Courier in normal and bold
- Helvetica in normal and bold
- Symbol

The **xpreview** command supports the following font sizes: 8, 10, 14, 18, 24, 30, and 36.

The **xpreview** command does not display files resulting from the **troff** command constructed for a device other than those described in this document.

To preview a file on a certain device, the **xpreview** command requires the fonts found in the following directories:

- **/usr/lib/X11/fonts** directory for files formatted for font files other than Japanese
- **/usr/lib/X11/fonts/JP** for Japanese font files

Multibyte Support

The **xpreview** command supports multibyte locales. Also, to display Japanese characters, Japanese 16-dot fonts (part of the Japanese BSL package) and 24- and 32-dot fonts (part of the AIXwindows font package) must be installed. To display Korean characters, Korean fonts (part of the Korean BSL package) must be installed.

Japanese support currently includes the following font sets:

- In 16-dot: RomanKn12, Kanji12, and IBM_JPN12
- In 24-dot: RomanKn17, Kanji17, and IBM_JPN17
- In 32-dot: RomanKn23, Kanji23, and IBM_JPN23, or RomanKn23G, Kanji23G, and IBM_JPN23G

Korean support currently includes the following font sets:

- In 16-dot, EnglHg16 and Hangul16
- In 24-dot, EnglHg24 and Hangul24

Flags

The **xpreview** command accepts the standard **X** Toolkit command line flags, as well as the following flags:

Item	Description
-	Requires input to be read from standard input.
-help	Indicates that a brief summary of the allowed command line flags should be printed.
-BackingStore *BackingStoreType*	The **-BackingStore** flag causes the server to save the window contents so that when it is scrolled around the viewport, the window is painted from contents saved in server backing store. Redisplays of the drawing window can take up to a second or so. The *BackingStoreType* parameter can have one of the following values: **Always**, **WhenMapped** or **NotUseful**. **Tip:** Enter a space between the **-BackingStore** flag and its *BackingStoreType* parameter. **Requirement:** Use of this flag requires that the server be started with backing store enabled.
-page *Number*	Specifies the page number of the document to be first displayed.
ToolKitFlag	The following standard **X** Toolkit flags are commonly used with the **xpreview** command: **-bg *Color*** Specifies the color to use for the background of the window. The default is white. **-bg *Color*** Specifies the color to use for the background of the window. The default is white. **-fg *Color*** Specifies the color to use for displaying text. The default is black. **-geometry *Geometry*** Specifies the preferred size and position of the window. **-display *Host*:*Display*** Specifies the **X** server to contact. **-xrm *ResourceString*** Specifies a resource string to be used.
File	Specifies the file to be printed.

Examples

1. To build files output by the **troff** command into files that are suitable for use with the **xpreview** command, enter the following commands:

```
troff-TX100 troff-input | xpreview
pic -TX100 pic-troff-input | tbl | troff -man -TX100 | xpreview
```

2. To build files output by the **troff** command into files that are suitable for use with the Japanese language version of the **xpreview** command, enter the following commands:

```
LANG=ja_JP
troff -TX100 troff-input | xpreview -
```

```
pic -TX100 pic-troff-input | tbl | troff -man -TX100 \
        | xpreview -
```

Files

Item	Description
/usr/lib/X11/app-defaults/XPreview	Contains user-configurable applications defaults file.
/usr/lib/X11/Ja_JP/app-defaults/XPreview	Contains user-configurable applications default file for the Japanese (IBM-943) locale.
/usr/lib/X11/ja_JP/app-defaults/XPreview	Contains user-configurable applications default file for the Japanese (IBM-eucJP) locale.
/usr/lib/X11/ko_KR/app-defaults/XPreview	Contains user-configurable applications default file for the Korean locale.
/usr/lib/X11/zh_TW/app-defaults/XPreview	Contains user-configurable applications default file for the Traditional Chinese locale.
/usr/lib/font/devX100	Contains troff fonts for devX100 devices.
/usr/lib/X11/fonts	Contains X fonts for 100 dpi devices.
/usr/lib/X11/fonts/JP	Contains X fonts for multi-byte characters.
/usr/lib/X11/fonts/JP	Contains X fonts for Japanese characters.

xprofiler Command

Purpose

Starts Xprofiler, a GUI-based AIX performance profiling tool.

Syntax

`xprofiler` [*program*] [`-b`] [`-s`] [`-z`] [`-a` *path*] [`-c` *file*] [`-L` *pathname*] [[`-e` *function*]...] [[`-E` *function*]...] [[`-f` *function*]...] [[`-F` *function*]...] [`-disp_max` *number_of_functions*] [[`gmon.out`]...]

`xprofiler` `-h` | `-help`

Description

The `xprofiler` command invokes Xprofiler, a GUI-based AIX performance profiling tool. Xprofiler is used to analyze the performance of both serial and parallel applications. Xprofiler uses data collected by the `-pg` compiling option and presents a graphical representation of the functions in the application in addition to providing textual data in several report windows. These presentation formats are intended to identify the functions which are most processor-intensive.

Flags

Item	Description
-a	To specify an alternate search path or paths for library files and source code files. If more than one path is specified, the paths must be embraced by "," and each path should be separated by either ":" or space.
-b	Suppresses the printing of the field descriptions for the Flat Profile, Call Graph Profile, and Function Index reports when they are written to a file with the Save As option of the File menu.
-c	Loads a configuration file that contains information to be used to determine which functions will be displayed when Xprofiler is brought up.
-disp_max	Sets the number of function boxes that Xprofiler initially displays in the function call tree. The value supplied with this flag can be any integer between 0 and 5,000. Xprofiler displays the function boxes for the most processor-intensive functions through the number you specify. For instance, if you specify 50, Xprofiler displays the function boxes for the 50 functions in your program that consume the most processor. After this, you can change the number of function boxes that are displayed via the Filter menu options. This flag has no effect on the content of any of the Xprofiler reports.
-e	De-emphasizes the general appearance of the function box or boxes for the specified functions in the function call tree, and limits the number of entries for these function in the Call Graph Profile report. This also applies to the specified function's descendants, as long as they have not been called by non-specified functions. In the function call tree, the function boxes for the specified functions appear greyed-out. Its size and the content of the label remain the same. This also applies to descendant functions, as long as they have not been called by non-specified functions. In the Call Graph Profile report, an entry for the specified function only appears where it is a child of another function, or as a parent of a function that also has at least one non-specified function as its parent. The information for this entry remains unchanged. Entries for descendants of the specified function do not appear unless they have been called by at least one non-specified function in the program.
-E	Changes the general appearance and label information of the function box or boxes for the specified functions in the function call tree. Also limits the number of entries for these functions in the Call Graph Profile report, and changes the processor data associated with them. These results also apply to the specified function's descendants, as long as they have not been called by non-specified functions in the program. In the function call tree, the function box for the specified function appears greyed-out, and its size and shape also changes so that it appears as a square of the smallest allowable size. In addition, the processor time shown in the function box label, appears as 0 (zero). The same applies to function boxes for descendant functions, as long as they have not been called by non-specified functions. This option also causes the processor time spent by the specified function to be deducted from the left side processor total in the label of the function box for each of the specified function's ancestors. In the Call Graph Profile report, an entry for the specified function only appears where it is a child of another function, or as a parent of a function that also has at least one non-specified function as its parent. When this is the case, the time in the self and descendants columns for this entry is set to 0 (zero). In addition, the amount of time that was in the descendants column for the specified function is subtracted from the time listed under the descendants column for the profiled function. As a result, be aware that the value listed in the % time column for most profiled functions in this report will change.

Item	Description
-f	De-emphasizes the general appearance of all function boxes in the function call tree, except for that of the specified function(s) and its descendant(s). In addition, the number of entries in the Call Graph Profile report for the non-specified functions and non-descendant functions is limited. The -f flag overrides the -e flag. In the function call tree, all function boxes except for that of the specified function(s) and it descendant(s) appear greyed-out. The size of these boxes and the content of their labels remain the same. For the specified function(s), and it descendants, the appearance of the function boxes and labels remain the same. In the Call Graph Profile report, an entry for a non-specified or non-descendant function only appears where it is a parent or child of a specified function or one of its descendants. All information for this entry remains the same.
-F	Changes the general appearance and label information of all function boxes in the function call tree except for that of the specified function(s) and its descendants. In addition, the number of entries in the Call Graph Profile report for the non-specified and non-descendant functions is limited, and the processor data associated with them is changed. The -F flag overrides the -E flag. In the function call tree, the function box for the specified function appears greyed-out, and its size and shape also changes so that it appears as a square of the smallest allowable size. In addition, the processor time shown in the function box label, appears as 0 (zero). In the Call Graph Profile report, an entry for a non-specified or non-descendant function only appears where it is a parent or child of a specified function or one of its descendants. The time in the self and descendants columns for this entry is set to 0 (zero). When this is the case, the time in the self and descendants columns for this entry is set to 0 (zero). As a result, be aware that the value listed in the % time column for most profiled functions in this report will change.
-h -help	Writes the Xprofiler usage to STDERR and then exits. The information includes xprofiler command line syntax and a description of Xprofiler runtime options.
-L	Uses an alternate path name for locating shared libraries. If you plan to specify multiple paths, use the Set File Search Paths option of the File menu on the Xprofiler GUI.
-s	If multiple gmon.out files are specified when Xprofiler is started, produces the gmon.sum profile data file. The gmon.sum file represents the sum of the profile information in all the specified profile files. Note that if you specify a single gmon.out file, the gmon.sum file contains the same data as the gmon.out file.
-z	Includes functions that have both zero processor usage and no call counts in the Flat Profile, Call Graph Profile, and Function Index reports. A function will not have a call count if the file that contains its definition was not compiled with the -pg option, which is common with system library files.

Example

To use xprofiler, you must first compile your program (for example, foo.c) with -pg:

```
xlc -pg -o foo foo.c
```

1. When the program foo is executed, one gmon.out file will be generated for each processor involved in the execution. To invoke xprofiler, enter:

   ```
   xprofiler foo [[gmon.out]...]
   ```

Files

Item	Description
/usr/lib/X11/app-defaults/Xprofiler	Location of the xprofiler command.

xrdb Command

Purpose

X Server resource database utilities.

Syntax

xrdb [**-display** *Display*] [**-help**] [**-quiet**] [**-retain**] [**-cpp** *FileName* | **-nocpp**] [**-D** *Name*=*Value*] [**-I** *Directory*] [**-U** *Name*] [**-all** | **-global** | **-screen** | **-screens**] [**-n**] [**-edit** *FileName* | [**-backup** *String*] | **-merge** [*FileName*] | **-load** [*FileName*] | **-query** | **-remove** | **symbols**] **-override**]

Description

The **xrdb** command gets or sets the contents of the RESOURCE_MANAGER property on the root window of screen 0 or the SCREEN_RESOURCES property on the root window of any or all screens, or everything combined. You normally run this program from your X startup file.

Most X clients use the RESOURCE_MANAGER and SCREEN_RESOURCES properties to get user preferences about color, fonts, and so on for applications. Having this information in the server (where it is available to all clients) instead of on disk solves the problem in previous versions of X that required you to maintain *defaults* files on every machine that you might use. It also allows for dynamic changing of defaults without editing files.

The RESOURCE_MANAGER property specifies resources that apply to all screens of the display. The SCREEN_RESOURCES property on each screen specifies additional (or overriding) resources to be used for that screen. (When there is only one screen, SCREEN_RESOURCES is normally not used; all resources are placed in the RESOURCE_MANAGER property.)

For compatibility, if there is no RESOURCE_MANAGER property defined (either because the **xrdb** command was not run or if the property was removed), the resource manager looks for a file called **.Xdefaults** in your home directory.

The file name (or the standard input if - or no file name is given) is optionally passed through the C preprocessor with the following symbols defined, based on the capabilities of the server being used:

Item	Description
SERVERHOST=*Hostname*	Specifies the hostname portion of the display to which you are connected.
SRVR_*name*	Turns the **SERVERHOST** hostname string into a legal identifier. For example `my-dpy.lcs.mit.edu` becomes `SRVR_my_dpy_lcs_mit_edu`.
HOST=*Hostname*	Specifies the hostname portion of the display to which you are connected.
DISPLAY_NUM=*num*	Specifies the number of the display on the server host.
CLIENTHOST=*Hostname*	Specifies the name of the host on which xrdb is running.
CLNT_*name*	Turns the **CLIENTHOST** hostname string into a legal identifier. For example `expo.lcs.mit.edu` becomes `CLNT_expo_lcs_mit_edu`.
WIDTH=*Number*	Specifies the width of the default screen in pixels.
HEIGHT=*Number*	Specifies the height of the default screen in pixels.
X_RESOLUTION=*Number*	Specifies the x resolution of the default screen in pixels per meter.

Item	Description
Y_RESOLUTION=*Number*	Specifies the y resolution of the default screen in pixels per meter.
PLANES=*Number*	Specifies the number of bit planes (the depth) of the root window of the default screen.
RELEASE=*Number*	Specifies the vendor release number for the server. The interpretation of this number varies depending on **VENDOR**.
REVISION=*Number*	Specifies the X protocol minor version supported by this server (currently 0).
VERSION=*Number*	Specifies the X protocol major version supported by this server (should always be 11).
VENDOR=*Vendor*	A string specifying the vendor of the server.
VNDR_*name*	Turns the **VENDOR** name string into a legal identifier. For example `MIT X Consortium` becomes `VNDR_MIT_X_Consortium`.
EXT_*name*	Turns each extension string into a legal identifier. A symbol is defined for each protocol extension supported by the server. For example `X3D-PEX` becomes `EXT_X3D_PEX`.
NUM_SCREENS=*num*	Specifies the total number of screens.
SCREEN_NUM=*num*	Specifies the number of current screen. from 0 (zero).
BITS_PER_RGB=*Number*	Specifies the number of significant bits in an RGB color specification. This is the log base 2 of the number of distinct shades of each primary that the hardware can generate. Note that it is not related to **PLANES**.
CLASS=*VisualClass*	Specifies the visual class of the root window of the default screen which is one of the following:
CLASS_*visualclass*=*visualid*	Specifies the visual class of the root window in a form can *#ifdef* on. The value is the numeric id of the visual.
	DirectColor, **GrayScale**, **PseudoColor**, **StaticColor**, **StaticGray**, **TrueColor**
CLASS_*visualclass_depth*=*num*	A symbol is defined for each visual supported for the screen. The symbol includes the class of the visual and its depth; the value is the numeric id of the visual. (If more than one visual has the same class and depth, the numeric id of the first one reported by the server is used.)S
COLOR	Defined only if **CLASS** is one of **StaticColor**, **PseudoColor**, **TrueColor**, or **DirectColor**.

Comment lines begin with an ! (exclamation mark) and are ignored.

Since **xrdb** can be read from standard input, use it to change the contents of properties directly from a terminal or from a shell script.

Flags

Item	Description
-all	Indicates that operation is performed on the screen-independent resource property (RESOURCE_MANAGER), as well as the screen-specific property (SCREEN_RESOURCES) on every screen of the display. For example, when used in conjunction with **-query**, the contents of all properties are output. For **-load** and **-merge**, the input file is processed once for each screen. The resources that occur in common in the output for every screen are collected and applied as the screen-independent resources. The remaining resources are applied for each individual per-screen property. This is the default mode of operation. This option is specific to X11R5.
-backup *String*	Specifies a suffix to append to the file name. Use it with **-edit** to generate a backup file. **-edit** is a prerequisite for **-backup** *String.*
-cpp *FileName*	Specifies the pathname of the C preprocessor program to use. Although the **xrdb** command was designed to use CPP, any program that acts as a filter and accepts the **-D**, **-I**, and **-U** flags can be used.
-D*Name***=***Value*	Passes through to the preprocessor and defines symbols to use with conditionals such as `#ifdef`.
-display *Display*	Specifies the X Server to use. It also specifies the screen to use for the **-screen** option, and it specifies the screen from which preprocessor symbols are derived for the **-global** option.
-edit *FileName*	Indicates that the contents of the specified properties should be edited into the given file, replacing any values listed there. This allows you to put changes you made to your defaults back into your resource file, preserving any comments or preprocessor lines.
-global	Indicates that the operation should only be performed on the screen-independent RESOURCE_MANAGER property. This option is specific to X11R5.
-help	Prints a brief description of the allowed flags.
-I*Directory*	(uppercase i) Passes through to the preprocessor and specifies a directory to search for files referenced with `#include`.
-load	Indicates that the input is loaded as the new value of the specified properties, replacing the old contents. This is the default action.
-merge	Indicates that the input merges with, instead of replaces, the current contents of the specified properties. This option performs a lexicographic sorted merge of the two inputs, which is probably not what you want, but remains for backward compatibility.
-n	Indicates that changes to the specified properties (when used with **-load** or **-merge**) or to the resource file (when used with **-edit**) should be shown on the standard output, but should not be performed. This option is specific to X11R5.
-nocpp	Indicates that the **xrdb** command should not run the input file through a preprocessor before loading it into properties.
-override	Indicates that the input should be added to, instead of replacing, the current contents of the specified properties. New entries override previous entries.
-query	Indicates that the current contents of the specified properties should print onto the standard output. Note that since preprocessor commands in the input resource file are part of the input file, not part of the property, they do not appear in the output from this flag.

Item	Description
-quiet	Indicates that a warning about duplicate entries should not display. This option is specific to X11R5.
-remove	Indicates that the specified properties should be removed from the server.
-retain	Indicates that the server should be instructed not to reset if the **xrdb** command is the first client. This should never be necessary under normal conditions, since the **xdm** and **xinit** commands always act as the first client. This option is specific to X11R5.
-screen	Indicates that the operation should only be performed on the SCREEN_RESOURCES property of the default screen of the display. This option is specific to X11R5.
-screens	Indicates that the operation should be performed on the SCREEN_RESOURCES property of each screen of the display. For **-load** and **-merge**, the input file is processed once for each screen. This option is specific to X11R5.
-symbols	Indicates that the symbols defined for the preprocessor should be printed onto the standard output.
-U*Name*	Passes through to the preprocessor and removes any definitions of this symbol.

Examples

1. To load a file into the database:

```
xrdb -load myfile
```

2. To take the contents of the database just loaded and edit or put it into newfile:

```
xrdb -edit newfile
```

Files

The **xrdb** command generalizes the **~/.Xdefaults** files.

xsend Command

Purpose

Sends secret mail in a secure communication channel.

Syntax

xsend *User*

Description

The **xsend** command sends messages that can be read only by the intended recipient. This command is similar to the **mail** command, but the mail sent with this command is intended to be secret.

The **xsend** command is used with the **enroll** command and the **xget** command to send secret mail. The **enroll** command sets up the password used to receive secret mail. The **xget** command uses that password to receive the mail.

The **xsend** command reads standard input until an EOF (Ctrl-D) or a . (period) is entered. It then encrypts this text along with some header information and sends it. After sending the encrypted message, the **xsend** command mails a standard mail message to the recipient informing them they have received secret mail.

Note: Secret mail can only be sent to local users.

Examples

1. To send secret mail, enter:

```
xsend ron
```

When you have issued the **xsend** command with the recipient's name, the mail system is used to enter the text of the message. When you finish entering the message to user `ron`, press the Enter key, then Ctrl-D or a . (period) to exit the mail editor and send the message. The **xsend** command encrypts the message before it is sent.

2. To send a file to another user, enter:

```
xsend lance <proposal
```

In this example, the file `proposal` is sent to user `lance`.

Files

Item	Description
/var/spool/secretmail/*.keys	Contains the encrypted key for *User.*
/var/spool/secretmail/*.[0-9]	Contains the encrypted mail messages for *User*.
/usr/bin/xsend	Contains the command executable files.

xset Command

Purpose

Sets options for your X-Windows environment.

Syntax

xset [**-display** *Display*] [**b** [*Volume* [*Pitch* [*Duration*]]] | **-b** | **b on** | **b off**] [**bc** | **-bc**] **c** [*Volume*] | **-c** | **c on** | **c off**] [[**-** | **+**] **fp** [**-** | **+** | **=**] **Path** [**,***Path***,** [*...*]]] [**fp default**] [**fp rehash**] [[**-**] **led** [*Integer*]] [**led on** | **led off**] [**m** [*Accelerator*] [*Threshold*]]] [**m** [**ouse**] **default**] [**p** *Pixel Color*] [[**-**] **r**] [**r on** | **r off**] [**s** [*Length* [*Period*]]] [**s blank** | **s noblank**] [**s expose** | **s noexpose**] [**s on** | **s off**] [**s activate**] [**s reset**] [**s default**] [**q**]

Description

The **xset** command customizes your X-Windows environment.

Flags

Item	Description
-display *Host***:***Display*	Specifies the X server to use. For more information about servers, see the **X** command.

Item	Description
b or **b on**	Turns the bell on. This is the default setting. **Note:** Not all hardware is able to vary the bell characteristics, but for that which can, all of the **b** flag permutations and its variables are available.
b [*Volume* [*Pitch* [*Duration*]]]	Specifies the bell volume, pitch, and duration. This flag accepts up to three numeric values. ***Volume*** If only one numeric is given then it is assumed to be *Volume*. The bell volume is set to that numeric as a percentage of the bell's maximum possible volume dependent on current hardware capabilities. ***Pitch*** The second numeric in hertz values, is the tonal sound of the bell. ***Duration*** The third numeric in milliseconds, is the length of time that the bell rings.
-b or **b off**	Turns the bell off.
bc or **-bc**	Controls bug compatibility mode in the server, if possible. A preceding **-** (dash) disables this mode; otherwise, bug compatibility mode is enabled. The server must support the MIT-SUNDRY-NONSTANDARD protocol extension for the **bc** flag to work. New application development should be performed with bug compatibility mode disabled. The **bc** flag is provided for pre-X11 Release 4 (X11R4) clients. Some pre-X11R4 clients pass illegal values in various protocol requests. Such clients, when run with an X11R4 server, end abnormally or otherwise fail to operate correctly. This flag explicitly reintroduces certain bugs into the X server so that such clients still can be run.
c or **c on**	Turns on the click. System default.
c *Volume*	A numeric from 0 to 100 that specifies a percentage of the click's maximum possible volume dependent on current hardware capabilities.
-c or **c off**	Turns off the click.

Item	Description
fp=*Path***,...**	Sets the font path to the directories given in the *Path* parameter. The directories are interpreted by the server, not by the client, and are server-dependent. The server ignores directories that do not contain font databases created by the **mkfontdir** command. All of the options and variables supported by the **fp** flag are available.
fp- or **-fp**	Deletes the font path specified by the *Path* parameter from the end of the current font path if the **-** (dash) precedes **fp** and from the front of the font path if the **-** (dash) follows **fp.**
fp+ or **+fp**	Adds the font path specified by the *Path* parameter to the bottom of font list if the **-** (dash) precedes **fp** and from the end of the font path if the **-** (dash) follows **fp**.
fp default	Resets the font path to the server's default.
fp rehash	Causes the server to reread the font databases in the current font path. Usually used only when adding new fonts to a font directory after running **mkfontdir** to recreate the font database.
led or**led on**	Turns all LEDs on.
-led *Integer*	Turns the LED specified by *Integer* off. Valid values are between 1 and 32.
led *Integer*	Turns the LED specified by *Integer* on. Valid values are between 1 and 32.
-led or **led off**	Turns all LEDs off. **Note:** Not all hardware assigns the same *Integer* variables to the same LED functions.
m	Allows you to control the precision of the mouse or other pointing device. If no variable or the **default** argument is specified, the system defaults are used. This flag accepts the following optional arguments and parameters: ***Acceleration*** Sets the multiplier for the mouse movement. The value can be specified as an integer or a fraction. ***Threshold*** Sets the minimum number of pixels needed to invoke a movement of the mouse. The value is specified in pixels. If only one parameter is given, it will be interpreted as the *Acceleration* parameter. **default** Uses the system defaults.

Item	Description
p	Controls pixel color values. The root background colors may be changed on some servers by altering the entries for BlackPixel and WhitePixel. Although these values are often **0** and **1**, they need not be. Also, a server may choose to allocate those colors privately, in which case the **xset** command generates an error. The **xset** command also generates an error if the map entry is a read-only color. Valid parameters are: ***Pixel*** Specifies the color map entry number in decimal. ***Color*** Specifies a color.
r or **r on**	Enables autorepeat.
-r or **r off**	Disables autorepeat.
s or **s default**	Sets screen saver parameters to the default screen-saver characteristics.
s [*Length*[*Period*]]	Specifies the length of time the server must be inactive for the screen saver to activate. *Period* specifies the period in which the background pattern must be changed to avoid burn in. The values of *Length* and *Period* are specified in seconds. If only one numerical parameter is given, it is read as a *Length* parameter.
s on or **s off**	Turns the screen saver functions on and off, respectively.
s activate	Causes the screen saver to activate, even if it has been turned off.
s reset	Causes the screen saver to deactivate if it was activated.
s blank	Sets the preference to blank the video (if the hardware can do so) rather than display a background pattern.
s noblank	Sets the preference to display a pattern rather than blank the video.
s expose	Sets the preference to allow window exposures (the server can freely discard window contents).
s noexpose	Sets the preference to disable screen saver unless the server can regenerate the screens without causing exposure events.

Item	Description
q	Reports information on the current settings. These settings will be reset to default values when you log out. **Note:** Not all X implementations are guaranteed to honor all of these options.

Examples

1. To set the bell volume to medium, the tone to 50 hertz, and length of time the bell rings to 50 milliseconds:

```
xset b 50,50,50
```

2. To set the font path to the **/usr/lib/X11/fonts** directory:

```
xset fp= /usr/lib/x11/fonts
```

3. To cause the server to reread the font databases in the current font path:

```
xset fp rehash
```

4. To see information on the current settings:

```
xset q
```

which produces output similar to the following:

```
Keyboard Control:
  auto repeat:  on    key click percent:  0    LED mask:  00000000
  auto repeating keys:  0000000000000000
                        0000000000000000
                        0000000000000000
                        0000000000000000
  bell percent:  50    bell pitch:  400    bell duration:  100

Pointer Control:
  acceleration:  2 = 2 / 1    threshold:  4

Screen Saver:
  prefer blanking:  no    allow exposures:  no
  timeout:  0    cycle:  0

Colors:
  default colormap:  0x8006e    BlackPixel:  0    WhitePixel:  1

Font Path:
   /usr/lib/X11/fonts/,/usr/lib/X11/fonts/75dpi/,/usr/lib/X11/fonts/100dpi/,/usr/
lib/X11/fonts/oldx10/,/usr/lib/X11/fonts/oldx11/,/usr/lib/X11/fonts/bmug/,/usr/l
ib/X11/fonts/info-mac/,/usr/lib/X11/fonts/JP/,/usr/lib/X11/fonts/misc/
```

xsetroot Command

Purpose

Sets the root window parameters for the **X** command.

Syntax

xsetroot [**-bg** *Color*] [**-cursor** *CursorFile MaskFile*] [**-cursor_name** *CursorName*] [**-def**] [**-display** *Display*] [**-fg** *Color*] [**-help**] [**-name** *String*] [**-rv**] [**-bitmap** *FileName* | **-gray** | **-grey** | **-mod** *X Y* | **-solid** *Color*]

Description

The **xsetroot** command allows you to tailor the appearance of the background (root) window on a workstation display running X. Normally, you experiment with the **xsetroot** command until you find a personalized look that you like, then put the **xsetroot** command that produces it into your X startup file. If no options are specified or if the **-def** flag is specified, the window is reset to its default state. The **-def** flag can be specified with other flags and only the unspecified characteristics are reset to the default state.

Only one of the background color (tiling) changing flags (**-bitmap**, **-solid**, **-gray**, **-grey**, or **-mod**) can be specified at a time.

Flags

Item	Description
-bg *Color*	Uses the *Color* parameter as the background color.
-bitmap *FileName*	Uses the bitmap specified in the file to set the window pattern. You can make your own bitmap files (little pictures) using the **bitmap** program. The entire background is made of repeated tiles of the bitmap.
-cursor *CursorFile MaskFile*	Changes the pointer cursor to what you want when it is outside of any window. Cursor and mask files are bitmaps (little pictures) that can be made with the **bitmap** program. You probably want the mask file to be all black until you get used to the way masks work.
-cursor_name *CursorName*	Changes the pointer cursor to one of the standard cursors from the cursor font.
-def	Resets unspecified attributes to the default values. (Restores the background to the familiar gray mesh and the cursor to the hollow x shape.)
-display *Display*	Specifies the server connection. See the **X** command.
-fg *Color*	Uses the *Color* parameter as the foreground color. Foreground and background colors are meaningful only with the **-cursor**, **-bitmap**, or **-mod** flags.
-gray	Makes the entire background gray.
-grey	Makes the entire background grey.
-help	Prints a usage message and exits.
-mod *X Y*	Makes a plaid-like grid pattern on your screen. The *X* and *Y* parameters are integers ranging from 1 to 16. Zero and negative numbers are taken as 1.
-name *String*	Sets the name of the root window to the *String* parameter. There is no default value. Usually a name is assigned to a window so that the window manager can use a text representation when the window is iconified. This flag is not used because you cannot iconify the background.
-rv	Exchanges the foreground and background colors. Normally the foreground color is black and the background color is white.
-solid *Color*	Sets the background of the root window to the specified color. This flag is only used on color servers.

xss Command

Purpose

Improves the security of unattended workstations.

Syntax

xss [**-e** *CommandString*] [**-timeout** *Seconds*] [**-display** *DisplayPtr*] [**-v**] [**-fg** *Color*] [**-bg** *Color*] [**-geometry** *wxh+x+y*]

Description

The **xss** command works with the newly added Massachusetts Institute of Technology (MIT) Screen Saver Extensions in order to implement a user controllable screen saver/lock. This command is designed to improve the security of unattended workstations.

The **xss** command executes a user-specified command string when it receives a screen saver timeout message, or when the user activates the pushbutton. When no user-specified command is given, the **xss** command defaults to the **xlock** command.

> **Note:** The **xss** command only uses the newly added MIT Screen Saver Extensions. The **xss** command does not work on an older X server, or when using an older X extension library.

Flags

Item	Description
-e *CommandString*	Sets the **xss** command to execute when either the screen saver times out, or the user activates the pushbutton. Note that if the *CommandString* parameter value is longer than one word, it must be surrounded by " " (double quotations).
-timeout *Seconds*	Sets the number of seconds of user inactivity before the screen saver times out, and causes the **xss** command to run the *CommandString* parameter.
-display *DisplayPtr*	Sets the connection to the X11 display.
-v	Turns on verbose mode.
-fg *Color*	Sets the foreground color of the pushbutton.
-bg *Color*	Sets the background color of the pushbutton.
-geometry *wxh+x+y*	Specifies the size and location of the client window.

Examples

When running remotely and using the **-display** flag for the **xss** command, remember that you may also have to use the **-display** flag option for the command that will be executed by the **xss** command. See the following running remote example:

1. Running remote:

```
xss -display myhost:0 -e "xlock -remote -display myhost:0"
```

2. Screen saver only:

```
xss -e "xlock -nolock"
```

3. Simple example:

```
xss -e xlock
```

xstr Command

Purpose

Extracts strings from C programs to implement shared strings.

Syntax

xstr [**-v**] [**-c**] [**-**] [*File*]

Description

The **xstr** command maintains a file **strings** into which strings in component parts of a large program are hashed. These strings are replaced with references to this array. This serves to implement shared constant strings, most useful if they are also read-only.

The command:

```
xstr -c File
```

extracts the strings from the C source in the *File* parameter, replacing string references by expressions of the form (**&xstr**[*number*]) for some number. An appropriate declaration of the xstr array is prepended to the file. The resulting C text is placed in the file **x.c**, to then be compiled. The strings from this file are appended into the **strings** file if they are not there already. Repeated strings and strings which are suffixes of existing strings do not cause changes to the file **strings**.

If a string is a suffix of another string in the file but the shorter string is seen first by the **xstr** command, both strings are placed in the file **strings**.

After all components of a large program have been compiled, a file **xs.c** declaring the common xstr array space can be created by a command of the form:

```
xstr
```

This **xs.c** file should then be compiled and loaded with the rest of the program. If possible, the array can be made read-only (shared), saving space and swap overhead.

The **xstr** command can also be used on a single file. The command:

```
xstr File
```

creates files **x.c** and **xs.c** as before, without using or affecting any **strings** file in the same directory.

It may be useful to run the **xstr** command after the C preprocessor if any macro definitions yield strings or if there is conditional code which contains strings which may not, in fact, be needed.

The **xstr** command reads from its standard input when the **-** (minus sign) flag is given and does not alter the **strings** file unless the **-c** flag is specified also.

An appropriate command sequence for running the **xstr** command after the C preprocessor is:

```
cc -E name.c | xstr -c -
cc -c x.c
mv x.o name.o
```

The **xstr** command does not touch the file **strings** unless new items are added, thus the **make** command can avoid remaking the **xs.o** file unless truly necessary.

Flags

Item	Description
-c	Extracts strings from the specified file, and places them in the **strings** file.
-v	Verbose mode. Tells when strings are found, or new in the **strings** file.
-	Reads from standard input.

Examples

1. To extract the strings from the C source in the *File.c* parameter, replacing string references by expressions of the form (**&xstr**[*number*]):

   ```
   xstr -c File.c
   ```

   An appropriate declaration of the xstr array is prepended to the file. The resulting C text is placed in the file **x.c**, to then be compiled.
2. To declare the common xstr array space in the **xs.c** file:

   ```
   xstr
   ```

Files

Item	Description
strings	File which contains the extracted strings.
x.c	Massaged C source.
xs.c	C source for definition of array xstr.
/tmp/xs*	Temporary file when **xstr** command does not touch the **strings** file.
/usr/ccs/bin/mkstr	Contains an executable file.
/**usr/ccs/bin/mkstr**	Contains an executable file for Berkeley environment.

xterm Command

Purpose

Provides a terminal emulator for the X Window System.

Note: The **xterm** command is ported from the Massachusetts Institute of Technology (MIT) X Window System, Version 11, Release 6 with no functional enhancements. The **xterm** command does not have support for globalization. For the localized and internationalized terminal emulator, the user can use the **aixterm** or **dtterm** commands.

Syntax

xterm [*--Xtoolkitoption*...] [*-Option* ...]

Description

The **xterm** program is a terminal emulator for the X Window System. It provides DEC VT102 and Tektronix 4014 compatible terminals for programs that cannot use the window system directly. If the underlying operating system supports terminal resizing capabilities, the **xterm** program uses the facilities to notify programs running in the window whenever it is resized.

The VT102 and Tektronix 4014 terminals each have their own window so that you can edit text in one and look at graphics in the other at the same time. To maintain the correct aspect ratio (height/width), Tektronix graphics are restricted to the largest box with a 4014 aspect ratio that will fit in the window. This box is located in the upper left area of the window.

Although both windows might be displayed at the same time, one of them is considered the *active window* for receiving keyboard input and terminal output. This is the window that contains the text cursor. The active window can be chosen through escape sequences, the VT Options menu in the VT102 window, and the Tek Options menu in the 4014 window.

Emulations

The VT102 emulation is fairly complete, but does not support smooth scrolling, VT52 mode, the flashing character attribute, or the double-wide and double-size character sets. The **termcap** file entries that work with the **xterm** command include **xterm**, **vt102**, **vt100** and ``ansi,'' and the **xterm** command automatically searches the **termcap** file in this order for these entries and then sets the **TERM** and the **TERMCAP** environment variables.

Many of the special **xterm** features might be modified under program control through a set of escape sequences different from the standard VT102 escape sequences.

The Tektronix 4014 emulation is also fairly good. It supports 12-bit graphics addressing, scaled to the window size. Four different font sizes and five different lines types are supported. There is no write-thru or defocused mode support.

The Tektronix text and graphics commands are recorded internally by the **xterm** command and may be written to a file by sending the COPY escape sequence (or through the Tektronix menu, as described in the following sections). The name of the file will be **COPY***yy***-***MM***-***dd***.***hh***:***mm***:***ss*, where *yy*, *MM*, *dd*, *hh*, *mm*, and *ss* are the year, month, day, hour, minute, and second when the copy is performed (the file is created in the directory that the **xterm** command is started in, or the home directory for a login **xterm**).

Other Features

The **xterm** command automatically highlights the text cursor when the pointer enters the window (selected) and unhighlights it when the pointer leaves the window (unselected). If the window is the focus window, the text cursor is highlighted no matter where the pointer is located.

In VT102 mode, there are escape sequences to activate and deactivate an alternate screen buffer, which is the same size as the display area of the window. When activated, the current screen is saved and replaced with the alternate screen. Saving of lines scrolled off the top of the window is disabled until the usual screen is restored.

The **termcap** file entry for the **xterm** command allows the **vi** command editor to switch to the alternate screen for editing and to restore the screen on exit.

In either VT102 or Tektronix mode, there are escape sequences to change the name of the windows.

Options

The **xterm** terminal emulator accepts all of the standard X Toolkit command-line options as well as the following (if the option begins with a + instead of a -, the option is restored to its default value):

Item	Description
-help	Causes the **xterm** command to print out a message describing its options.
-132	Usually, the VT102 DECCOLM escape sequence that switches between 80- and 132-column mode is ignored. This option causes the DECCOLM escape sequence to be recognized, and the xterm window will resize appropriately.

Item	Description
-ah	Indicates that the **xterm** command should always highlight the text cursor. By default, the **xterm** command will display a hollow text cursor whenever the focus is lost or the pointer leaves the window.
+ah	Indicates that the **xterm** command should do text cursor highlighting based on focus.
-b *Number*	Specifies the size of the inner border (the distance between the outer edge of the characters and the window border) in pixels. The default is 2.
-cc *CharacterClassRange*:*Value*[**,**...]	Sets classes indicated by the given ranges for use in selecting by words.
-cn	Indicates that newlines should not be cut in line-mode selections.
+cn	Indicates that newlines should be cut in line-mode selections.
-cr *Color*	Specifies the color to use for the text cursor. The default is to use the same foreground color that is used for text.
-cu	Indicates that the **xterm** command should work around a bug in the **more** program that causes it to incorrectly display lines that are exactly the width of the window and are followed by a line beginning with a tab (the leading tabs are not displayed). This option is so named because it was originally thought to be a bug in the **curses** function cursor motion package.
+cu	Indicates that **xterm** should not work around the **more** function bug previously mentioned.
-e *Program* [*Arguments*]	Specifies the program (and its command-line arguments) to be run in the xterm window. It also sets the window title and icon name to be the base name of the program being run if neither the -**T** nor the -**n** option is given on the command line. **Note:** This must be the last option on the command line.
-fb *Font*	Specifies a font to be used when displaying bold text. This font must be the same height and width as the normal font. If only one of the normal or bold fonts is specified, it will be used as the normal font and the bold font will be produced by overstriking this font. The default is to do overstriking of the normal font.
-i	Turns on the **useInsertMode** resource.
+i	Turns off the **useInsertMode** resource.

Item	Description
-j	Indicates that the **xterm** command should do jump scrolling. Usually, text is scrolled one line at a time; this option allows the **xterm** command to move multiple lines at a time so that it does not fall as far behind. Its use is strongly recommended because it makes the **xterm** command much faster when scanning through large amounts of text. The VT100 escape sequences for enabling and disabling smooth scrolling as well as the VT Options menu can be used to turn this feature on or off.
+j	Indicates that the **xterm** command should not do jump scrolling.
-ls	Indicates that the shell that is started in the xterm window is a login shell (in other words, the first character of the *ArgumentVector* parameter is a dash, indicating to the shell that it should read the user's **.login** or **.profile** file).
+ls	Indicates that the shell that is started should not be a login shell (in other words, it will be a usual subshell).
-mb	Indicates that the **xterm** command should ring a margin bell when the user types near the right end of a line. This option can be turned on and off from the VT Options menu.
+mb	Indicates that the margin bell should not be rung.
-mc *Milliseconds*	Specifies the maximum time between multiclick selections.
-ms *Color*	Specifies the color to be used for the pointer cursor. The default is to use the foreground color.
-nb *Number*	Specifies the number of characters from the right end of a line at which the margin bell, if enabled, will ring. The default is 10.
-rw	Indicates that reverse wraparound should be allowed. This allows the cursor to back up from the leftmost column of one line to the rightmost column of the previous line. This is very useful for editing long shell command lines and is encouraged. This option can be turned on and off from the VT Options menu.
+rw	Indicates that reverse wraparound should not be allowed.

Item	Description
-aw	Indicates that auto wraparound should be allowed. This allows the cursor to automatically wrap to the beginning of the next line when it is at the rightmost position of a line and text is output.
+aw	Indicates that auto wraparound should not be allowed.

Item	Description
-s	Indicates that the **xterm** command may scroll asynchronously, meaning that the screen does not have to be kept completely up to date while scrolling. This allows the **xterm** command to run faster when network latencies are high and is typically useful when running across a large Internet or many gateways.
+s	Indicates that the **xterm** command should scroll synchronously.
-sb	Indicates that some number of lines that are scrolled off the top of the window should be saved and that a scrollbar should be displayed so that those lines can be viewed. This option can be turned on and off from the VT Options menu.
+sb	Indicates that a scrollbar should not be displayed.
-sf	Indicates that Sun Function Key escape codes should be generated for function keys.
+sf	Indicates that the standard escape codes should be generated for function keys.
-si	Indicates that output to a window should not automatically reposition the screen to the bottom of the scrolling region. This option can be turned on and off from the VT Options menu.
+si	Indicates that output to a window should cause it to scroll to the bottom.
-sk	Indicates that pressing a key while using the scrollbar to review previous lines of text should cause the window to be repositioned automatically in the usual position at the bottom of the scroll region.
+sk	Indicates that pressing a key while using the scrollbar should not cause the window to be repositioned.
-sl *Number*	Specifies the number of lines to save that have been scrolled off the top of the screen. The default is 64.
-t	Indicates that the **xterm** command should start in Tektronix mode, rather than in VT102 mode. Switching between the two windows is done using the Options menus.
+t	Indicates that the **xterm** command should start in VT102 mode.
-tm *String*	Specifies a series of terminal-setting keywords followed by the characters that should be bound to those functions, similar to the **stty** program. Allowable keywords include: **intr**, **quit**, **erase**, **kill**, **eof**, **eol**, **swtch**, **start**, **stop**, **brk**, **susp**, **dsusp**, **rprnt**, **flush**, **weras**, and **lnext**. Control characters might be specified as ^*Character* (for example, ^c or ^u), and ^? may be used to indicate Delete.

Item	Description
-tn *Name*	Specifies the name of the terminal type to be set in the **TERM** environment variable. This terminal type must exist in the **termcap** database and should have **li#** and **co#** entries.
-ut	Indicates that the **xterm** command should not write a record into the **/etc/utmp** system log file.
+ut	Indicates that the **xterm** command should write a record into the **/etc/utmp** system log file.
-vb	Indicates that a visual bell is preferred over an audible one. Instead of ringing the terminal bell whenever the Ctrl+G key sequence signal is received, the window will flash.
+vb	Indicates that a visual bell should not be used.
-wf	Indicates that the **xterm** command should wait for the window to be mapped the first time before starting the subprocess so that the initial terminal size settings and environment variables are correct. It is the application's responsibility to catch subsequent terminal size changes.
+wf	Indicates that the **xterm** command should not wait before starting the subprocess.
-C	Indicates that this window should receive console output. This is not supported on all systems. To obtain console output, you must be the owner of the console device, and you must have read and write permission for it. If you are running X windows under **xdm** on the console screen, you may need to have the session startup and reset programs explicitly change the ownership of the console device in order to get this option to work.
-S*ccn*	Specifies the last two letters of the name of a pseudoterminal to use in worker mode, plus the number of the inherited file descriptor. The option is parsed ` `%c%c%d''. This allows the **xterm** command to be used as an input and output channel for an existing program and is sometimes used in specialized applications.

The following command-line arguments are provided for compatibility with older versions. They may not be supported in the next release as the X Toolkit provides standard options that accomplish the same task.

Item	Description
%*geom*	Specifies the preferred size and position of the Tektronix window. It is shorthand for specifying the ***tekGeometry** resource.
#*geom*	Specifies the preferred position of the icon window. It is shorthand for specifying the ***iconGeometry** resource.
-T *String*	Specifies the title for the **xterm** program's windows. It is equivalent to **-title**.

Item	Description
-n *String*	Specifies the icon name for the **xterm** program's windows. It is shorthand for specifying the ***iconName** resource. Note that this is not the same as the Toolkit option **-name** (see the following). The default icon name is the application name.
-r	Indicates that reverse video should be simulated by swapping the foreground and background colors. It is equivalent to **-rv**.
-w *Number*	Specifies the width in pixels of the border surrounding the window. It is equivalent to **-borderwidth** or **-bw**.

The following standard X Toolkit command-line arguments are commonly used with the **xterm** command:

Item	Description
-bg *Color*	Specifies the color to use for the background of the window. The default is white.
-bd *Color*	Specifies the color to use for the border of the window. The default is black.
-bw *Number*	Specifies the width in pixels of the border surrounding the window.
-fg *Color*	Specifies the color to use for displaying text. The default is black.
-fn *Font*	Specifies the font to be used for displaying usual text. The default is fixed.
-name *Name*	Specifies the application name under which resources are to be obtained, rather than the default executable file name. The *Name* parameter should not contain . (dot) or * (asterisk) characters.
-title *String*	Specifies the window title string, which may be displayed by window managers if the user so chooses. The default title is the command line specified after the **-e** option, if any; otherwise, the application name.
-rv	Indicates that reverse video should be simulated by swapping the foreground and background colors.
-geometry *Geometry*	Specifies the preferred size and position of the VT102 window; see the **X** command.
-display *Display*	Specifies the X server to contact; see the **X** command.
-xrm *ResourceString*	Specifies a resource string to be used. This is especially useful for setting resources that do not have separate command-line options.
-iconic	Indicates that the **xterm** command should ask the window manager to start it as an icon rather than as the usual window.

Resources

The program understands all of the core X Toolkit resource names and classes as well as:

Item	Description
iconGeometry (class **IconGeometry**)	Specifies the preferred size and position of the application when iconified. It is not necessarily obeyed by all window managers.
termName (class **TermName**)	Specifies the terminal type name to be set in the **TERM** environment variable.

Item	Description
title (class **Title**)	Specifies a string that may be used by the window manager when displaying this application.
ttyModes (class **TtyModes**)	Specifies a string containing terminal-setting keywords and the characters to which they may be bound. Allowable keywords include: **intr**, **quit**, **erase**, **kill**, **eof**, **eol**, **swtch**, **start**, **stop**, **brk**, **susp**, **dsusp**, **rprnt**, **flush**, **weras**, and **lnext**. Control characters may be specified as ^*Character* (for example, ^c or ^u) and ^? may be used to indicate Delete. This is very useful for overriding the default terminal settings without having run an **stty** program every time an **xterm** window is started.
useInsertMode (class **useInsertMode**)	Forces the use of insert mode by adding appropriate entries to the **TERMCAP** environment variable. This is useful if the system termcap is broken. The default is **false**.
utmpInhibit (class **UtmpInhibit**)	Specifies whether **xterm** should try to record the user's terminal in **/etc/utmp**.
sunFunctionKeys (class **SunFunctionKeys**)	Specifies whether Sun Function Key escape codes should be generated for function keys instead of standard escape sequences.
waitForMap (class **WaitForMap**)	Specifies whether the **xterm** command should wait for the initial window map before starting the subprocess. The default is False.

The following resources are specified as part of the **vt100** widget (class **VT100**):

Item	Description
allowSendEvents (class **AllowSendEvents**)	Specifies whether synthetic key and button events (generated using the X protocol **SendEvent** request) should be interpreted or discarded. The default is False, meaning they are discarded. Note that allowing such events creates a large security hole.
alwaysHighlight (class **AlwaysHighlight**)	Specifies whether **xterm** should always display a highlighted text cursor. By default, a hollow text cursor is displayed whenever the pointer moves out of the window or the window loses the input focus.
appcursorDefault (class **AppcursorDefault**)	If True, the cursor keys are initially in application mode. The default is False.

Item	Description
appkeypadDefault (class **AppkeypadDefault**)	If True, the keypad keys are initially in application mode. The default is False.
autoWrap (class **AutoWrap**)	Specifies whether auto wraparound should be enabled. The default is True.
bellSuppressTime (class **BellSuppressTime**)	Specifies the number of milliseconds after a bell command is sent during which additional bells will be suppressed. The default is 200. If set to nonzero, additional bells will also be suppressed until the server reports that processing of the first bell has been completed; this feature is most useful with the visible bell.
boldFont (class **BoldFont**)	Specifies the name of the bold font to use instead of overstriking.
c132 (class **C132**)	Specifies whether the VT102 DECCOLM escape sequence should be honored. The default is False.
charClass (class **CharClass**)	Specifies comma-separated lists of character class bindings of the form [*low*-]*high*:*value*. These are used in determining which sets of characters should be treated the same when doing cut and paste. See "Character Classes" on page 4715.
curses (class **Curses**)	Specifies whether the last column bug in the **curses** function should be worked around. The default is False.
cutNewline (class **cutNewline**)	If **false**, triple clicking to select a line does not include the Newline at the end of the line. If **true**, the Newline is selected. The default is **true**.
cutToBeginningofLines (class **CutToBeginningOfLine**)	If **false**, triple clicking to select a line selects only from the current word forward. If **true**, the entire line is selected. The default is **true**.
background (class **Background**)	Specifies the color to use for the background of the window. The default is white.
foreground (class **Foreground**)	Specifies the color to use for displaying text in the window. Setting the class name instead of the instance name is an easy way to have everything that would usually be displayed in the text color to change color. The default is black.
cursorColor (class **Foreground**)	Specifies the color to use for the text cursor. The default is black.
eightBitInput (class **EightBitInput**)	If True, meta characters input from the keyboard are presented as a single character with the eighth bit turned on. If False, meta characters are converted into a 2-character sequence with the character itself preceded by **ESC**. The default is True.
eightBitOutput (class **EightBitOutput**)	Specifies whether 8-bit characters sent from the host should be accepted as is or stripped when printed. The default is True.
font (class **Font**)	Specifies the name of the normal font. The default is fixed.

Item	Description
font1 (class **Font1**)	Specifies the name of the first alternative font.
font2 (class **Font2**)	Specifies the name of the second alternative font.
font3 (class **Font3**)	Specifies the name of the third alternative font.
font4 (class **Font4**)	Specifies the name of the fourth alternative font.
font5 (class **Font5**)	Specifies the name of the fifth alternative font.
font6 (class **Font6**)	Specifies the name of the sixth alternative font.
geometry (class **Geometry**)	Specifies the preferred size and position of the VT102 window.
hpLowerleftBugCompat (class **hpLowerleftBugCompat**)	Specifies whether to work around a bug in **xdb**, which ignores termcap and always sends ESC F to move to the lower left corner. **true** causes **xterm** in interpret ESC F as a request to move to the lower left corner of the screen. The default is **false**.
internalBorder (class **BorderWidth**)	Specifies the number of pixels between the characters and the window border. The default is 2.
jumpScroll (class **JumpScroll**)	Specifies whether jump scrolling should be used. The default is True.

Item	Description
loginShell (class **LoginShell**)	Specifies whether the shell to be run in the window should be started as a login shell. The default is False.
marginBell (class **MarginBell**)	Specifies whether the bell should be run when the user types near the right margin. The default is False.
multiClickTime (class **MultiClickTime**)	Specifies the maximum time in milliseconds between multiclick select events. The default is 250 milliseconds.
multiScroll (class **MultiScroll**)	Specifies whether scrolling should be done asynchronously. The default is False.
nMarginBell (class **Column**)	Specifies the number of characters from the right margin at which the margin bell should be rung, when enabled.
pointerColor (class **Foreground**)	Specifies the foreground color of the pointer. The default is **XtDefaultForeground**.
pointerColorBackground (class **Background**)	Specifies the background color of the pointer. The default is **XtDefaultBackground**.
pointerShape (class **Cursor**)	Specifies the name of the shape of the pointer. The default is **xterm**.

Item	Description
resizeGravity (class **ResizeGravity**)	Affects the behavior when the window is resized to be taller or shorter. **NorthWest** specifies that the top line of text on the screen stays fixed. If the window is made shorter, lines are dropped from the bottom; if the window is made taller, blank lines are added at the bottom. This is compatible with the behavior in MIT version X11R4. **SouthWest** (the default) specifies that the bottom line of text on the screen stays fixed. If the window is made taller, additional saved lines will be scrolled down onto the screen; if the window is made shorter, lines will be scrolled off the top of the screen, and the top saved lines will be dropped.
reverseVideo (class **ReverseVideo**)	Specifies whether reverse video should be simulated. The default is False.
reverseWrap (class **ReverseWrap**)	Specifies whether reverse wraparound should be enabled. The default is False.
saveLines (class **SaveLines**)	Specifies the number of lines to save beyond the top of the screen when a scrollbar is turned on. The default is 64.
scrollBar (class **ScrollBar**)	Specifies whether the scrollbar should be displayed. The default is False.
scrollTtyOutput (class **ScrollCond**)	Specifies whether output to the terminal should automatically cause the scrollbar to go to the bottom of the scrolling region. The default is True.
scrollKey (class **ScrollCond**)	Specifies whether pressing a key should automatically cause the scrollbar to go to the bottom of the scrolling region. The default is False.
scrollLines (class **ScrollLines**)	Specifies the number of lines that the **scroll-back** and **scroll-forw** actions should use as a default. The default value is 1.
signalInhibit (class **SignalInhibit**)	Specifies whether the entries in the Main Options menu for sending signals to **xterm** should be disallowed. The default is False.
tekGeometry (class **Geometry**)	Specifies the preferred size and position of the Tektronix window.
tekInhibit (class **TekInhibit**)	Specifies whether the escape sequence to enter Tektronix mode should be ignored. The default is False.
tekSmall (class **TekSmall**)	Specifies whether the Tektronix mode window should start in its smallest size if no explicit geometry is given. This is useful when running the **xterm** command on displays with small screens. The default is False.
tekStartup (class **TekStartup**)	Specifies whether **xterm** should start up in Tektronix mode. The default is False.

Item	Description
titeInhibit (class **TiteInhibit**)	Specifies whether **xterm** should remove **ti** and **te termcap** file entries (used to switch between alternate screens during startup of many screen-oriented programs) from the **TERMCAP** string. If set, the **xterm** command also ignores the escape sequence to switch to the alternate screen.
translations (class **Translations**)	Specifies the key and button bindings for menus, selections, programmed strings, and so forth. For more information, seeActions" .
visualBell (class **VisualBell**)	Specifies whether a visible bell (flashing) should be used instead of an audible bell when the Ctrl+G key sequence signal is received. The default is False.

The following resources are specified as part of the **tek4014** widget (class **Tek4014**):

Item	Description
width (class **Width**)	Specifies the width of the Tektronix window in pixels.
height (class **Height**)	Specifies the height of the Tektronix window in pixels.
fontLarge (class **Font**)	Specifies the large font to use in the Tektronix window.
font2 (class **Font**)	Specifies font number 2 to use in the Tektronix window.
font3 (class **Font**)	Specifies font number 3 to use in the Tektronix window.
fontSmall (class **Font**)	Specifies the small font to use in the Tektronix window.
initialFont (class **InitialFont**)	Specifies which of the four Tektronix fonts to use initially. Values are the same as for the **set-tek-text** action. The default is large.
ginTerminator (class **GinTerminator**)	Specifies what characters should follow a GIN report or status report. The possibilities are `none,' which sends no terminating characters; CRonly, which sends CR; and CR&EOT, which sends both CR and EOT. The default is none.

The resources that may be specified for the various menus are described in the documentation for the **Athena SimpleMenu** widget. Following is a list of the names and classes of the entries in each of the menus.

The mainMenu has the following entries:

Item	Description
securekbd (class **SmeBSB**)	Invokes the **secure()** action.
allowsends (class **SmeBSB**)	Invokes the **allow-send-events(toggle)** action.
redraw (class **SmeBSB**)	Invokes the **redraw()** action.

Item	Description
line1 (class **SmeLine**)	This is a separator.
suspend (class **SmeBSB**)	Invokes the **send-signal(tstp)** action on systems that support job control.
continue (class **SmeBSB**)	Invokes the **send-signal(cont)** action on systems that support job control.
interrupt (class **SmeBSB**)	Invokes the **send-signal(int)** action.
hangup (class **SmeBSB**)	Invokes the **send-signal(hup)** action.
terminate (class **SmeBSB**)	Invokes the **send-signal(term)** action.
kill (class **SmeBSB**)	Invokes the **send-signal(kill)** action.
line2 (class **SmeLine**)	This is a separator.
quit (class **SmeBSB**)	Invokes the **quit()** action.

The vtMenu has the following entries:

Item	Description
scrollbar (class **SmeBSB**)	Invokes the **set-scrollbar(toggle)** action.
jumpscroll (class **SmeBSB**)	Invokes the **set-jumpscroll(toggle)** action.
reversevideo (class **SmeBSB**)	Invokes the **set-reverse-video(toggle)** action.
autowrap (class **SmeBSB**)	Invokes the **set-autowrap(toggle)** action.
reversewrap (class **SmeBSB**)	Invokes the **set-reversewrap(toggle)** action.
autolinefeed (class **SmeBSB**)	Invokes the **set-autolinefeed(toggle)** action.
appcursor (class **SmeBSB**)	Invokes the **set-appcursor(toggle)** action.
appkeypad (class **SmeBSB**)	Invokes the **set-appkeypad(toggle)** action.
scrollkey (class **SmeBSB**)	Invokes the **set-scroll-on-key(toggle)** action.
scrollttyoutput (class **SmeBSB**)	Invokes the **set-scroll-on-tty-output(toggle)** action.
allow132 (class **SmeBSB**)	Invokes the **set-allow132(toggle)** action.
cursesemul (class **SmeBSB**)	Invokes the **set-cursesemul(toggle)** action.
visualbell (class **SmeBSB**)	Invokes the **set-visualbell(toggle)** action.
marginbell (class **SmeBSB**)	Invokes the **set-marginbell(toggle)** action.
altscreen (class **SmeBSB**)	This entry is currently disabled.
line1 (class **SmeLine**)	This is a separator.
softreset (class **SmeBSB**)	Invokes the **soft-reset()** action.
hardreset (class **SmeBSB**)	Invokes the **hard-reset()** action.
clearsavedlines (class **SmeBSB**)	Invokes the **clear-saved-lines()** action.
line2 (class **SmeLine**)	This is a separator.
tekshow (class **SmeBSB**)	Invokes the **set-visibility(tek,toggle)** action.
tekmode (class **SmeBSB**)	Invokes the **set-terminal-type(tek)** action.
vthide (class **SmeBSB**)	Invokes the **set-visibility(vt,off)** action.

The fontMenu has the following entries:

Item	Description
fontdefault (class **SmeBSB**)	Invokes the **set-vt-font(d)** action.
font1 (class **SmeBSB**)	Invokes the **set-vt-font(1)** action.
font2 (class **SmeBSB**)	Invokes the **set-vt-font(2)** action.
font3 (class **SmeBSB**)	Invokes the **set-vt-font(3)** action.
font4 (class **SmeBSB**)	Invokes the **set-vt-font(4)** action.
font5 (class **SmeBSB**)	Invokes the **set-vt-font(5)** action.
font6 (class **SmeBSB**)	Invokes the **set-vt-font(6)** action.
fontescape (class **SmeBSB**)	Invokes the **set-vt-font(e)** action.
fontsel (class **SmeBSB**)	Invokes the **set-vt-font(s)** action.

The tekMenu has the following entries:

Item	Description
tektextlarge (class **SmeBSB**)	Invokes the **set-tek-text(l)** action.
tektext2 (class **SmeBSB**)	Invokes the **set-tek-text(2)** action.
tektext3 (class **SmeBSB**)	Invokes the **set-tek-text(3)** action.
tektextsmall (class **SmeBSB**)	Invokes the **set-tek-text(s)** action.
line1 (class **SmeLine**)	This is a separator.
tekpage (class **SmeBSB**)	Invokes the **tek-page()** action.
tekreset (class **SmeBSB**)	Invokes the **tek-reset()** action.
tekcopy (class **SmeBSB**)	Invokes the **tek-copy()** action.
line2 (class **SmeLine**)	This is a separator.
vtshow (class **SmeBSB**)	Invokes the **set-visibility(vt,toggle)** action.
vtmode (class **SmeBSB**)	Invokes the **set-terminal-type(vt)** action.
tekhide (class **SmeBSB**)	Invokes the **set-visibility(tek,toggle)** action.

The following resources are useful when specified for the **Athena Scrollbar** widget:

Item	Description
thickness (class **Thickness**)	Specifies the width in pixels of the scrollbar.
background (class **Background**)	Specifies the color to use for the background of the scrollbar.
foreground (class **Foreground**)	Specifies the color to use for the foreground of the scrollbar. The *thumb* of the scrollbar is a simple checkerboard pattern with alternating pixels for foreground and background colors.

Pointer Usage

After the VT102 window is created, the **xterm** command allows you to select text and copy it within the same or other windows.

The selection functions are invoked when the pointer buttons are used with no modifiers, and when they are used with the Shift key. The assignment of the functions to keys and buttons may be changed through the resource database.

Pointer button 1 (usually left) is used to save text into the cut buffer. Move the cursor to beginning of the text, and then hold the button down while moving the cursor to the end of the region and releasing the button. The selected text is highlighted and is saved in the global cut buffer and made the PRIMARY selection when the button is released.

Double-clicking selects by words, triple-clicking selects by lines, and quadruple-clicking goes back to characters. Multiple-click is determined by the amount of time from button up to button down, so you can change the selection unit in the middle of a selection. If the key or button bindings specify that an X selection is to be made, the **xterm** command will leave the selected text highlighted for as long as it is the selection owner.

Pointer button 2 (usually middle) "types" (pastes) the text from the PRIMARY selection, if any, otherwise from the cut buffer, inserting it as keyboard input.

Pointer button 3 (usually right) extends the current selection. If pressed while closer to the right edge of the selection than the left, it extends or contracts the right edge of the selection. If you contract the selection past the left edge of the selection, the **xterm** command assumes you really meant the left edge, restores the original selection, and then extends or contracts the left edge of the selection.

And the opposite also applies: if pressed while closer to the left edge of the selection than the right, it extends/contracts the left edge of the selection. If you contract the selection past the right edge of the selection, the **xterm** command assumes you really meant the right edge, restores the original selection, and then extends/contracts the right edge of the selection. Extension starts in the selection unit mode that the last selection or extension was performed in; you can multiple-click to cycle through them.

By cutting and pasting pieces of text without trailing new lines, you can take text from several places in different windows and form a command to the shell, for example, or take output from a program and insert it into your favorite editor. Because the cut buffer is globally shared among different applications, regard it as a "file" whose contents you know. The terminal emulator and other text programs should be treating it as if it were a text file; in other words, the text is delimited by new lines.

The scroll region displays the position and amount of text currently showing in the window (highlighted) relative to the amount of text actually saved. As more text is saved (up to the maximum), the size of the highlighted area decreases.

Clicking button 1 with the pointer in the scroll region moves the adjacent line to the top of the display window.

Clicking button 3 moves the top line of the display window down to the pointer position.

Clicking button 2 moves the display to a position in the saved text that corresponds to the pointer's position in the scrollbar.

Unlike the VT102 window, the Tektronix window does not allow the copying of text. It does allow Tektronix GIN mode, and in this mode the cursor will change from an arrow to a cross. Pressing any key will send that key and the current coordinates of the cross cursor. Pressing button one, two, or three will return the letters l, m, and r, respectively.

If the Shift key is pressed when a pointer button is pressed, the corresponding uppercase letter is sent. To distinguish a pointer button from a key, the high bit of the character is set (but this bit is usually stripped unless the terminal mode is RAW; see the **tty** command for details).

Menus

The **xterm** command has four menus, named mainMenu, vtMenu, fontMenu, and tekMenu. Each menu opens under the correct combinations of key and button presses. Most menus are divided into two section, separated by a horizontal line. The top portion contains various modes that can be altered. A check mark is displayed next to a mode that is currently active. Selecting one of these modes toggles its state. The bottom portion of the menu lists command entries; selecting one of these performs the indicated function.

The xterm menu opens when the control key and pointer button one are pressed in a window. The mainMenu contains items that apply to both the VT102 and Tektronix windows. The **Secure Keyboard** mode is used when typing in passwords or other sensitive data in an unsecure environment.

Notable entries in the command section of the menu are **Continue**, **Suspend**, **Interrupt**, **Hangup**, **Terminate**, and **Kill**, which send the **SIGCONT**, **SIGTSTP**, **SIGINT**, **SIGHUP**, **SIGTERM**, and **SIGKILL** signals, respectively, to the process group of the process running under **xterm** (usually the shell). The **Continue** function is especially useful if the user has accidentally pressed Ctrl+Z, suspending the process.

The vtMenu sets various modes in the VT102 emulation, and is opened when the control key and pointer button two are pressed in the VT102 window. In the command section of this menu, the soft reset entry will reset scroll regions. This can be convenient when some program has left the scroll regions set incorrectly (often a problem when using VMS or TOPS-20).

The full reset entry will clear the screen, reset tabs to every eight columns, and reset the terminal modes (such as wrap and smooth scroll) to their initial states just after the **xterm** command has finished processing the command-line options.

The fontMenu sets the font used in the VT102 window. In addition to the default font and a number of alternatives that are set with resources, the menu offers the font last specified by the Set Font escape sequence (See " Control Sequences") and the current selection as a font name (if the PRIMARY selection is owned).

The tekMenu sets various modes in the Tektronix emulation, and is opened when the control key and pointer button two are pressed in the Tektronix window. The current font size is checked in the Modes section of the menu. The **PAGE** entry in the command section clears the Tektronix window.

Security

X windows environments differ in their security consciousness. MIT servers, run under **xdm**, are capable of using a *magic cookie* authorization scheme that can provide a reasonable level of security for many people. If your server is only using a host-based mechanism to control access to the server (see the **xhost** command), and if you enable access for a host and other users are also permitted to run clients on that same host, there is every possibility that someone can run an application that will use the basic services of the X protocol to snoop on your activities, potentially capturing a transcript of everything you type at the keyboard.

This is of particular concern when you want to type in a password or other sensitive data. The best solution to this problem is to use a better authorization mechanism than host-based control, but a simple mechanism exists for protecting keyboard input in the **xterm** command.

The xterm menu contains a **Secure Keyboard** entry that, when enabled, ensures that all keyboard input is directed *only* to the **xterm** command (using the **GrabKeyboard** protocol request). When an application prompts you for a password (or other sensitive data), you can enable **Secure Keyboard** using the menu, type in the data, and then disable **Secure Keyboard** using the menu again.

Only one X client at a time can secure the keyboard, so when you attempt to enable **Secure Keyboard** it may fail. In this case, the bell will sound. If the **Secure Keyboard** succeeds, the foreground and background colors will be exchanged (as if you selected the **Reverse Video** entry in the Modes menu); they will be exchanged again when you exit secure mode. If the colors do *not* switch, be *very* suspicious that you are being spoofed.

If the application you are running displays a prompt before asking for the password, it is safest to enter secure mode *before* the prompt gets displayed, and to make sure that the prompt gets displayed correctly (in the new colors), to minimize the probability of spoofing. You can also bring up the menu again and make sure that a check mark is displayed next to the entry.

Secure Keyboard mode will be disabled automatically if your xterm window becomes iconified (or otherwise unmapped), or if you start up a reparenting window manager (that places a title bar or other decoration around the window) while in **Secure Keyboard** mode. (This is a feature of the X protocol not easily overcome.) When this happens, the foreground and background colors will be switched back and the bell will sound in warning.

Character Classes

Clicking the middle mouse button twice in rapid succession will cause all characters of the same class (such as letters, white space, punctuation) to be selected. Because different people have different preferences for what should be selected (for example, should file names be selected as a whole or only the separate subnames), the default mapping can be overridden through the use of the **charClass** (class **CharClass**) resource.

This resource is a series of comma-separated *range*:*value* pairs. The *range* is either a single number or *low-high* in the range of 0 to 127, corresponding to the ASCII code for the character or characters to be set. The *value* is arbitrary, although the default table uses the character number of the first character occurring in the set.

The default table is:

```
static int charClass[128] = {
/* NUL  SOH  STX  ETX  EOT  ENQ  ACK  BEL */
    32,   1,   1,   1,   1,   1,   1,   1,
/*  BS   HT   NL   VT   NP   CR   SO   SI */
     1,  32,   1,   1,   1,   1,   1,   1,
/* DLE  DC1  DC2  DC3  DC4  NAK  SYN  ETB */
     1,   1,   1,   1,   1,   1,   1,   1,
/* CAN   EM  SUB  ESC   FS   GS   RS   US */
     1,   1,   1,   1,   1,   1,   1,   1,
/*  SP    !    "    #    $    %    &    ' */
    32,  33,  34,  35,  36,  37,  38,  39,
/*   (    )    *    +    ,    -    .    / */
    40,  41,  42,  43,  44,  45,  46,  47,
/*   0    1    2    3    4    5    6    7 */
    48,  48,  48,  48,  48,  48,  48,  48,
/*   8    9    :    ;    <    =    >    ? */
    48,  48,  58,  59,  60,  61,  62,  63,
/*   @    A    B    C    D    E    F    G */
    64,  48,  48,  48,  48,  48,  48,  48,
/*   H    I    J    K    L    M    N    O */
    48,  48,  48,  48,  48,  48,  48,  48,
/*   P    Q    R    S    T    U    V    W */
    48,  48,  48,  48,  48,  48,  48,  48,
/*   X    Y    Z    [    \    ]    ^    _ */
    48,  48,  48,  91,  92,  93,  94,  48,
/*   `    a    b    c    d    e    f    g */
    96,  48,  48,  48,  48,  48,  48,  48,
/*   h    i    j    k    l    m    n    o */
    48,  48,  48,  48,  48,  48,  48,  48,
/*   p    q    r    s    t    u    v    w */
    48,  48,  48,  48,  48,  48,  48,  48,
/*   x    y    z    {    |    }    ~  DEL */
    48,  48,  48, 123, 124, 125, 126,   1};
```

For example, the string `33:48,37:48,45-47:48,64:48` indicates that the exclamation mark, percent sign, dash, period, slash, and & characters should be treated the same way as characters and numbers. This is useful for cutting and pasting electronic mailing addresses and file names.

Actions

It is possible to rebind keys (or sequences of keys) to arbitrary strings for input by changing the translations for the **vt100** or **tek4014** widgets. Changing the translations for events other than key and button events is not expected, and will cause unpredictable behavior. The following actions are provided for using within the vt100 or tek4014 translations resources:

Item	Description
bell([*Percent*]**)**	Rings the keyboard bell at the specified percentage above or below the base volume.
ignore()	Ignores the event but checks for special pointer position escape sequences.
insert()	Inserts the character or string associated with the key that was pressed.

Item	Description
insert-seven-bit()	Is a synonym for **insert()**.
insert-eight-bit()	Inserts an 8-bit (meta) version of the character or string associated with the key that was pressed. The exact action depends on the value of the **eightBitInput** resource.
insert-selection(*SourceName* [**,** ...]**)**	Inserts the string found in the selection or cutbuffer indicated by the *SourceName* parameter. Sources are checked in the order given (case is significant) until one is found. Commonly used selections include PRIMARY, SECONDARY, and CLIPBOARD. Cut buffers are typically named CUT_BUFFER0 through CUT_BUFFER7.
keymap(*Name***)**	Dynamically defines a new translation table whose resource name is *Name* with the suffix *Keymap* (case is significant). The name None restores the original translation table.
pop-up menu(*MenuName***)**	Displays the specified popup menu. Valid names (case is significant) include mainMenu, vtMenu, fontMenu, and tekMenu.
secure()	Toggles the **Secure Keyboard** mode described in the section named " Security" , and is invoked from the **securekbd** entry in mainMenu.
select-start()	Begins text selection at the current pointer location. See the section entitled " Pointer Usage" for information on making selections.
select-extend()	Tracks the pointer and extends the selection. Only bind this to Motion events.
select-end(*DestName* [**,** ...]**)**	Puts the currently selected text into all of the selections or cutbuffers specified by *DestName*.
select-cursor-start()	Is similar to **select-start** except that it begins the selection at the current text cursor position.
select-cursor-end(*DestName* [**,** ...]**)**	Is similar to **select-end** except that it should be used with **select-cursor-start**.
set-vt-font(*d/1/2/3/4/5/6/e/s* [**,***NormalFont* [**,** *BoldFont*]]**)**	Sets the font or fonts currently being used in the VT102 window. The first argument is a single character that specifies the font to be used: *d* or *D* indicates the default font (the font initially used when the **xterm** command was started), *1* through *6* indicate the fonts specified by the *font1* through *font6* resources, *e* or *E* indicates the normal and bold fonts that have been set through escape codes (or specified as the second and third action arguments, respectively), and *s* or *S* indicates the font selection (as made by programs such as the **xfontsel** program) specified by the second action argument.

Item	Description
start-extend()	Is similar to **select-start** except that the selection is extended to the current pointer location.
start-cursor-extend()	Is similar to **select-extend** except that the selection is extended to the current text cursor position.
string(*String***)**	Inserts the specified text string as if it had been typed. Quotation is necessary if the string contains white space or nonalphanumeric characters. If the string argument begins with the characters ``0x,'' it is interpreted as a hex character constant.
scroll-back(*Count* [**,***Units*]**)**	Scrolls the text window backward so that text that had previously scrolled off the top of the screen is now visible. The *Count* argument indicates the number of *Units* (which may be *page*, *halfpage*, *pixel*, or *line*) by which to scroll.
scroll-forw(*Count* [**,***Units*]**)**	Scrolls is similar to **scroll-back** except that it scrolls the other direction.
allow-send-events(*On/Off/Toggle***)**	Sets or toggles the **allowSendEvents** resource and is also invoked by the **allowsends** entry in mainMenu.
redraw()	Redraws the window and is also invoked by the **redraw** entry in mainMenu.
send-signal(*SigName***)**	Sends the signal named by *SigName* to the **xterm** subprocess (the shell or program specified with the **-e** command-line option) and is also invoked by the **suspend**, **continue**, **interrupt**, **hangup**, **terminate**, and **kill** entries in mainMenu. Allowable signal names are (case is not significant): **tstp (if supported by the operating system),** **suspend (same as tstp),** **cont (if supported by the operating system),** **int,** **hup,** **term,** **quit,** **alrm,** **alarm (same as alrm), and** **kill.**
quit()	Sends a **SIGHUP** to the subprogram and exits. It is also invoked by the **quit** entry in mainMenu.

Item	Description
set-scrollbar(*On/Off/Toggle***)**	Toggles the **scrollbar** resource and is also invoked by the **scrollbar** entry in vtMenu.
set-jumpscroll(*On/Off/Toggle***)**	Toggles the **jumpscroll** resource and is also invoked by the **jumpscroll** entry in vtMenu.

Item	Description
set-reverse-video(*On/Off/Toggle***)**	Toggles the **reverseVideo** resource and is also invoked by the **reversevideo** entry in vtMenu.
set-autowrap(*On/Off/Toggle***)**	Toggles automatic wrapping of long lines and is also invoked by the **autowrap** entry in vtMenu.
set-reversewrap(*On/Off/Toggle***)**	Toggles the **reverseWrap** resource and is also invoked by the **reversewrap** entry in vtMenu.
set-autolinefeed(*On/Off/Toggle***)**	Toggles automatic insertion of linefeeds and is also invoked by the **autolinefeed** entry in vtMenu.
set-appcursor(*On/Off/Toggle***)**	Toggles the handling Application Cursor Key mode and is also invoked by the **appcursor** entry in vtMenu.
set-appkeypad(*On/Off/Toggle***)**	Toggles the handling of Application Keypad mode and is also invoked by the **appkeypad** entry in vtMenu.
set-scroll-on-key(*On/Off/Toggle***)**	Toggles the **scrollKey** resource and is also invoked from the **scrollkey** entry in vtMenu.
set-scroll-on-tty-output(*On/Off/Toggle***)**	Toggles the **scrollTtyOutput** resource and is also invoked from the **scrollttyoutput** entry in vtMenu.
set-allow132(*On/Off/Toggle***)**	Toggles the **c132** resource and is also invoked from the **allow132** entry in vtMenu.
set-cursesemul(*On/Off/Toggle***)**	Toggles the **curses** resource and is also invoked from the **cursesemul** entry in vtMenu.
set-visual-bell(*On/Off/Toggle***)**	Toggles the **visualBell** resource and is also invoked by the **visualbell** entry in vtMenu.
set-marginbell(*On/Off/Toggle***)**	Toggles the **marginBell** resource and is also invoked from the **marginbel**l entry in vtMenu.
set-altscreen(*On/Off/Toggle***)**	Toggles between the alternate and current screens.
soft-reset()	Resets the scrolling region and is also invoked from the **softreset** entry in vtMenu.
hard-reset()	Resets the scrolling region, tabs, window size, and cursor keys and clears the screen. It is also invoked from the **hardreset** entry in vtMenu.
clear-saved-lines()	Performs **hard-reset** (see previous entry) and also clears the history of lines saved off the top of the screen. It is also invoked from the **clearsavedlines** entry in vtMenu.
set-terminal-type(*Type***)**	Directs output to either the vt or tek windows, according to the *Type* string. It is also invoked by the **tekmode** entry in vtMenu and the **vtmode** entry in tekMenu.
set-visibility(*vt/tek*, *On/Off/Toggle***)**	Controls whether or not the vt or tek windows are visible. It is also invoked from the **tekshow** and **vthide** entries in vtMenu and the **vtshow** and **tekhide** entries in tekMenu.

Item	Description
set-tek-text(*large/2/3/small***)**	Sets font used in the Tektronix window to the value of the resources **tektextlarge**, **tektext2**, **tektext3**, and **tektextsmall** according to the argument. It is also by the entries of the same names as the resources in tekMenu.
tek-page()	Clears the Tektronix window and is also invoked by the **tekpage** entry in tekMenu.
tek-reset()	Resets the Tektronix window and is also invoked by the **tekreset** entry in tekMenu.
tek-copy()	Copies the escape codes used to generate the current window contents to a file in the current directory beginning with the name **COPY**. It is also invoked from the **tekcopy** entry in tekMenu.
visual-bell()	Flashes the window quickly.

The Tektronix window also has the following action:

Item	Description
gin-press(*l/L/m/M/r/R***)**	Sends the indicated graphics input code.

The default bindings in the VT102 window are:

```
Shift <KeyPress> Prior:     scroll-back(1,halfpage) \n\
Shift <KeyPress> Next:      scroll-forw(1,halfpage) \n\
Shift <KeyPress> Select:    select-cursor-start \
                            select-cursor-end(PRIMARY,
                            CUT_BUFFER0) \n\
Shift <KeyPress> Insert:    insert-selection(PRIMARY,
                            CUT_BUFFER0) \n\
~Meta<KeyPress>:            insert-seven-bit \n\
Meta<KeyPress>:             insert-eight-bit \n\
!Ctrl <Btn1Down>:           pop-up menu(mainMenu) \n\
!Lock Ctrl <Btn1Down>:      pop-up menu(mainMenu) \n\
~Meta <Btn1Down>:           select-start \n\
~Meta <Btn1Motion>:         select-extend \n\
!Ctrl <Btn2Down>:           pop-up menu(vtMenu) \n\
!Lock Ctrl <Btn2Down>:      pop-up menu(vtMenu) \n\
~Ctrl ~Meta <Btn2Down>:     ignore \n\
~Ctrl ~Meta <Btn2Up>:       insert-selection(PRIMARY,
                            CUT_BUFFER0) \n\
!Ctrl <Btn3Down>:           pop-up menu(fontMenu) \n\
!Lock Ctrl <Btn3Down>:      pop-up menu(fontMenu) \n\
~Ctrl ~Meta <Btn3Down>:     start-extend \n\
~Meta <Btn3Motion>:         select-extend \n\
<BtnUp>:                    select-end(PRIMARY, CUT_BUFFER0) \n\
<BtnDown>:                  bell(0)
```

The default bindings in the Tektronix window are:

```
~Meta<KeyPress>:            insert-seven-bit \n\
Meta<KeyPress>:             insert-eight-bit \n\
!Ctrl <Btn1Down>:           pop-up menu(mainMenu) \n\
!Lock Ctrl <Btn1Down>:      pop-up menu(mainMenu) \n\
!Ctrl <Btn2Down>:           pop-up menu(tekMenu) \n\
!Lock Ctrl <Btn2Down>:      pop-up menu(tekMenu) \n\
Shift ~Meta<Btn1Down>:      gin-press(L) \n\
~Meta<Btn1Down>:            gin-press(l) \n\
Shift ~Meta<Btn2Down>:      gin-press(M) \n\
~Meta<Btn2Down>:            gin-press(m) \n\
Shift ~Meta<Btn3Down>:      gin-press(R) \n\
~Meta<Btn3Down>:            gin-press(r)
```

The following is an example of how the **keymap** action is used to add special keys for entering commonly typed works:

```
*VT100.Translations:          #override <Key>F13: keymap(dbx)
*VT100.dbxKeymap.translations:
 \

    <Key>F14:    keymap(None) \n\
    <Key>F17:    string("next") string(0x0d) \n\
    <Key>F18:    string("step") string(0x0d) \n\
    <Key>F19:    string("continue") string(0x0d) \n\
    <Key>F20:    string("print ")
                 insert-selection(PRIMARY,CUT_BUFFER0)
```

Environment

The **xterm** command sets the environment variables **TERM** and **TERMCAP** properly for the size window you have created. It also uses and sets the **DISPLAY** environment variable to specify which bitmap display terminal to use. The **WINDOWID** environment variable is set to the X window ID number of the xterm window.

Bugs

Large pastes do not work on some systems. This is not a bug in the **xterm** command; it is a bug in the pseudo terminal driver of those systems. The **xterm** command feeds large pastes to the pty only as fast as the pty will accept data, but some pty drivers do not return enough information to know if the write operation has succeeded.

Many of the options are not resettable after the **xterm** command starts.

Only fixed-width, character-cell fonts are supported.

Control Sequences

This section lists control sequences available for the **xterm** command.

Definitions

The following information shows how to interpret key sequences in this section.

Item	Description
c	The literal characters *c*.
C	A single (required) character.
P_s	A single (usually optional) numeric parameter, composed of one or more digits.
P_m	A multiple numeric parameter composed of any number of single numeric parameters, seperated by a **;** (semi-colon) character or characters.
P_t	A text parameter composed of printable characters.

VT100 Mode

Most of these control sequences are standard VT102 control sequences, but there are some sequences here from later DEC VT terminals, too. Major VT102 features not supported are smooth scrolling, double-size characters, flashing characters, and VT52 mode.

There are additional control sequences to provide xterm-dependent functions, like the scrollbar or window size. Where the function is specified by DEC or ISO 6429, the code assigned to it is given in parentheses. The escape codes to designate character sets are specified by ISO 2022; see that document for a discussion of character sets.

Control Sequence	Description
BEL	Bell (Ctrl+G)
BS	Backspace (Ctrl+H)

Control Sequence	Description
TAB	Horizontal Tab (HT) (Ctrl+l)
LF	Line Feed or New Line (NL) (Ctrl+J)
VT	Vertical Tab (Ctrl+K) same as LF
FF	Form Feed or New Page (NP) (Ctrl+L) same as LF
CR	Carriage return (Ctrl+M)
SO	Shift Out (Ctrl+N) –> Switch to ALternate Character Set: Invokes the G1 character set.
SI	Shift In (Ctrl+O) –> Switch to Standard Character Set: Invokes the G0 character set (the default).
ESC # 8	DEC Screen Test (DCECALN)
ESC (*C*	Designate G0 Character Set (ISO 2022) ***C* = 0** DEC Special Character and Line Drawing Set ***C* = A** United Kingdom (UK) ***C* = B** United States (USASCII)
ESC) *C*	Designate G1 Character Set (ISO 2022) ***C* = 0** DEC Special Character and Line Drawing Set ***C* = A** United Kingdom (UK) ***C* = B** United States (USASCII)
ESC * *C*	Designate G2 Character Set (ISO 2022) ***C* = 0** DEC Special Character and Line Drawing Set ***C* = A** United Kingdom (UK) ***C* = B** United States (USASCII)
ESC + *C*	Designate G3 Character Set (ISO 2022) ***C* = 0** DEC Special Character and Line Drawing Set ***C* = A** United Kingdom (UK) ***C* = B** United States (USASCII)
ESC 7	Save Cursor (DECSC)
ESC 8	Restore Cursor (DECRC)
ESC =	Application Keypad (DECPAM)
ESC >	Normal Keypad (DECNM)

Control Sequence	Description
ESC D	Index (IND)
ESC E	Next Line (NEL)
ESC H	Tab Set (HTS)
ESC M	Reverse Index (RI)
ESC N	Single Shift Select of G2 Character Set (SS2): Affects next character only.
ESC P	Single Shift Select of G3 Character Set (SS2): Affects next character only.
ESC O ***P*** $_t$ **ESC **	Device Control String (DCS). xterm implements no DCS functions; ***P*** $_t$ is ignored. ***P*** $_t$ need not be printable characters.
ESC Z	Return Terminal ID (DECID). Obsolete form of **ESC [** ***c*** (DA)
ESC [***P*** $_s$ **@**	Insert ***P*** $_s$ (Blank) Character of Characters (default=1) (ICH)
ESC [***P*** $_s$ **A**	Cursor Up ***P*** $_s$ Times (default=1) (CUU)
ESC [***P*** $_s$ **B**	Cursor Down ***P*** $_s$ Times (default=1) (CUD)
ESC [***P*** $_s$ **C**	Cursor Forward ***P*** $_s$ Times (default=1) (CUF)

Item	Description
ESC [***P*** $_s$ **D**	Cursor Backward ***P*** $_s$ Times (default=1) (CUB)
ESC [***P*** $_s$ **;** ***P*** $_s$ **H**	Cursor Position [row;column] (default=1) (CUP)
ESC [***P*** $_s$ **J**	Erase in Display (ED) ***P*** $_s$ **= 0** Clear Below (Default) ***P*** $_s$ **= 1** Clear Above ***P*** $_s$ **= 2** Clear All
ESC [***P*** $_s$ **K**	Erase in Line (EL) ***P*** $_s$ **= 0** Clear to Right (Default) ***P*** $_s$ **= 1** Clear to Left ***P*** $_s$ **= 2** Clear All
ESC [***P*** $_s$ **L**	Insert ***P*** $_s$ Lines (default=1) (IL)
ESC [***P*** $_s$ **M**	Delete ***P*** $_s$ Lines (default=1) (DL)
ESC [***P*** $_s$ **P**	Delete ***P*** $_s$ Characters (default=1) (DCH)

Item	Description
ESC [P_s ; P_s ; P_s ; P_s ; P_s T	Initiate hilite mouse tracking. Parameters are [*Func*;*Startx*;Starty;*FirstRow*;*LastRow*]. See Mouse Tracking.
ESC [P_s c	SendDevice Attributes (DA)Delete **P_s** Characters (default=1) (DCH) **P_s =0 or omitted** Request attribute from terminal **ESC [? 1 ; 2 c** ("I am a VT100 with Advanced Video Option.")
ESC [P_s ; P_s f	Horizontal and Vertical Position [row;column] (default = [1,1]) (HVP)
ESC [P_s g	Tab Clear (TBC) **P_s = 0** Clear Current Column (default) **P_s = 3** Clear All
ESC [P_m h	Set Modes (SM) **P_s = 4** Insert Mode (IRM) **P_s = 2 0** Automatic Newline (LNM)
ESC [P_m l	Reset Modes (RM) **P_s = 4** Replace Mode (IRM) **P_s = 2 0** Normal Linefeed (LNM)
ESC [P_m m	Character Attributes (SGR) **P_s = 0** Nomal (default) **P_s = 1** Bold **P_s = 4** Underscore **P_s = 5** Blink (displayed as Bold) **P_s = 7** Inverse

Item	Description
ESC [P_s **n**	Device Status Report (DSR) P_s **= 5** Status Report**ESC [0 n** ("OK") P_s **= 6** Report Cursor Position (CPR)[row;column] as**ESC [r ;** *c* **R** P_s **= 2 0** Automatic Newline (LNM)
ESC [P_s **;** P_s **r**	Set Scroll Region [top;bottom] (default = fullsize of window) (DECSTBM)
ESC [P_s **x**	Request Terminal Parameters (DECREQTPARM)

Item	Description
ESC [? P_m h	DEC Private Mode (DECSET) **P_s = 1** Application Cursor Keys (DECCKM) **P_s = 2** Designate USASCII for character sets G0–G3. (In VT102, this selects VT52 mode (DECANM), which **xterm** does not support.) **P_s = 3** 132 Column Mode (DECCOLM) **P_s = 4** Smooth (Slow) Scroll (DECSCLM) **P_s = 5** Reverse Video (DECSCNM) **P_s = 6** Origin Mode (DECOM) **P_s = 7** Wraparound Mode (DECAWM) **P_s = 8** Auto-repeat Keys (DECARM) **P_s = 9** Set Mouse X and Y on button press. See “Mouse Tracking” on page 4727. **P_s = 3 8** Enter Tektronix Mode (DECTEK) **P_s = 4 0** Allow 80 <—> 132 Mode **P_s = 4 1** **curses** function fix **P_s = 4 4** Turn On Margin Bell **P_s = 4 5** Reverse Wraparound Mode **P_s = 4 7** Use Alternate Screen Buffer (unless disabled by **titelnhibit** resource) **P_s = 1 0 0 0** Set Mouse X and Y on button press and release. See “Mouse Tracking” on page 4727. **P_s = 1 0 0 1** Use Hilite Mouse Tracking.

Item	Description
ESC [? P_m l	DEC Private Mode Reset (DECRST) **P_s = 1** Normal Cursor Keys (DECCKM) **P_s = 3** 80 Column Mode (DECCOLM) **P_s = 4** Jump Fast Scroll (DECSCLM) **P_s = 5** Normal Video (DECSCNM) **P_s = 6** Normal Cursor Mode (DECOM) **P_s = 7** No Wraparound Mode (DECAWM) **P_s = 8** No Auto-repeat Keys (DECARM) **P_s = 9** Do not Send Mouse X and Y on button press. **P_s = 4 0** Disallow 80 <—> 132 Mode **P_s = 4 1** No **curses** function fix **P_s = 4 4** Turn Off Margin Bell **P_s = 4 5** No Reverse Wraparound Mode **P_s = 4 7** Use Normal Screen Buffer **P_s = 1 0 0 0** Do not Send Mouse X and Y on button press and release. **P_s = 1 0 0 1** Do not Use Hilite Mouse Tracking. xxx
ESC [? P_m r	Restore DEC Private Mode Values. The value of **P_s** previously saved is restored. **P_s** values are the same as DECSET.
ESC [? P_m s	Save DEC Private Mode Values. **P_s** values are the same as DECSET.
ESC]? P_s ; P_t BEL	Set Text Parameters **P_s = 0** Change Icon Name and Window Title to **P_t** **P_s = 1** Change Icon Name to **P_t** **P_s = 2** Change Window Title to **P_t** **P_s = 5 0** Set Font to **P_t**

Item	Description
ESC P_t **ESC **	Private Mesage (PM). **xterm** implements no PM functions; P_t need not be printable characters.
ESC _ P_t **ESC **	Application Program Command (APC). Private Mesage (PM). **xterm** implements no APC functions; P_t is ignored. $P_t$need not be printable characters.
ESC c	Full Reset (RIS)
ESC n	Select the G2 Character Set (LS2)
ESC o	Select the G3 Character Set (LS3)
ESC l	Invoke the G3 Character Set as GR (LS3R). Has no visible effect in **xterm**.
ESC }	Invoke the G2 Character Set as GR (LS2R). Has no visible effect in **xterm**.
ESC	Invoke the G1 Character Set as GR (LS1R). Has no visible effect in **xterm**.

XTERM Description Limitation

The xterm terminal description in the DEC.TI file on AIX Version 4 provides underline mode by using the SGR attribute. The SMUL and RMUL attributes are not currently defined in the XTERM terminal description on AIX Version 4. Use the more generic capability named SGR.

```
tput sgr x y
```

Where *x* is either a 1 or a 0 to turn standout mode on or off respectively, and *y* is either a 1 or a 0 to turn underline mode on or off respectively. See the article "**terminfo** file format" for more details on the SGR capability.

```
tput sgr 0 1    turn off standout; turn on underline
tput sgr 0 0    turn off standout; turn off underline
tput sgr 1 1    turn on standout; turn on underline
tput sgr 1 0    turn on standout; turn off underline
```

Mouse Tracking

The **VT** widget can be set to send the mouse position and other information on button presses. These modes are typically used by editors and other full-screen applications that want to make use of the mouse.

There are three mutually exclusive modes, each enabled (or disabled) by a different parameter in the DECSET (or DECRST) escape sequence. Parameters for all mouse tracking escape sequences generated by the **xterm** command encode numeric parameters in a single character as *value*+040. The screen coordinate system is 1-based.

For example **!** is 1. The screen screen coordinate system is 1-based.

X10 compatibility mode sends an escape sequence on button press encoding the location and the mouse button pressed. It is enabled by specifying parameter 9 to DECSET. On button press, the **xterm** command sends the following "6 characters" . *C*b is button-1. *C*x and *C*y are the *x* and *y* coordinates of the mouse when the button was pressed.

ESC [M $C_bC_xC_y$

Normal tracking mode sends an escape sequence on both button press and release. Modifier information is also sent. It is enabled by specifying parameter 1000 to DECSET. On button press or release, the **xterm** command sends the following "key sequence" :

ESC [M $C_bC_xC_y$

The low two bits of C_b encode button information: 0=MB1 pressed, 1=MB2 pressed, 2=MB3 pressed, 3=release. The upper bits encode what modifiers were down when the button was pressed and are added together. 4=Shift, 8=Meta, 16=Control. C_x and C_y are the *x* and *y* coordinates of the mouse event. The upper left corner is (1,1).

Mouse hilite tracking notifies a program of a button press, receives a range of lines from the program, highlights the region covered by the mouse within that range until button release, and then sends the program the release coordinates. It is enabled by specifying parameter 1001 to DECSET.

Attention: Use of this mode requires a cooperating program or it will hang the **xterm** command. On button press, the same information as for normal tracking is generated; the **xterm** command then waits for the program to send mouse tracking information. *All X events are ignored until the following proper escape sequence is received from the pty*:

ESC [P_s **;** P_s **;** P_s **;** P_s **; T**

The parameters are *Func*, *Startx*, *Starty*, *FirstRow*, and *LastRow*. The *Func* parameter is nonzero to initiate hilite tracking and 0 (zero) to abort. The *Startx* and *Starty* parameters give the starting x and y location for the highlighted region. The ending location tracks the mouse, but is never above row *FirstRow* and is always above row *LastRow*. (The top of the screen is row 1.) When the button is released, the **xterm** command reports the ending position one of two ways: if the start and end coordinates are valid text locations, the **xterm** command reports the "ending position" as follows:

ESC [t C_xC_y

If either coordinate is past the end of the line, the **xterm** command reports the "ending position" as follows:

ESC [T $C_xC_yC_xC_yC_xC_y$

The parameters are *Startx*, *Starty*, *Endx*, *Endy*, *Mousex*, and *Mousey*. The *Startx*, *Starty*, *Endx*, and *Endy* parameters give the starting and ending character positions of the region. The *Mousex* and *Mousey* parameters give the location of the mouse at button up, which might not be over a character.

Tektronix 4014 Mode

Most of these sequences are standard Tektronix 4014 control sequences. The major features missing are the write-thru and defocused modes. This document does not describe the commands used in the various Tektronix plotting modes but does describe the commands to switch modes.

xwd Command

Purpose

Dumps the image of an Enhanced X-Windows window.

Syntax

xwd [**-add** *Value*] [**-frame**] [**-display** *Display*] [**-help**] [**-nobdrs**] [**-xy**] [**-out** *File*] [**-root** | **-id** *id* | **-name** *Name*] [**-icmap**] [**-screen**]

Description

The **xwd** command is an Enhanced X-Windows window dumping utility. The **xwd** command allows you to store window images in a specially formatted dump file. This file can then be read by various other X utilities that perform functions such as redisplaying, printing, editing, formatting, archiving, and image processing. Select the target window by clicking the mouse in the desired window. The keyboard bell rings once at the beginning of the dump and twice when the dump is completed.

Flags

Item	Description
-add *Value*	Specifies a signed value to add to every pixel. This option is specific to X11R5.
-frame	This option indicates that the window manager frame should be included when manually selecting a window.
-display *Display*	Specifies the server connection.
-help	Prints the usage command syntax summary.
-nobdrs	Specifies that the window dump does not include the pixels that compose the X window border. This is useful if you want to include the window contents in a document as an illustration. The result of the **-nobdrs** flag depends on which window manager is running. Many window managers remove all borders from the client. For example, the **XGetWindowAttributes** function returns the value of 0 for the `border_width` field regardless of the border width when the client was started. Therefore, any border that is visible on the screen belongs to the window manager; the client has no knowledge of it. In this case, the **-nobdrs** flag has no effect.
-out *File*	Specifies the output file on the command line. The default is to output to standard out.
-root	Indicates that the root window should be selected for the window dump, without requiring the user to select a window with the pointer. This option is specific to X11R5.
-id *id*	Indicates that the window with the specified resource id should be selected for the window dump, without requiring the user to select a window with the pointer. This option is specific to X11R5.
-name *Name*	Indicates that the window with the specified WM_NAME property should be selected for the window dump, without requiring the user to select a window with the pointer. This option is specific to X11R5.
-icmap	Forces the first installed colormap of the screen to be used to obtain RGB values. By default, the colormap of the chosen window is used. This option is specific to X11R5.
-screen	Indicates that the GetImage request used to obtain the image should be done on the root window, rather than directly on the specified window. In this way, you can obtain pieces of the other windows that overlap the specified window and, more importantly, capture menus or other popups that are independent windows but appear over the specified window. This option is specific to X11R5.
-xy	Selects xy format dumping instead of the default z format. This option applies to color displays only.

File

Item	Description
XWDFile.h	X Window dump file format definition file.

xwud Command

Purpose

Retrieves and displays the dumped image of an Enhanced X-Windows window.

Syntax

xwud [**-in** *FileName*] [**-noclick**] [**-geometry** *Geometry*] [**-display** *Display*] [**-new**] [**-std** *MapType*] [**-raw**] [**-vis** **visual_type** | **visual_id**] [**-help**] [**-rv**] [**-plane** *Number*] [**-fg** *Color*] [**-bg** *Color*]

Description

The **xwud** command retrieves the dumped image of an Enhanced X-Windows window. It does so by displaying in a window an image saved in a specially formatted dump file previously produced by the **xwd** command. The dump file format is determined by the **XWDFile.h** file.

You can use flags to specify color display, window size and position, input field, and visual class or identification. You can also select a single bit plane of the image to display.

Flags

Item	Description
-bg *Color*	Specifies the color to display for the **0** (zero) bits in the image if a bitmap image (or a single plane of an image) is displayed.
-display *Display*	Specifies the server to connect to; see the **X** command.
-fg *Color*	Specifies the color to display for the **1** bits in the image if a bitmap image (or a single plane of an image) is displayed.
-geometry *Geometry*	Specifies the size and position of the window. Typically, you will only specify the position and let the size default to the actual size of the image.
-help	Prints a short description of the allowable options.
-in *FileName*	Specifies the input file on the command line. If the input file is not specified, the standard input is assumed.
-new	Creates a new color map for displaying the image. If the image characteristics match those of the display, this flag can display the image on the screen faster, but at the cost of using a new color map (which on most terminals causes other windows to go technicolor).
-noclick	Prevents the application from ending when a button in the window is clicked. You can end the application by typing a q or Q character, or the Ctrl-C key sequence.
-plane *Number*	Selects a single bit plane of the image to display. Planes are numbered, with 0 (zero) being the least significant bit. Use this flag to determine which plane to pass to the **xpr** command for printing.

Item	Description
-raw	Displays the dumped image in whatever color values currently exist on the screen. This flag is useful when undumping an image back onto the same screen that the image originally came from, while the original windows are still on the screen. This results in getting the image on the screen faster.
-rv	Swaps the foreground and background colors if a bitmap image (or a single plane of an image) displays. This flag is useful when displaying a bitmap image that has the color sense of pixel values 0 and 1 reversed from what they are on the display.
-std *MapType*	Uses the specified Standard Colormap to display the image. You can obtain the map type by converting the type to uppercase letters, prepending **RGB_** and appending **_MAP**. Typical map types are **best**, **default**, and **gray**. See the **/usr/lpp/X11/Xamples/clients/xstdcmap** for information about creating Standard Colormaps.
-vis visual_type | **visual_id**	Specifies a particular visual type or visual id. The default picks the **best** one or you can specify **default**, which is the same class as the colormap of the root window. You can specify a particular class: **StaticGray**, **GrayScale**, **StaticColor**, **PseudoColor**, **DirectColor**, **TrueColor**. Specify **Match** to use the same class as the source image. Specify an exact visual id (specific to the server) as a hexadecimal number (prefixed with 0x) or as a decimal number. This string is not case sensitive.

Environment Variables

Item	Description
DISPLAY	Gets the default display.

Example

To retrieve a specific file from the dump window, enter:

```
xwud -in FileName
```

y

The following AIX commands begin with the with the letter *y*.

yacc Command

Purpose

Generates an LALR(1) parsing program from input consisting of a context-free grammar specification.

Syntax

yacc [**-b** *Prefix*] [**-C**] [**-d**] [**-l**] [**-Nn***Number*] [**-Nm***Number*] [**-Nr***Number*] [**-p** *Prefix*] [**-s**] [**-t**] [**-v**] [**-y** *Path*] *Grammar*

Description

The **yacc** command converts a context-free grammar specification into a set of tables for a simple automaton that executes an LALR(1) parsing algorithm. The grammar can be ambiguous; specified precedence rules are used to break ambiguities.

You must compile the output file, **y.tab.c**, with a C language compiler to produce a **yyparse** function. This function must be loaded with the **yylex** lexical analyzer, as well as with the **main** subroutine and the **yyerror** error-handling subroutine (you must provide these subroutines). The **lex** command is useful for creating lexical analyzers usable by the **yyparse** subroutine. Simple versions of **main** and **yyerror** subroutines are available through the **yacc** library, **liby.a**. Also, **yacc** can be used to generate C++ output.

You can compile the **yacc**-generated C file **(y.tab.c)** with the **-DYACC_MSG** option to include code necessary to use the Message Facility. When you use this option during compilation, error messages generated by the **yyparse** subroutine and the **YYBACKUP** macro are extracted from the **yacc_user.cat** catalog.

This allows you to receive error messages in languages other than English in non-English locales. If the catalog cannot be found or opened, the **yyparse** and **YYBACKUP** subroutines display the default English messages.

The **yacc** command is affected by the **LANG**, **LC_ALL**, **LC_CTYPE**, and **LC_MESSAGES** environment variables.

Flags

Item	Description
-b *Prefix*	Use *Prefix* instead of **y** as the prefix for all output file names. The code file **y.tab.c**, the header file **y.tab.h** (created when **-d** is specified), and the description file **y.output** (created when **-v** is specified) are changed to *Prefix*.**tab.c**, *Prefix*.**tab.h**, and *Prefix*.**output**, respectively.
-C	Produces the **y.tab.C** file instead of the **y.tab.c** file for use with a C++ compiler. To use the I/O Stream Library for input and output, define the macro, **_CPP_IOSTREAMS**.
-d	Produces the file **y.tab.h**. This contains the **#define** statements that associate the **yacc**-assigned token codes with your token names. This allows source files other than **y.tab.c** to access the token codes by including this header file.
-l	Does not include any **#line** constructs in **y.tab.c**. Use this only after the grammar and associated actions are fully debugged.

Item	Description
-Nn*Number*	Changes the size of the token and nonterminal names array to *Number*. The default value is 8000. Valid values are only those greater than 8000.
-Nm*Number*	Changes the size of the memory states array to *Number*. Default value is 40000. Valid values are only those greater than 40000.
-Nr*Number*	Changes the internal buffer sizes to handle large grammars. The default value is 2000. Valid values are only those greater than 2000.
-p *Prefix*	Use *Prefix* instead of **yy** as the prefix for all external names created by the **yacc** command. External names affected include: **yychar**, **yylval**, **yydebug**, **yyparse( )**, **yylex( )**, and **yyerror( )**. (Previously, **-p** was used to specify an alternate parser; now, **-y***Path* can be used to specify an alternate parser.)
-s	Breaks the **yyparse** function into several smaller functions. Since its size is somewhat proportional to that of the grammar, it is possible for the **yyparse** function to become too large to compile, optimize, or execute efficiently.
-t	Compiles run-time debugging code. By default, this code is not included when **y.tab.c** is compiled. However, the run-time debugging code is under the control of the preprocessor macro, **YYDEBUG**. If **YYDEBUG** has a nonzero value, the C compiler (**cc**) includes the debugging code, regardless of whether the **-t** flag is used. **YYDEBUG** should have a value of 0 if you don't want the debugging code included by the compiler. Without compiling this code, the **yyparse** subroutine will have a faster operating speed. The **-t** flag causes compilation of the debugging code, but it does not actually turn on the debug mode. To get debug output, the **yydebug** variable must be set either by adding the C language declaration, `int yydebug=1` to the declaration section of the **yacc** grammar file or by setting **yydebug** through **dbx**.
-v	Prepares the file **y.output**. It contains a readable description of the parsing tables and a report on conflicts generated by grammar ambiguities.
-y *Path*	Uses the parser prototype specified by *Path* instead of the default **/usr/lib/yaccpar** file. (Previously, **-p** was used to specify an alternate parser.)

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. The following command:

   ```
   yacc grammar.y
   ```

   draws **yacc** rules from the **grammar.y** file, and places the output in **y.tab.c**.
2. The following command:

   ```
   yacc -d grammar.y
   ```

   functions the same as example 1, but it also produces the **y.tab.h** file which would contain C-style **#define** statements for each of the tokens defined in the **grammar.y** file.

Files

Item	Description
y.output	Contains a readable description of the parsing tables and a report on conflicts generated by grammar ambiguities.
y.tab.c	Contains an output file.
y.tab.h	Contains definitions for token names.
yacc.tmp	Temporary file.
yacc.debug	Temporary file.
yacc.acts	Temporary file.
/usr/ccs/lib/yaccpar	Contains parser prototype for C programs.
/usr/ccs/lib/liby.a	Contains a run-time library.

yes Command

Purpose

Outputs an affirmative response repetitively.

Syntax

yes [*charstring*]

Description

The **yes** command outputs an affirmative response repetitively. Use the **yes** command as piped input to another command that requires an affirmative response before it completes the specified action. For example, the **yes** command is useful when deleting multiple files from a directory. The Ctl-C key sequence terminates the continuous affirmative responses.

> **Note:** The current locale is determined by the **LC_MESSAGES** environment variable or the *charstring* parameter, if specified. The *charstring* parameter can be any single character or character stream. If you enter an *charstring* parameter after issuing the **yes** command, the *charstring* parameter displays to the screen until you type the Ctl-C key sequence.

Example

To display the word `first` to the screen, type:

```
yes first
```

This statement displays the word until you enter the Ctl-C key sequence.

File

Item	Description
/usr/bin/yes	Contains the **yes** command.

ypbind Daemon

Purpose

Enables client processes to bind, or connect, to an NIS server.

Syntax

/usr/lib/netsvc/yp/ypbind [**-s -ypset -ypsetme**]

Description

The **ypbind** daemon binds, or connects, processes on a Network Information Services (NIS) client to services on an NIS server. This daemon, which runs on every NIS client, is started and stopped by the following System Resource Controller (SRC) commands:

```
startsrc -s ypbind
```

```
stopsrc -s ypbind
```

When a client requests information from a Network Information Services (NIS) map, the **ypbind** daemon broadcasts on the network for a server. When the server responds, it gives the daemon the Internet address and port number of a host. This is the host that provides the information the client is seeking. The **ypbind** daemon stores this address information in the **/var/yp/binding** directory using a file name of **domainname.version**. Then, the next time the client wants to access an NIS map, the client's **ypbind** daemon refers to the addresses in the **domainname.version** file.

The **ypbind** daemon can maintain bindings to several domains and their servers **-ypsetme** simultaneously. The default domain is the one specified by the **domainname** command at startup time.

Note:

1. If a domain becomes unbound (usually when the server crashes or is overloaded), the **ypbind** daemon broadcasts again to find another server.
2. To force a client to bind to a specific server, use the **ypset** command.
3. To find out which server a client is bound to, use the **ypwhich** command.
4. If the **/var/yp/binding/domainname/ypservers** file exists, **ypbind** will attempt to contact the servers listed in that file before broadcasting. The file should contain a list of server IP addresses, one per line.
5. By default, the NIS client will wait indefinitely for the NIS server, during which time, logins to the client system are not possible. It is possible, however, to limit the length of this wait. If the **YPBIND_MAXWAIT** environment variable is set (usually in **/etc/environment**) before the **ypbind** daemon is started, this value (in seconds) will limit the amount of time the NIS client will wait for the NIS server. If this limit is exceeded, the client behaves as if NIS were unavailable and continues using local files. This will allow local logins, such as root.
6. If a domain becomes unbound and it is listed in the `/var/yp/binding/domainnanme/ypservers` file, by default `ypbind` daemon attempts to contact the server that is currently down; however, if the `YPBIND_SKIP` environment variable is set to `1` (usually set in the `/etc/environment` file) before the `ypbind` daemon is started, the server that is currently down will not be contacted again.

Flags

Item	Description
-s	Runs the **ypbind** daemon in a secure mode on privileged communications ports.
-ypset	Indicates the local host accepts **ypset** commands from local or remote hosts.

Item	Description
-ypsetme	Indicates that the local host accepts **ypset** commands only from the local host. This flag overrides the **-ypset** flag if both are specified.

Note:

1. If neither the **-ypset** or **-ypsetme** flags are specified, the local host rejects all **ypset** commands from all hosts. This is the most secure mode because the NIS server cannot change.
2. If neither the **-ypset** or **-ypsetme** flags are specified, the local host rejects all **ypset** commands from all hosts. This is the most secure mode because the NIS server cannot change. However, if no NIS servers exist on the networks directly connected to the client machine, then the **-ypsetme** flag must be used and the NIS server should be specified with the **ypset** command.

Files

Item	Description
/var/yp/binding directory	Contains Internet addresses and port numbers for NIS servers.
/var/yp/binding/domainname/ypservers	Contains a list of internet addresses, one per line, of servers to attempt to contact before broadcasting.
domainname.version	Binary file that contains the address and port number of the current NIS server.

ypcat Command

Purpose

Prints out a Network Information Services (NIS) map.

Syntax

To Display the Network Information Services Database

/usr/bin/ypcat [**-k**] [**-t**] [**-d** *DomainName*] *MapName*

To Display the Nickname Translation Table

/usr/bin/ypcat -x

Description

The **ypcat** command prints out the Network Information Services (NIS) map you specify with the *MapName* parameter. You can specify either a map name or a map nickname. Because the **ypcat** command uses the NIS service, you do not need to specify a server.

Flags

Item	Description
-k	Displays the keys for those maps in which the values are null or for which the key is not part of the value. (None of the maps derived from files that have an ASCII version in the **/etc** directory fall into this class.)
-t	Indicates that the name specified by the *MapName* parameter is *not* a nickname. This flag causes the **ypcat** command to bypass the nickname translation table and search only for the map specified by the *MapName* parameter.

Item	Description
-d *DomainName*	Searches the specified domain for the specified map.
-x	Displays the nickname translation table. This table lists the map nicknames the command knows of and indicates the map name (as specified by the *MapName* parameter) associated with each nickname.

Examples

1. To look at the networkwide password map, **passwd.byname**, type:

   ```
   ypcat passwd
   ```

   In this example, `passwd` is the nickname for the **passwd.byname** map.
2. To locate a map, type:

   ```
   ypcat -t passwd
   ```

   In this example, the **ypcat** command bypasses any maps with the nickname of `passwd` and searches for a map with the full name of `passwd`.
3. To display a map in another domain, type:

   ```
   ypcat -d polaris passwd
   ```

   In this example, the **ypcat** command locates the map named `passwd` in the domain named `polaris`.
4. To display the map nickname translation table, type:

   ```
   ypcat -x
   ```

   In this example, the **ypcat** command displays a list of map nicknames and their associated map names.

ypinit Command

Purpose

Sets up NIS maps on a Network Information Services (NIS) server.

Syntax

To Set up NIS on an NIS Master Server

/usr/sbin/ypinit [**-o**] [**-n**] [**-q**] **-m** [*WorkerName ...*]

To Set up NIS on an NIS Worker Server

/usr/sbin/ypinit -s *MasterName*

Description

The **ypinit** command sets up NIS maps on a Network Information Services (NIS) master server or NIS worker server. Only users with root user authority can use the **ypinit** command.

By default, the **ypinit** command uses the ASCII system files as input files for the map being created.

Flags

Item	Description
-m [*WorkerName...*]	Indicates that the local host is to be the NIS master. If the **-q** flag is used the **-m** flag can be followed by the names of the machines that will be the NIS worker servers.
-n	Indicates that the **ypinit** command is not to stop if it finds errors.
-o	Allows any existing maps for the current NIS domain to be overwritten.
-q	Indicates that the **ypinit** command is to get arguments from the command line instead of prompting for input.
-s *MasterName*	Copies NIS maps from the server workstation you specify in the *MasterName* parameter.

Examples

1. To set up an NIS master server that functions as the master for all NIS maps, type the following command on the command line:

   ```
   ypinit -m
   ```

   This command invokes the **make** procedure, which follows the instructions in the **/var/yp/Makefile** file.
2. To set up an NIS worker server, type:

   ```
   ypinit -s zorro
   ```

   In this example, the **ypinit** command copies the NIS maps onto your workstation from the NIS server named `zorro`, making your workstation an NIS worker server.
3. To set up an NIS master server without being prompted for input, type:

   ```
   ypinit -o -n -q -m worker
   ```

   Note: If the system has previously been configured as an NIS master server, ensure that the directory, **/var/yp/binding**, is removed before executing **ypinit**. If old information is stored in **/var/yp/binding**, it may cause errors to occur during configuration of the NIS master server.

Files

Item	Description
/etc/bootparams	Lists clients that diskless clients can use for booting.
/etc/passwd	Contains an entry for each user that has permission to log on to the machine.
/etc/group	Contains an entry for each user group allowed to log on to the machine.
/etc/hosts	Contains an entry for each host on the network.
/var/yp/Makefile	Contains rules for making NIS maps.
/etc/networks	Contains the name of each network in the DARPA Internet.
/etc/netmasks	Lists network masks used to implement IP standard subnetting.
/etc/netid	Contains identification information for machines, hosts, and groups.
/etc/rpc	Contains map information for RPC programs.

Item	Description
/etc/services	Contains an entry for each server available through the Internet.
/etc/protocols	Defines Internet protocols used on the local host.
/etc/netgroup	Contains information about each user group on the network.
/etc/ethers	Contains the Ethernet addresses of hosts on the Internet network.
/etc/publickey	Contains public or secret keys for NIS maps.

ypmatch Command

Purpose

Displays the values of given keys within a Network Information Services (NIS) map.

Syntax

To Display Key Values for an NIS Map

/usr/bin/ypmatch [**-d** *Domain*] [**-k**] [**-t**] *Key*... *MapName*

To Display the NIS Map Nickname Table

/usr/bin/ypmatch -x

Description

The **ypmatch** command displays the values associated with one or more keys within a Network Information Services (NIS) map. Use the *MapName* parameter to specify either the name or nickname of the map you want to search.

When you specify multiple keys in the *Key* parameter, the system searches the same map for all of the keys. Because pattern matching is not available, match the capitalization and length of each key exactly. If the system does not find a match for the key or keys you specify, a diagnostic message is displayed.

Flags

Item	Description
-d *Domain*	Specifies a domain other than the default domain.
-k	Prints a key followed by a colon before printing the value of the key. This is useful only if the keys are not duplicated in the values or if you have specified so many keys that the output could be confusing.
-t	Inhibits translation of nickname to map name.
-x	Displays the map nickname table. This lists the nicknames (as specified by the *MapName* parameter) the command knows of and indicates the map name associated with each nickname.

Examples

To display the value associated with a particular key, type:

```
ypmatch -d ibm -k host1 hosts
```

In this example, the **ypmatch** command displays the value of the `host1` key from the `hosts` map in the `ibm` domain.

yppasswd Command

Purpose

Changes your network password in Network Information Services (NIS).

Syntax

yppasswd [**-f** [*Name*] | **-s** [*Name* [*ShellProg*]]]

Description

The **yppasswd** command changes (or installs) a network password and associates it with the name you specify in the *UserName* parameter. To create or change a password, you must be the owner of the password you want to change. The Network Information Services (NIS) password can be different from the one on your own machine. Root users on an NIS server can change the password of another user without knowing the user's original password. To do this, the Root user enters their password in place of the user's original password. Root users on an NIS client, however, do not have this privilege.

When you enter the **yppasswd** command on the command line, the system prompts you to enter the old password. When you do this, the system prompts you to enter the new password. The password you enter can be as small as four characters long if you use a mixture of uppercase and lowercase characters. Otherwise, the password has to be six characters long or longer. These rules are relaxed if you are insistent enough.

If you enter the old password incorrectly, you have to enter the new password before the system will give you an error message. The system requires both passwords because the **update** protocol sends them to the server at the same time. The server catches the error and notifies you that you entered the old password incorrectly.

To verify the new password, the system prompts you to enter it again. For this new password to take effect, the **yppasswdd** daemon must be running on your NIS server.

Note: The **yppasswd** command cannot establish rules for passwords as does the **passwd** command.

Flags

Item	Description
-f [*Name*]	Changes user *Name*'s gecos information in the NIS maps. Gecos information is general information stored in the **/etc/passwd** file.
-s [*Name* [*ShellProg*]]	Changes user *Name*'s login shell in the NIS maps.

Example

1. To change a user's NIS password, enter:

   ```
   yppasswd Joe
   ```

   This example demonstrates how to change the NIS password for the user named `Joe`. The system prompts you to enter Joe's old password and then his new password.
2. To change the login shell to **/bin/ksh** for the user named `Joe`, if the **yppasswdd** daemon has not been started with the **-noshell** flag, enter:

   ```
   yppasswd -s Joe /bin/ksh
   ```
3. To change the gecos information in the **passwd** file for the user named `Joe`, if the **yppasswdd** daemon has not been started with the **-nogecos** flag, enter:

```
yppasswd  -f Joe
Old NIS password:
Joe's current gecos:
John Doe Test User Id
Change (yes) or (no)? >y
To?>Joe User Test User Id
```

yppasswdd Daemon

Purpose

Receives and executes requests from the **yppasswd** command.

Syntax

rpc.yppasswdd *FileName* [**-nogecos**] [**-nopw**] [**-noshell**] [**-r** | **-m** [*Argument...*]]

Description

The **yppasswdd** daemon is a server that receives and executes requests for new passwords from the **yppasswd** command. These requests require the daemon to verify the user's old password and change it. The daemon changes the password in the file you specify in the *FileName* parameter, which has the same format as the **/etc/passwd** file.

To make it possible to update the Network Information Services (NIS) password map from remote machines, the **yppasswdd** daemon must be running on the master server that contains the NIS password map.

Note: The **yppasswdd** daemon is not run by default, nor can it be started up from the **inetd** daemon like other Remote Procedure Call (RPC) daemons.

The **yppasswdd** daemon can be started and stopped with the following System Resource Controller (SRC) commands:

```
startsrc -s yppasswdd
```

```
stopsrc -s yppasswdd
```

Flags

Item	Description
-m	Runs the **make** command using the makefile in the **/var/yp** directory. This adds the new or changed password to the NIS password map. Any arguments that follow the **-m** flag are passed to the **make** command.
-nogecos	Indicates the server will not accept changes for gecos information from the **yppasswd** command.
-nopw	Indicates that the server will not accept password changes from the **yppasswdd** command.
-noshell	Indicates the server will not accept changes for user shells from the **yppasswd** command.
-r	Directly updates the **/var/yp/***domainname***/passwd.byname** and **/var/yp/***domainname***/passwd.byuid** database files on the Master server as well as any Worker servers with new or changed passwords. This option is faster than the **-m** flag because the **make** command is not run. The **-r** flag is useful when the database files are large (several thousand entries or more).

Note: The System Resource Controller (SRC) starts the **yppasswdd** daemon with the **-m** flag specified by default. Use the **chssys** command to change the default to the **-r** flag.

Example

To propagate updated passwords immediately, invoke the **yppasswdd** daemon as follows:

```
startsrc -s yppasswdd
```

Files

Item	Description
/etc/inetd.conf	Defines how the **inetd** daemon handles Internet service requests.
/var/yp/Makefile	Contains rules for making NIS maps.
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.
/etc/security/passwd	Stores password information.

yppoll Command

Purpose

Displays the order number (ID number) of the Network Information Services (NIS) map currently in use on the server.

Syntax

/usr/sbin/yppoll [**-h** *Host*] [**-d** *Domain*] *MapName*

Description

The **yppoll** command uses the **ypserv** daemon to display the order number of the map you specify in the *MapName* parameter. An order number is a map's ID number and is assigned by the system. This number changes whenever a map is updated. Use the **yppoll** command whenever you want to make sure your servers are using the most current version of a particular map.

The **yppoll** command can run on systems that have either version 1 or version 2 of the Network Information Services (NIS) protocol installed. Be aware, however, that each version of the protocol has its own set of diagnostic messages.

Note: When specifying a *MapName*, be sure to enter the map's full name. The **yppoll** command does not recognize map nicknames.

Flags

Item	Description
-h *Host*	Enables you to specify a server other than the default server. To find out which server the command defaults to, use the **ypwhich** command.
-d *Domain*	Enables you to specify a domain other than the default domain. To find out which domain the command defaults to, use the **domainname** command.

Examples

1. To look at a map located on a particular host, type:

```
/usr/sbin/yppoll -h thor netgroups.byuser
```

In this example, the **yppoll** command displays the order number for the `netgroups.byuser` map located on the host named `thor`.

2. To look at a map on a domain, type:

```
/usr/sbin/yppoll -d atlantis hosts.byname
```

In this example, the **yppoll** command displays the order number for the `hosts.byname` map located in the domain `atlantis`.

yppush Command

Purpose

Prompts the Network Information Services (NIS) worker servers to copy updated NIS maps.

Syntax

/usr/sbin/yppush [**-v**] [**-d** *Domain*] *MapName*

Description

The **yppush** command, which is issued from the **/usr/etc/yp** directory, prompts the Network Information Services (NIS) worker servers to make copies of updated NIS maps. The *MapName* variable specifies that map to be transferred to the worker servers of the master servers. To get a list of the servers it needs to prompt, the **yppush** command reads the **ypservers** map, specified by the *Domain* parameter or the current default domain. When prompted, each worker server uses the **ypxfr** command to copy and transfer the map back to its own database.

You can use the System management interface tool (SMIT) to run this command. To use SMIT, type:

```
smit yppush
```

Note: If your system uses version 1 of the NIS protocol, the **ypxfr** command is not the transfer agent.

Flags

Item	Description
-d *Domain*	Specifies a domain other than the default domain. The maps for the specified domain must exist.
-v	Displays messages as each server is called and then displays one message for each server's response, if you are using the version 2 protocol. If this flag is omitted, the command displays error messages only. **Note:** Version 1 of the NIS protocol does not display messages. If your system uses version 1, use the **yppoll** command to verify that the transfer took place.

Examples

1. To copy a map from another domain to the worker servers, type:

```
/usr/sbin/yppush -d atlantis netgroup
```

In this example, the **yppush** command copies the `netgroup` map from the `atlantis` domain.

2. To display the in-progress status of the **yppush** command as it calls each worker server, type:

```
/usr/sbin/yppush -v -d atlantis netgroup
```

In this example, the **yppush** command displays in-progress messages as it copies the `netgroup` map from the `atlantis` domain onto each of the network's worker servers.

Files

Item	Description
/var/yp/*DomainName***/ypservers.{dir, pag}**	Lists servers that the **yppush** command prompts to make copies of updated NIS maps.

ypserv Daemon

Purpose

Looks up information in local Network Information Services (NIS) maps.

Syntax

/usr/lib/netsvc/yp/ypserv

Description

The **ypserv** daemon looks up information in its local Network Information Services (NIS) maps. The operations performed by the **ypserv** daemon are defined for the implementor by the NIS Protocol Specification and for the programmer by the **/usr/include/rpcsvc/yp_prot.h** header file. Communication with the **ypserv** daemon is by means of Remote Procedure Calls (RPC).

The **ypserv** daemon runs only on server machines. The **ypserv** daemon is started and stopped by the following System Resource Controller (SRC) commands:

```
startsrc -s ypserv
```

```
stopsrc -s ypserv
```

The **ypserv** daemon performs the following operations on a specified map within an NIS domain:

Item	Description
Match	Takes a key and returns the associated value.
Get_first	Returns the first key-value pair from the map.
Get_next	Enumerates the next key-value pair in the map.
Get_all	Ships the entire NIS map to a requestor in response to a single RPC request.
Get_order_number	Supplies information about a map instead of map entries. The order number actually exists in the map as a key-value pair, but the server does not return it through the normal lookup functions. However, the pair will be visible if you examine the map with the **makedbm** command.
Get_master_name	Supplies information about a map instead of map entries. The master name actually exists in the map as a key-value pair, but the server does not return it through the normal lookup functions. However, the pair will be visible if you examine the map with the **makedbm** command.

Log information is written to the **/var/yp/ypserv.log** file if it exists when the **ypserv** daemon starts running.

If the **/var/yp/securenets** file exists, the **ypservr** command only responds to hosts within the ip range specified in this file.

Files

Item	Description
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.
/var/yp/ypserv.log	Contains the log for the **ypserv** daemon.

ypset Command

Purpose

Directs a client machine to a specific server.

Syntax

/usr/sbin/ypset [**-V1**] [**-d** *Domain*] [**-h** *Host*] *Server*

Description

The **ypset** command directs the **ypbind** daemon on the client to the **ypserv** daemon on the server. The **ypbind** daemon goes to the server you specify in the *Server* parameter to get Network Information Services (NIS) services for the domain you specify in the *Domain* parameter. The **ypbind** daemon gets the NIS services from the **ypserv** daemon on the server.

After the binding is set, it is not tested until a client process (such as the **ypcat** command or the **ypwhich** command) tries to get a binding for the domain. If the attempt to bind fails (the specified server is down or is not running the **ypserv** daemon), the **ypbind** daemon makes another attempt to bind for the same domain.

Specify either a name or an Internet Protocol (IP) address in the *Server* parameter. If you specify a name, the **ypset** command attempts to resolve the name to an IP address through the use of the NIS service. This works only if your machine has a current valid binding for the domain in question. In most cases, you should specify the server as an IP address.

In cases where several hosts on the local network are supplying NIS services, the **ypbind** daemon can rebind to another host. If a server is down or is not running the **ypserv** daemon, the **ypbind** daemon rebinds the client to another server. In this way, the network information service balances the load among the available NIS servers.

Use the **ypset** command if the network:

- Does not support broadcasting.
- Supports broadcasting but does not have an NIS server.
- Accesses a map that exists only on a particular NIS server.

An alternative to using **ypset** is to use the **/var/yp/binding/*domain_name*/ypservers** file. This file, if present, should contain a list of NIS servers to attempt to bind to, one server per line. If the **ypbind** daemon cannot bind to any of the servers in the **ypservers** file, then it will attempt to use the server specified by **ypset**. If that fails, it will broadcast on the subnet for a NIS server.

Flags

Item	Description
-d *Domain*	Specifies a domain other than the default domain.
-h *Host*	Sets the binding for the **ypbind** daemon on the specified host instead of on the local host. The host can be specified as a name or as an IP address.
-V1	Binds the specified server for the (old) version 1 NIS protocol.

Example

To set a server to bind on a host in a particular domain, enter:

```
ypset -d ibm -h venus mars
```

In this example, the **ypset** command causes the host named `venus` to bind to the server named `mars`.

ypupdated Daemon

Purpose

Updates information in Network Information Services (NIS) maps.

Syntax

/usr/lib/netsvc/yp/rpc.ypupdated [**-i** | **-s**]

Description

The **ypupdated** daemon updates information in Network Information Services (NIS) maps. Before it can update information, however, the daemon consults the **updaters** file in the **/var/yp** directory to determine which NIS maps should be updated and how they should be changed.

By default, the **ypupdated** daemon requires the most secure method of authentication available to it, either DES (secure) or UNIX (insecure).

The **ypupdated** daemon is started and stopped by the following System Resource Controller (SRC) commands:

```
startsrc -s ypupdated

stopsrc -s ypupdated
```

Flags

Item	Description
-s	Accepts only calls authenticated using the secure Remote Procedure Call (RPC) mechanism (AUTH_DES authentication). This disables programmatic updating of NIS maps unless the network supports these calls.
-i	Accepts RPC calls with the insecure AUTH_UNIX credentials. This allows programmatic updating of NIS maps in all networks.

Examples

To start the **ypupdated** daemon from the command line, type:

```
startsrc -s ypupdated
```

File

Item	Description
/var/yp/updaters	A makefile for updating NIS maps.

ypwhich Command

Purpose

Identifies either the Network Information Services (NIS) server or the server that is the master for a given map.

Syntax

To Identify the NIS Server

/usr/bin/ypwhich [**-d** *Domain*] [**-V1** | **-V2**] [*HostName*]

To Identify the Master NIS Server for a Map

/usr/bin/ypwhich [**-t**] [**-d** *Domain*] [**-m** [*MapName*]]

To Display the Map Nickname Table

/usr/bin/ypwhich -x

Description

The **ypwhich** command identifies which server supplies Network Information Services (NIS) services or which server is the master for a map, depending on how the **ypwhich** command is invoked. If invoked without arguments, this command displays the name of the NIS server for the local machine. If you specify a host name, the system queries that host to find out which master it is using.

Flags

Item	Description
-d *Domain*	Uses the specified domain instead of the default domain.
-V1	Indicates which server is serving the old version 1 NIS protocol client processes.
-V2	Indicates which server is serving the current version 2 NIS protocol client processes. If neither version is specified, the **ypwhich** command attempts to locate the server that supplies the version 2 services. If there is no version 2 server currently bound, the **ypwhich** command then attempts to locate the server supplying version 1 services. Because servers and clients are both backward-compatible, the user need seldom be concerned about which version is currently in use.
-t	Inhibits nickname translation, which is useful if there is a map name identical to a nickname.
-m *MapName*	Finds the master NIS server for a map. No host can be specified with the **-m** flag. The *MapName* variable can be a map name or a nickname for a map. When the map name is omitted, the **-m** flag produces a list of available maps.

Item	Description
-x	Displays the map nickname table. This lists the nicknames (*MapName*) the command knows of and indicates the map name associated with each nickname.

Examples

1. To find the master server for a map, type:

   ```
   ypwhich -m passwd
   ```

   In this example, the **ypwhich** command displays the name of the server for the passwd map.
2. To find the map named passwd, rather than the map nicknamed passwd, type:

   ```
   ypwhich -t -m passwd
   ```

   In this example, the **ypwhich** command displays the name of the server for the map whose full name is passwd.
3. To find out which server serves clients that run the old version 1 of the NIS protocol, type:

   ```
   ypwhich -V1
   ```
4. To display a table of map nicknames, type:

   ```
   ypwhich -x
   ```

ypxfr Command

Purpose

Transfers a Network Information Services (NIS) map from an NIS server to a local host.

Syntax

/usr/sbin/ypxfr [**-f**] [**-c**] [**-d** *Domain*] [**-h** *Host*] [**-s** *Domain*] [**-C** *TID Program IPAddress Port*] [**-S**] *MapName*

Description

The **ypxfr** command transfers a Network Information Services (NIS) map from an NIS server to the local host as follows:

1. Creates a temporary map in the **/var/yp/***Domain* directory (which must already exist) on the client.
2. Fetches the map entries from the server and fills in the map on the client, one at a time.
3. Gets and loads the map parameters (order number and server).
4. Deletes any old versions of the map.
5. Moves the temporary map to the real map name.

If the **/var/yp/securenets** file exists, the **ypxfr** command only responds to hosts that are listed in this file.

The *MapName* variable specifies the name of a map that will be transferred from an NIS server.

If run interactively, the **ypxfr** command sends output to the user's terminal. If invoked without a controlling terminal, the **ypxfr** command appends its output to the **/var/yp/ypxfr.log** file (if the file already exists). This file records each transfer attempt and its results. The **ypxfr** command is most often invoked from the root user's **crontab** file or by the **ypserv** daemon.

To maintain consistent information between servers, use the **ypxfr** command to update every map in the NIS database periodically. Be aware though that some maps change more frequently than others and therefore need to be updated more frequently. For instance, maps that change infrequently, such as every few months, should be updated at least once a month. Maps that change frequently, such as several times a day, should be checked hourly for updates. The **services.byname** map, for example, may not change for months at a time, while the **hosts.byname** map may change several times a day.

To perform periodic updates automatically, use a **crontab** entry. To update several maps at one time, group commands together in a shell script. Examples of a shell script can be found in the **/usr/etc/yp** directory in the following files: **ypxfr_1perday**, **ypxfr_2perday**, **ypxfr_1perhour**.

You can use the System management interface tool (SMIT) to run this command. To use SMIT, enter:

```
smit ypxfr
```

Flags

Item	Description
-C *TID Program IPAddress Port*	Tells the **ypxfr** command where to find the **yppush** command. The **ypserv** daemon invokes the **ypxfr** command to call back a **yppush** command to the host. Use the parameters to indicate the following: ***TID*** Specifies the transaction ID of the **yppush** command. ***Program*** Specifies the program number associated with the **yppush** command. ***IPAddress*** Specifies the Internet Protocol address of the port where the **yppush** command resides. ***Port*** Specifies the port that the **yppush** command is listening on. **Note:** This option is only for use by the **ypserv** daemon.
-c	Prevents sending of a request to `Clear Current Map` to the local **ypserv** daemon. Use this flag if the **ypserv** daemon is not running locally at the time you are running the **ypxfr** command. Otherwise, the **ypxfr** command displays an error message and the transfer fails.
-d *Domain*	Specifies a domain other than the default domain. The maps for the specified domain must exist.
-f	Forces the transfer to occur even if the version at the master is not more recent than the local version.
-h *Host*	Gets the map from host specified, regardless of what the map says the master is. If a host is not specified, the **ypxfr** command asks the NIS service for the name of the master and tries to get the map from there. The *Host* variable can contain a name or an Internet address in the form `a.b.c.d.`
-S	Requires the **ypserv** server, from which it obtains the maps to be transferred, use *privileged* IP ports. Because only root user processes are typically allowed to use privileged ports, this feature adds an extra measure of security to the transfer. If the map being transferred is a secure map, the **ypxfr** command sets the permissions on the map to `0600.`

Item	Description
-s *Domain*	Specifies a source domain from which to transfer a map that should be the same across domains (such as the **services.byname** map).

Examples

To get a map from a host in another domain, enter:

```
/usr/sbin/ypxfr -d ibm -h venus passwd.byname
```

In this example, the **ypxfr** command gets the `passwd.byname` map from the host name `venus` in the `ibm` domain.

Files

Item	Description
/var/yp/ypxfr.log	Contains the log file.
/usr/sbin/ypxfr_1perday	Contains the script to run one transfer each day, for use with the **cron** daemons.
/usr/sbin/ypxfr_2perday	Contains the script to run two transfers each day.
/usr/sbin/ypxfr_1perhour	Contains the script for hourly transfers of volatile maps.

z

The following AIX commands begin with the with the letter *z*.

zcat Command

Purpose

Expands a compressed file to standard output.

Syntax

zcat [**-n**] [**-V**] [*File ...*]

Description

The **zcat** command allows the user to expand and view a compressed file without uncompressing that file. The **zcat** command does not rename the expanded file or remove the **.Z** extension. The **zcat** command writes the expanded output to standard output.

Flags

Item	Description
-n	Omits the compressed file header from the compressed file. **Note:** Use the **-n** option if the file was compressed using the **-n** option.
-V	Writes the current version and compile options to standard error.

Parameters

Item	Description
File ...	Specifies the compressed files to expand.

Return Values

If the **zcat** command exits with a status of 1 if any of the following events occur:

- The input file was not produced by the **compress** command.
- An input file cannot be read or an output file cannot be written.

If no error occurs, the exit status is 0.

Exit Status

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

To view the `foo.Z` file without uncompressing it, enter:

```
zcat foo.Z
```

The uncompressed contents of the `foo.Z` file are written to standard output. The file is not renamed.

zdump Command

Purpose

Displays the time zone information.

Syntax

```
zdump [--version] [--help] [-v] [-V] [-c [loyear,]hiyear] [-t [lotime,]hitime] zonename ...
```

Description

The **zdump** command prints the time zone information for each zone name that is specified in the command line.

Flags

Item	Description
--version	Displays the version information of the **zdump** command.
-v	For each zone name that is specified in the command line, the **zdump** command prints the following items: • The lowest possible value of time. • The time at one day after the lowest possible value of time. • The times both one second before and exactly at each detected discontinuity of time. • The highest possible value of time. • The time at one day before the highest possible value of time. Each line is followed by `isdst=`*D*, where *D* is a positive, zero, or negative value that depends on whether the specified time is Daylight Saving Time, standard time, or an unknown time type. Each line is also followed by `gmtoff=`*N*, where the specified local time is *N* seconds east of Greenwich.
-V	Prints an output that is similar to the output of the **-v** flag except that this flag does not display the times that are relative to the extreme time values. This flag generates output that is easier to compare with different time representations.
-c [*loyear*,]*hiyear*	Reports the verbose output for the specified years. Cutoff time is computed by using the proleptic Gregorian calendar with year 0 and Universal Time (UT) ignoring leap seconds. The lower year attribute (*loyear*) is exclusive and the upper year attribute (*hiyear*) is inclusive. For example, a *loyear* value of 1970 excludes a transition that occurs at 1970-01-01 00:00:00 Coordinated Universal Time, but a *hiyear* value of 1970 includes the transition. The default cutoff time is -500,2500.

Item	Description
-t [*lotime*,]*hitime*	Reports verbose output for a particular time that is specified in the *lotime* and *hitime* attributes in the syntax of `yyyy-mm-dd hh:mm:ss` Coordinated Universal Time (UTC). For example, 1970-01-01 00:00:00. The cutoff time includes the leap seconds depending on the *zonename* parameter. Similar to the **-c** flag, the lower time limit is exclusive and the upper time limit is inclusive.

Parameters

Item	Description
zonename	Represents the name of the zone of which the time zone information is displayed.

Exit Status

0
The command completed successfully.

>0
An error occurred.

Examples

1. To report time zone information for Singapore, enter the following command:

```
zdump -v Singapore
```

2. To display time zone information for Turkey stopping near the start of the year 2035, enter the following command:

```
zdump -v -c 2035 Turkey
```

3. To report the time zone information for New York during the years 2015 - 2017, enter the following command:

```
zdump -v -c 2015,2017 America/New_York
```

The output will be similar to the following sample:

```
America/New_York  Fri Dec 13 20:45:52 1901 UT = Fri Dec 13 15:45:52 1901 EST isdst=0
gmtoff=-18000
America/New_York  Sat Dec 14 20:45:52 1901 UT = Sat Dec 14 15:45:52 1901 EST isdst=0
gmtoff=-18000
America/New_York  Sun Mar  8 06:59:59 2015 UT = Sun Mar  8 01:59:59 2015 EST isdst=0
gmtoff=-18000
America/New_York  Sun Mar  8 07:00:00 2015 UT = Sun Mar  8 03:00:00 2015 EDT isdst=1
gmtoff=-14400
America/New_York  Sun Nov  1 05:59:59 2015 UT = Sun Nov  1 01:59:59 2015 EDT isdst=1
gmtoff=-14400
America/New_York  Sun Nov  1 06:00:00 2015 UT = Sun Nov  1 01:00:00 2015 EST isdst=0
gmtoff=-18000
America/New_York  Sun Mar 13 06:59:59 2016 UT = Sun Mar 13 01:59:59 2016 EST isdst=0
gmtoff=-18000
America/New_York  Sun Mar 13 07:00:00 2016 UT = Sun Mar 13 03:00:00 2016 EDT isdst=1
gmtoff=-14400
America/New_York  Sun Nov  6 05:59:59 2016 UT = Sun Nov  6 01:59:59 2016 EDT isdst=1
gmtoff=-14400
America/New_York  Sun Nov  6 06:00:00 2016 UT = Sun Nov  6 01:00:00 2016 EST isdst=0
gmtoff=-18000
America/New_York  Mon Jan 18 03:14:07 2038 UT = Sun Jan 17 22:14:07 2038 EST isdst=0
gmtoff=-18000
America/New_York  Tue Jan 19 03:14:07 2038 UT = Mon Jan 18 22:14:07 2038 EST isdst=0
gmtoff=-18000
```

4. To report the time zone information for New York during the years 2015 - 2017 without the lowest and highest time values, enter the following command:

```
zdump -V -c 2015,2017 America/New_York
```

The output will be similar to the following sample:

```
America/New_York  Sun Mar  8 06:59:59 2015 UT = Sun Mar  8 01:59:59 2015 EST isdst=0
gmtoff=-18000
America/New_York  Sun Mar  8 07:00:00 2015 UT = Sun Mar  8 03:00:00 2015 EDT isdst=1
gmtoff=-14400
America/New_York  Sun Nov  1 05:59:59 2015 UT = Sun Nov  1 01:59:59 2015 EDT isdst=1
gmtoff=-14400
America/New_York  Sun Nov  1 06:00:00 2015 UT = Sun Nov  1 01:00:00 2015 EST isdst=0
gmtoff=-18000
America/New_York  Sun Mar 13 06:59:59 2016 UT = Sun Mar 13 01:59:59 2016 EST isdst=0
gmtoff=-18000
America/New_York  Sun Mar 13 07:00:00 2016 UT = Sun Mar 13 03:00:00 2016 EDT isdst=1
gmtoff=-14400
America/New_York  Sun Nov  6 05:59:59 2016 UT = Sun Nov  6 01:59:59 2016 EDT isdst=1
gmtoff=-14400
America/New_York  Sun Nov  6 06:00:00 2016 UT = Sun Nov  6 01:00:00 2016 EST isdst=0
gmtoff=-18000
```

Files

Item	Description
/usr/sbin/zdump	Contains the SystemV **zdump** command.
/usr/share/lib/zoneinfo	Contains the standard time zone directory.

zic Command

Purpose

Provides a time zone compiler.

Syntax

zic [**-v**] [**-l** *LocalTime*] [**-p** *Posixrules*] [**-d** *Directory*] [**-L** *Leapseconds*] [**-y** *YearIsType*] [*FileName* ...]

Description

The zic **command** processes text from the files named on the command line and creates the time conversion binary files specified in this input. The time zone information is processed from the standard input if file name is specified as - (hyphen).

Input lines in the specified files are made up of fields. Field separators are be any number of white space characters. A pound sign (#) in the input file indicates a comment that extends to the end of the line on which the pound sign appears. White space characters and pound signs can be enclosed in double quotes (") if they are to be used as part of a field. Blank lines are ignored.

A rule line has the form:

```
Rule  NAME  FROM   TO   TYPE   IN   ON   AT   SAVE   LETTER/S
```

The fields that make up the rule line are as follows:

NAME
Provides a random name for the set of rules for which this Rule is applicable.

FROM
Provides the first year in which the rule applies. Specifying `minimum` (min) indicates the minimum year with a representable time value. Specifying `maximum` (max) indicates the maximum year with a representable time value.

TO
Provides the final year in which the rule applies. This is indicated as a valid time value or by specifying `minimum` and `maximum`. Specifying `only` is used to repeat the value of the **FROM** field.

TYPE
Provides the type of year in which the rule applies.

The **TYPE** field has the following values:

'-'
The rule applies in all years between **FROM** and **TO**, inclusively.

uspres
The rule applies in U.S. Presidential election years.

nonpres
The rule applies in years other than U.S. Presidential election years.

If the **TYPE** field has a value other than what is mentioned above, the **zic** command runs the **/usr/sbin/yearistype year type** command to check the type of year.

The **yearistype** command accepts two parameters; the year and the type of year. An exit status of 0 is taken to mean that the year is of the given type. Otherwise, 1 is returned as exit status.

IN
Represents the month in which the rule takes effect. Month names may be abbreviated.

ON
Prepresents the day on which the rule takes effect. Recognized forms include:

- **lastFri** represents the last Friday in the month.
- **lastMon** represents the last Monday in the month.
- A number representing the day of the month. For example, 5 represents the fifth of the month.
- **lastSun** represents the last Sunday in the month.
- lastMon represents the last Monday in the month.
- Sun>=8 represents first Sunday on or after the eighth.
- Sun<=25 represents last Sunday on or before the 25th.

Names of days of the week can be abbreviated or spelled out in full. Note that there must be no spaces within the **ON** field.

AT
Represents the time of day at which the rule takes effect. Recognized forms include:

- A number representing time in hours. For example, 2 indicates two hours.
- 2:00 indicates two o'clock in hours and minutes.
- 15:00 represents 3 o'clock in the afternoon using the 24-hour format time format.
- 1:28:14 indicates one o'clock, twenty-eight minutes and fourteen seconds, using the hours, minutes, seconds format.

Any of these forms may be followed by the letter `w` if the given time is local *wall-clock* time or `s` if the given time is local *standard* time. In the absence of `w` or `s`, wall-clock time is assumed.

Regions with more than two types of local time are required to use the local standard time in the **AT** field of the earliest transition time's rule to ensure the accuracy of the earliest transition time that is stored in the resulting time-zone binary.

SAVE
Prepresents the amount of time to be added to local standard time when the rule is in effect. This field has the same format as the **AT** field The `w` and `s` suffixes are not valid with this field.

LETTER/S
Provides the *variable part* of the time-zone abbreviations that are used when this rule is in effect. When this field contains `-` (hyphen), the variable is null. The `S` character is used to indicate EST and the `D` character is used to indicate EDT.

A zone line has the form:

```
Zone   NAME   GMTOFF   RULES/SAVE   FORMAT   [UNTIL]
```

The fields that make up a zone line are:

NAME
Indicates the name of the time zone. This is the name used to create the time conversion information file for the zone.

GMTOFF
Indicated the amount of time to add to GMT to get standard time in this zone. This field has the same format as the **AT** and **SAVE** fields of rule lines. Begin the field with a minus sign if time must be subtracted from GMT.

RULES/SAVE
Indicated the name of the rules that apply in the time zone or, alternately, an amount of time to add to local standard time. If value of this field is `-` (hyphen), then standard time always applies in the time zone.

FORMAT
Indicates the format for time zone abbreviations in this time zone. The `%s` characters are used to show where the variable part of the time zone abbreviation goes.

UNTIL
Indicates the time at which the GMT offset or the rules change for a location. It is specified as year, month, day, and time of day. If this is specified, the time zone information is generated from the given GMT offset and rule change until the time specified.

The next line must be a *continuation* line. The continuation line places information starting at the time specified in the **UNTIL** field of the previous line into the file used by the previous line. This line has the same format as a zone line, except that the Zone string and the name are omitted. Continuation lines can contain an **UNTIL** field, just as zone lines do, indicating that the next line is a further continuation.

A link line has the form:

```
Link    LINK-FROM    LINK-TO
```

The **LINK-FROM** field should appear as the **NAME** field in a zone line; the **LINK-TO** field is used as an alternate name for that zone.

Except for continuation lines, lines can appear in any order in the input.

The `zic` command has a limitation of compiling input containing a date before 14 December 1901 because dates before this time cannot be represented by a 32-bit `time_t` data type.

Flags

Item	Description
-d *Directory*	Creates time conversion information files in the *Directory* directory, instead of the**/usr/share/lib/zoneinfo/** standard directory .

Item	Description
-l *TimeZone*	Use the *TimeZone* time zone as local time. The **zic** command acts as if the file contained a link similar to the following: `Link  timezone  localtime`
-L *Leapseconds*	Reads the leap second information from the **leapseconds** file. If this option is not used, leap second information does not appear in the output.
-p *Posixrules*	Use the *posixrules* rules when handling POSIX-format time zone environment variables. The **zic** command acts as if the file contains a link as in this example: `Link timezone posixrules`
-v	Provides a message if a year that appears in a data file is outside the range of years representable by system time values (0:00:00 AM GMT, January 1, 1970, to 3:14:07 AM GMT, January 19, 2038).
-y *YearIsType*	Uses the given **yearistype** command rather than **/usr/sbin/yearistype** command when used to check year types.

Parameters

Item	Description
FileName	A file containing input lines that specify the time conversion information files to be created. If *FileName* is `-` (hyphen), then standard input is read.

Examples

1. A rule line can have the following format:

```
Rule  USA  1970    max    -    Sep   Sun<=14 3:00   0     S
```

2. A zone line can have the following format:

```
Zone    Turkey          3:00    Turkey          EET%s
```

3. A link line can have the following format:

```
Link    MET     CET
```

4. To compile a **timezone.infile** file containing input time zone information and place the binaries into the standard time zone **/usr/share/lib/zoneinfo/** directory, type:

```
zic timezone.infile
```

5. To compile a **timezone.infile** file containing input time zone information and place the binaries into a directory specified with **-d** option , type:

```
 zic -d tzdir timezone.infile
```

6. To report warnings during compilation of the time zone input file when the range of years are incorrect , type:

```
zic -v timezone.infile
```

7. To compile a **timezone.infile** file that contains input time zone information using the **yearistype** file specified with **-y** flag to check year types, type:

```
zic -y year timezone.infile
```

Exit Status

0
The command completed successfully.

>0
An error occurred.

Files

Item	Description
/usr/sbin/yearistype	Contains the **yearistype** command used to check year types.
/usr/sbin/zic	Contains the SystemV **zic** command.
/usr/share/lib/zoneinfo	Standard directory used for files create by the **zic** command.

Notices

This information was developed for products and services offered in the US.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (your company name) (year).

Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. _enter the year or years_.

Privacy policy considerations

IBM Software products, including software as a service solutions, (“Software Offerings”) may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as the customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at http://www.ibm.com/privacy and IBM's Online Privacy Statement at http://www.ibm.com/privacy/details the section entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at http://www.ibm.com/software/info/product-privacy.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at Copyright and trademark information at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

The registered trademark Linux is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Red Hat®, JBoss®, OpenShift®, Fedora®, Hibernate®, Ansible®, CloudForms®, RHCA®, RHCE®, RHCSA®, Ceph®, and Gluster® are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Index

Special Characters

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z